

# MITSUBISHI

Mitsubishi Programmable Controller

MELSEC **Q** series

---

## QCPU Programming Manual

Common Instruction 1/2

# QSERIES



# ● SAFETY PRECAUTIONS ●

(Always read these cautions before using the product)

Before using this product, please read this manual and the related manuals introduced in this manual, and pay full attention to safety to handle the product correctly.

Please store this manual in a safe place and make it accessible when required. Always forward a copy of the manual to the end user.

# REVISIONS

\*The manual number is given on the bottom left of the back cover.

Print Date	*Manual Number	Revision
Dec., 2008	SH (NA)-080809ENG-A	First edition
Mar., 2009	SH (NA)-080809ENG-B	<p><b>Partial corrections</b></p> <p>Section 3.3, 3.8, 5.1.3, 6.1.7, 6.2.14, 7.3.3, 7.11.18, 7.11.19, 7.12.1.5, 12.7, 7.12.11, 7.12.25, 7.12.26, 7.13.4, 7.13.5, 7.15.7, 7.15.8</p>
Jul., 2009	SH (NA)-080809ENG-C	<p>Revision because of function support by the Universal model QCPU having a serial number "11043" or later</p> <p><b>Partial corrections</b></p> <p>Section 2.1, 2.5.6, 2.5.18, 2.5.20, 7.6.9, 7.12.7, 7.12.11, 12.1.3, 12.1.4, APPENDIX 1.2, 1.3, 1.4.2, 3, 5.1</p> <p><b>Additions</b></p> <p>Section 2.5.16, 7.16, 7.18.10</p> <p><b>Modification</b></p> <p>Section 2.5.21 → 2.5.22, Section 2.5.22 → 2.5.21, Section 9.13 → 7.6.10,            Section 9.14 → 7.6.1, Section 9.15 → 7.16, Section 9.15.1 → 7.16.1, Section 9.15.2 → 7.16.2,            Section 9.15.3 → 7.16.3, Section 9.1 → 7.18.9, Section 9.2 → 7.18.11, Section 9.3 → 7.18.12,            Section 9.4 → 7.18.13, Section 9.5 → 7.18.14, Section 9.6 → 7.18.15, Section 9.7 → 7.18.16,            Section 9.8 → 7.18.17, Section 9.9 → 7.18.18, Section 9.10 → 7.18.19, Section 9.11 → 9.1,            Section 9.11.1 → 9.1.1, Section 9.11.2 → 9.1.2, Section 9.12 → 9.2, Section 9.12.1 → 9.2.1,            Chapter 10 → 11, Chapter 11 → 10</p>

Japanese Manual Version SH-080804-B

This manual confers no industrial property rights or any rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

© 2008 MITSUBISHI ELECTRIC CORPORATION

# INTRODUCTION

This manual explains the common instructions required for programming of the QCPU.

- The common instructions refer to all instructions except those dedicated to special function modules (such as AJ71QC24 and AJ71PT32-S3) and to AD57 models, as well as PID control instructions, SFC instructions and ST instructions.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the Q series programmable controller to handle the product correctly.

## ■ Relevant CPU module

CPU module	Model
Basic model QCPU	Q00JCPU, Q00CPU, Q01CPU
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
Redundant CPU	Q12PRHCPU, Q25PRHCPU
Universal model QCPU	Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU

# CONTENTS

SAFETY PRECAUTIONS .....	A - 1
REVISIONS .....	A - 2
INTRODUCTION .....	A - 3
CONTENTS .....	A - 4
MANUALS.....	A - 14

## Common Instructions 1/2

<b>1. GENERAL DESCRIPTION</b>	<b>1 - 1 to 1 - 8</b>
1.1 Related Programming Manuals	1 - 2
1.2 Abbreviations and Generic Names	1 - 5
<b>2. INSTRUCTION TABLES</b>	<b>2 - 1 to 2 - 62</b>
2.1 Types of Instructions	2 - 2
2.2 How to Read Instruction Tables	2 - 4
2.3 Sequence Instructions	2 - 6
2.3.1 Contact instructions .....	2 - 6
2.3.2 Association instructions .....	2 - 7
2.3.3 Output instructions.....	2 - 8
2.3.4 Shift instructions .....	2 - 8
2.3.5 Master control instructions.....	2 - 9
2.3.6 Termination instructions .....	2 - 9
2.3.7 Other instructions .....	2 - 9
2.4 Basic instructions	2 - 10
2.4.1 Comparison operation instructions .....	2 - 10
2.4.2 Arithmetic operation instructions .....	2 - 16
2.4.3 Data conversion instructions .....	2 - 22
2.4.4 Data transfer instructions.....	2 - 24
2.4.5 Program branch instructions.....	2 - 27
2.4.6 Program execution control instructions .....	2 - 27
2.4.7 I/O refresh instructions .....	2 - 27
2.4.8 Other convenient instructions .....	2 - 28
2.5 Application Instructions	2 - 29
2.5.1 Logical operation instructions .....	2 - 29
2.5.2 Rotation instructions .....	2 - 32
2.5.3 Shift instructions .....	2 - 33
2.5.4 Bit processing instructions.....	2 - 34
2.5.5 Data processing instructions .....	2 - 35
2.5.6 Structure creation instructions .....	2 - 38
2.5.7 Data table operation instructions.....	2 - 40
2.5.8 Buffer memory access instructions.....	2 - 41
2.5.9 Display instructions.....	2 - 41
2.5.10 Debugging and failure diagnosis instructions .....	2 - 42

2.5.11	Character string processing instructions .....	2 - 43
2.5.12	Special function instructions .....	2 - 46
2.5.13	Data control instructions .....	2 - 49
2.5.14	Switching instructions .....	2 - 51
2.5.15	Clock instructions .....	2 - 52
2.5.16	Expansion clock instruction .....	2 - 55
2.5.17	Program control instructions .....	2 - 56
2.5.18	Other instructions .....	2 - 57
2.5.19	Instructions for Data Link .....	2 - 59
2.5.20	Multiple CPU dedicated instruction .....	2 - 60
2.5.21	Multiple CPU high-speed transmission dedicated instruction .....	2 - 60
2.5.22	Redundant system instructions (For Redundant CPU) .....	2 - 61

### 3. CONFIGURATION OF INSTRUCTIONS 3 - 1 to 3 - 48

3.1	Configuration of Instructions .....	3 - 2
3.2	Designating Data .....	3 - 3
3.2.1	Using bit data .....	3 - 3
3.2.2	Using word (16 bits) data .....	3 - 4
3.2.3	Using double word data (32 bits) .....	3 - 6
3.2.4	Using real number data .....	3 - 8
3.2.5	Using character string data .....	3 - 11
3.3	Indexing .....	3 - 12
3.4	Indirect Specification .....	3 - 23
3.5	Reducing Instruction Processing Time .....	3 - 25
3.5.1	Subset Processing .....	3 - 25
3.5.2	Operation processing with standard device registers (Z) (only Universal model QCPU) .....	3 - 26
3.6	Cautions on Programming (Operation Errors) .....	3 - 27
3.7	Conditions for Execution of Instructions .....	3 - 33
3.8	Counting Step Number .....	3 - 34
3.9	Operation when the OUT, SET/RST, or PLS/PLF Instructions Use the Same Device .....	3 - 39
3.10	Precautions for Use of File Registers .....	3 - 44

### 4. HOW TO READ INSTRUCTIONS 4 - 1 to 4 - 4

### 5. SEQUENCE INSTRUCTIONS 5 - 1 to 5 - 60

5.1	Contact Instructions .....	5 - 2
5.1.1	Operation start, series connection, parallel connection (LD,LDI,AND,ANI,OR,ORI) ....	5 - 2
5.1.2	Pulse operation start, pulse series connection, pulse parallel connection (LDP,LDF,ANDP,ANDF,ORP,ORF) .....	5 - 5
5.1.3	Pulse NOT operation start, pulse NOT series connection, pulse NOT parallel connection (LDPI,LDFI,ANDPI,ANDFI,ORPI,ORFI) .....	5 - 7
5.2	Association Instructions .....	5 - 10
5.2.1	Ladder block series connection and parallel connection (ANB,ORB) .....	5 - 10
5.2.2	Operation results push,read,pop (MPS,MRD,MPP) .....	5 - 12

5.2.3	Operation results inversion (INV) .....	5 - 15
5.2.4	Operation result conversions (MEP,MEF) .....	5 - 17
5.2.5	Pulse conversions of edge relay operation results (EGP,EGF).....	5 - 18
5.3	<b>Output Instructions</b> .....	5 - 20
5.3.1	Out instruction (excluding timers, counters, and annunciators) (OUT).....	5 - 20
5.3.2	Timers (OUT T,OUTH T) .....	5 - 22
5.3.3	Counter (OUT C) .....	5 - 26
5.3.4	Annunciator output (OUT F) .....	5 - 28
5.3.5	Setting devices (except for annunciators) (SET) .....	5 - 30
5.3.6	Resetting devices (except for annunciators) (RST).....	5 - 32
5.3.7	Setting and resetting the annunciators (SET F,RST F) .....	5 - 35
5.3.8	Leading edge and trailing edge outputs (PLS,PLF).....	5 - 37
5.3.9	Bit device output reverse (FF) .....	5 - 40
5.3.10	Pulse conversions of direct outputs (DELTA(P)) .....	5 - 42
5.4	<b>Shift Instructions</b> .....	5 - 44
5.4.1	Bit device shifts (SFT(P)).....	5 - 44
5.5	<b>Master Control Instructions</b> .....	5 - 47
5.5.1	Setting and resetting the master control (MC,MCR).....	5 - 47
5.6	<b>Termination Instructions</b> .....	5 - 51
5.6.1	End main routine program (FEND).....	5 - 51
5.6.2	End sequence program (END) .....	5 - 53
5.7	<b>Other instructions</b> .....	5 - 55
5.7.1	Sequence program stop (STOP) .....	5 - 55
5.7.2	No operations (NOP,NOPLF,PAGE n) .....	5 - 57

## 6. BASIC INSTRUCTIONS 6 - 1 to 6 - 168

6.1	<b>Comparison Operation Instructions</b> .....	6 - 2
6.1.1	BIN 16-bit data comparisons (=,<>,>,<=,<,>=) .....	6 - 2
6.1.2	BIN 32-bit data comparisons (D=D<>,D>,D<=,D<,D>=) .....	6 - 4
6.1.3	Floating decimal point data comparisons (Single precision) (E=E,<>,E>,E<=,E<,E>=).....	6 - 6
6.1.4	Floating decimal point data comparisons (Double precision) (ED=,ED<>,ED>,ED<=,ED<,ED>=) .....	6 - 8
6.1.5	Character string data comparisons (\$=,\$<>,\$>,\$<=,\$<,\$>=) .....	6 - 11
6.1.6	BIN block data comparisons (BKCMP <input type="checkbox"/> ,BKCMP <input type="checkbox"/> P) .....	6 - 15
6.1.7	BIN 32-bit block data comparisons (DBKCMP <input type="checkbox"/> ,DBKCMP <input type="checkbox"/> P).....	6 - 18
6.2	<b>Arithmetic Operation Instructions</b> .....	6 - 22
6.2.1	BIN 16-bit addition and subtraction operations (+(P),-(P)) .....	6 - 22
6.2.2	BIN 32-bit addition and subtraction operations (D+(P),D-(P)) .....	6 - 26
6.2.3	BIN 16-bit multiplication and division operations (*(P),/(P)).....	6 - 30
6.2.4	BIN 32-bit multiplication and division operations (D*(P),D/(P)) .....	6 - 32
6.2.5	BCD 4-digit addition and subtraction operations (B+(P),B-(P)) .....	6 - 34
6.2.6	BCD 8-digit addition and subtraction operations (DB+(P),DB-(P)).....	6 - 38
6.2.7	BCD 4-digit multiplication and division operations (B*(P),B/(P)) .....	6 - 42
6.2.8	BCD 8-digit multiplication and division operations (DB*(P),DB/(P)) .....	6 - 44
6.2.9	Addition and subtraction of floating decimal point data (Single precision) (E+(P),E-(P)) .....	6 - 46



6.2.10	Addition and subtraction of floating decimal point data (Double precision) (ED+(P),ED-(P)) .....	6 - 50
6.2.11	Multiplication and division of floating decimal point data (Single precision) (E*(P),E/(P)) .....	6 - 54
6.2.12	Multiplication and division of floating decimal point data (Double precision) (ED*(P),ED/(P)) .....	6 - 56
6.2.13	Block addition and subtraction (BK+(P),BK-(P)).....	6 - 59
6.2.14	BIN 32-bit data block addition and subtraction operations (DBK+(P),DBK-(P)) .....	6 - 62
6.2.15	Linking character strings (\$(P)) .....	6 - 65
6.2.16	Incrementing and decrementing 16-bit BIN data (INC(P),DEC(P)) .....	6 - 69
6.2.17	Incrementing and decrementing 32-bit BIN data (DINC(P),DDEC(P)) .....	6 - 71
<b>6.3</b>	<b>Data conversion instructions</b> .....	<b>6 - 73</b>
6.3.1	Conversion from BIN data to 4-digit and 8-digit BCD (BCD(P),DBCD(P)) .....	6 - 73
6.3.2	Conversion from BCD 4-digit and 8-digit data to BIN data (BIN(P),DBIN(P)) .....	6 - 75
6.3.3	Conversion from BIN 16 and 32-bit data to floating decimal point (Single precision) (FLT(P),DFLT(P)) .....	6 - 78
6.3.4	Conversion from BIN 16 and 32-bit data to floating decimal point (Double precision) (FLTD(P),DFLTD(P)).....	6 - 81
6.3.5	Conversion from floating decimal point data to BIN16- and 32-bit data (Single precision) (INT(P),DINT(P)) .....	6 - 83
6.3.6	Conversion from floating decimal point data to BIN16- and 32-bit data (Double precision) (INTD(P),DINTD(P)).....	6 - 86
6.3.7	Conversion from BIN 16-bit to BIN 32-bit data (DBL(P)).....	6 - 88
6.3.8	Conversion from BIN 32-bit to BIN 16-bit data (WORD(P)).....	6 - 89
6.3.9	Conversion from BIN 16 and 32-bit data to Gray code (GRY(P),DGRY(P)) .....	6 - 90
6.3.10	Conversion of Gray code to BIN 16 and 32-bit data (GBIN(P),DGBIN(P)).....	6 - 92
6.3.11	Complement of 2 of BIN 16- and 32-bit data (sign reversal) (NEG(P),DNEG(P)) .....	6 - 94
6.3.12	Floating-point sign inversion (Single precision) (ENEG(P)) .....	6 - 96
6.3.13	Floating-point sign inversion (Double precision) (EDNEG(P)) .....	6 - 97
6.3.14	Conversion from block BIN 16-bit data to BCD 4-digit data (BKBCD(P)).....	6 - 98
6.3.15	Conversion from block BCD 4-digit data to block BIN 16-bit data (BKBIN(P)).....	6 - 100
6.3.16	Single precision to Double precision conversion (ECON(P)) .....	6 - 102
6.3.17	Double precision to Single precision conversion (EDCON(P)).....	6 - 104
<b>6.4</b>	<b>Data Transfer Instructions</b> .....	<b>6 - 106</b>
6.4.1	16-bit and 32-bit data transfers (MOV(P),DMOV(P)).....	6 - 106
6.4.2	Floating-point data transfer (Single precision) (EMOV(P)) .....	6 - 108
6.4.3	Floating-point data transfer (Double precision) (EDMOV(P)) .....	6 - 110
6.4.4	Character string transfers (\$MOV(P)).....	6 - 112
6.4.5	16-bit and 32-bit negation transfers (CML(P),DCML(P)).....	6 - 114
6.4.6	Block 16-bit data transfers (BMOV(P)) .....	6 - 117
6.4.7	Identical 16-bit data block transfers (FMOV(P)) .....	6 - 120
6.4.8	Identical 32-bit data block transfers (DFMOV(P)).....	6 - 122
6.4.9	16-bit and 32-bit data exchanges (XCH(P),DXCH(P)) .....	6 - 124
6.4.10	Block 16-bit data exchanges (BXCH(P)) .....	6 - 126
6.4.11	Upper and lower byte exchanges (SWAP(P)) .....	6 - 128
<b>6.5</b>	<b>Program Branch Instructions</b> .....	<b>6 - 129</b>
6.5.1	Pointer branch instructions (CJ,SCJ,JMP) .....	6 - 129
6.5.2	Jump to END (GOEND).....	6 - 132

6.6	Program Execution Control Instructions	6 - 133
6.6.1	Interrupt disable/enable instructions, interrupt program mask (DI,EI,IMASK) .....	6 - 133
6.6.2	Recovery from interrupt programs (IRET) .....	6 - 139
6.7	I/O Refresh Instructions	6 - 141
6.7.1	I/O refresh (RFS(P)) .....	6 - 141
6.8	Other Convenient Instructions	6 - 143
6.8.1	Counter 1-phase input up or down (UDCNT1) .....	6 - 143
6.8.2	Counter 2-phase input up or down (UDCNT2) .....	6 - 146
6.8.3	Teaching timer (TTMR) .....	6 - 149
6.8.4	Special function timer (STMR).....	6 - 151
6.8.5	Rotary table shortest direction control (ROTC) .....	6 - 154
6.8.6	Ramp signal (RAMP).....	6 - 157
6.8.7	Pulse density measurement (SPD) .....	6 - 160
6.8.8	Fixed cycle pulse output (PLSY) .....	6 - 162
6.8.9	Pulse width modulation (PWM) .....	6 - 164
6.8.10	Matrix input (MTR).....	6 - 166

## 7. APPLICATION INSTRUCTIONS 7 - 1 to 7 - 452

7.1	Logical operation instructions	7 - 2
7.1.1	Logical products with 16-bit and 32-bit data (WAND(P),DAND(P)).....	7 - 3
7.1.2	Block logical products (BKAND(P)) .....	7 - 9
7.1.3	Logical sums of 16-bit and 32-bit data (WOR(P),DOR(P)).....	7 - 11
7.1.4	Block logical sum operations (BKOR(P)).....	7 - 17
7.1.5	16-bit and 32-bit exclusive OR operations (WXOR(P),DXOR(P)).....	7 - 19
7.1.6	Block exclusive OR operations (BKXOR(P)).....	7 - 25
7.1.7	16-bit and 32-bit data exclusive NOR operations (WXNR(P),DXNR(P)).....	7 - 27
7.1.8	Block exclusive NOR operations (BKXNR(P)).....	7 - 33
7.2	Rotation instruction	7 - 35
7.2.1	Right rotation of 16-bit data (ROR(P),RCR(P)) .....	7 - 35
7.2.2	Left rotation of 16-bit data (ROL(P),RCL(P)) .....	7 - 38
7.2.3	Right rotation of 32-bit data (DROR(P),DRCR(P)) .....	7 - 41
7.2.4	Left rotation of 32-bit data (DROL(P),DRCL(P)).....	7 - 44
7.3	Shift instruction	7 - 46
7.3.1	n-bit shift to right or left of 16-bit data (SFR(P),SFL(P)) .....	7 - 46
7.3.2	1-bit shift to right or left of n-bit data (BSFR(P),BSFL(P)) .....	7 - 49
7.3.3	n-bit shift to right or left of n-bit data (SFTBR(P),SFTBL(P)) .....	7 - 51
7.3.4	1-word shift to right or left of n-word data (DSFR(P),DSFL(P)).....	7 - 54
7.3.5	n-bit shift to right or left of n-word data (SFTWR(P),SFTWL(P)).....	7 - 56
7.4	Bit processing instructions	7 - 59
7.4.1	Bit set and reset for word devices (BSET(P),BRST(P)) .....	7 - 59
7.4.2	Bit tests (TEST(P),DTEST(P)).....	7 - 61
7.4.3	Batch reset of bit devices (BKRST(P)) .....	7 - 64
7.5	Data processing instructions	7 - 66
7.5.1	16-bit and 32-bit data searches (SER(P),DSER(P)).....	7 - 66
7.5.2	16-bit and 32-bit data checks (SUM(P),DSUM(P)).....	7 - 69
7.5.3	Decoding from 8 to 256 bits (DECO(P)) .....	7 - 71
7.5.4	Encoding from 256 to 8 bits (ENCO(P)) .....	7 - 73

7.5.5	7-segment decode (SEG(P)).....	7 - 75
7.5.6	4-bit dissociation of 16-bit data (DIS(P)).....	7 - 77
7.5.7	4-bit linking of 16-bit data (UNI(P)).....	7 - 79
7.5.8	Dissociation or linking of random data (NDIS(P),NUNI(P)).....	7 - 81
7.5.9	Data dissociation and linking in byte units (WTOB(P),BTOW(P)).....	7 - 85
7.5.10	Maximum value search for 16- and 32-bit data (MAX(P),DMAX(P)).....	7 - 89
7.5.11	Minimum value search for 16- and 32-bit data (MIN(P),DMIN(P)).....	7 - 92
7.5.12	BIN 16 and 32 bits data sort operations (SORT,DSORT).....	7 - 95
7.5.13	Calculation of totals for 16-bit data (WSUM(P)).....	7 - 99
7.5.14	Calculation of totals for 32-bit data (DWSUM(P)).....	7 - 101
7.5.15	Calculation of averages for 16-bit or 32-bit data (MEAN(P),DMEAN(P)).....	7 - 103
<b>7.6</b>	<b>Structure creation instructions</b>	<b>7 - 105</b>
7.6.1	FOR to NEXT instruction loop (FOR,NEXT).....	7 - 105
7.6.2	Forced end of FOR to NEXT instruction loop (BREAK(P)).....	7 - 108
7.6.3	Subroutine program calls (CALL(P)).....	7 - 110
7.6.4	Return from subroutine programs (RET).....	7 - 115
7.6.5	Subroutine program output OFF calls (FCALL(P)).....	7 - 116
7.6.6	Subroutine calls between program files (ECALL(P)).....	7 - 120
7.6.7	Subroutine output OFF calls between program files (EFCALL(P)).....	7 - 125
7.6.8	Subroutine program call (XCALL).....	7 - 129
7.6.9	Refresh instruction (COM).....	7 - 134
7.6.10	Select Refresh Instruction (COM).....	7 - 137
7.6.11	Select Refresh Instruction (CCOM).....	7 - 141
7.6.12	Index modification of entire ladder (IX,IXEND).....	7 - 144
7.6.13	Designation of modification values in index modification of entire ladders (IXDEV,IXSET).....	7 - 148
<b>7.7</b>	<b>Data Table Operation Instructions</b>	<b>7 - 151</b>
7.7.1	Writing data to the data table (FIFW(P)).....	7 - 151
7.7.2	Reading oldest data from tables (FIFR(P)).....	7 - 153
7.7.3	Reading newest data from data tables (FPOP(P)).....	7 - 155
7.7.4	Deleting and inserting data from and in data tables (FDEL(P),FINS(P)).....	7 - 157
<b>7.8</b>	<b>Buffer memory access instruction</b>	<b>7 - 160</b>
7.8.1	Reading 1/2-word data from the intelligent function module (FROM(P),DFRO(P)).....	7 - 160
7.8.2	Writing 1/2-word data to intelligent function module (TO(P),DTO(P)).....	7 - 163
<b>7.9</b>	<b>Display instructions</b>	<b>7 - 166</b>
7.9.1	Print ASCII code instruction (PR).....	7 - 166
7.9.2	Print comment instruction (PRC).....	7 - 169
7.9.3	Error display and annunciator reset instruction (LEDR).....	7 - 172
<b>7.10</b>	<b>Debugging and failure diagnosis instructions</b>	<b>7 - 175</b>
7.10.1	Special format failure checks (CHKST,CHK).....	7 - 175
7.10.2	Changing check format of CHK instruction (CHKCIR,CHKEND).....	7 - 179
<b>7.11</b>	<b>Character string processing instructions</b>	<b>7 - 183</b>
7.11.1	Conversion from BIN 16-bit or 32-bit to decimal ASCII (BINDA(P),DBINDA(P)).....	7 - 183
7.11.2	Conversion from BIN 16-bit or 32-bit data to hexadecimal ASCII (BINHA(P),DBINHA(P)).....	7 - 186

7.11.3	Conversion from BCD 4-digit and 8-digit to decimal ASCII data (BCDDA(P),DBCDDA(P)).....	7 - 189
7.11.4	Conversion from decimal ASCII to BIN 16-bit and 32-bit data (DABIN(P),DDABIN(P)).....	7 - 192
7.11.5	Conversion from hexadecimal ASCII to BIN 16-bit and 32-bit data (HABIN(P),DHABIN(P)).....	7 - 195
7.11.6	Conversion from decimal ASCII to BCD 4-digit or 8-digit data (DABCD(P),DDABCD(P)).....	7 - 198
7.11.7	Reading device comment data (COMRD(P)) .....	7 - 201
7.11.8	Character string length detection (LEN(P)) .....	7 - 204
7.11.9	Conversion from BIN 16-bit or 32-bit to character string (STR(P),DSTR(P)) .....	7 - 206
7.11.10	Conversion from character string to BIN 16-bit or 32-bit data (VAL(P),DVAL(P)) ....	7 - 212
7.11.11	Conversion from floating decimal point to character string data (ESTR(P)).....	7 - 217
7.11.12	Conversion from character string to floating decimal point data (EVAL(P)) .....	7 - 224
7.11.13	Conversion from hexadecimal BIN to ASCII (ASC(P)) .....	7 - 228
7.11.14	Conversion from ASCII to hexadecimal BIN (HEX(P)).....	7 - 230
7.11.15	Extracting character string data from the right or left (RIGHT(P),LEFT(P)).....	7 - 232
7.11.16	Random selection from and replacement in character strings (MIDR(P),MIDW(P)) .....	7 - 235
7.11.17	Character string search (INSTR(P)) .....	7 - 239
7.11.18	Insertion of character string (STRINS(P)).....	7 - 241
7.11.19	Deletion of character string (STRDEL(P)) .....	7 - 243
7.11.20	Floating decimal point to BCD (EMOD(P)).....	7 - 245
7.11.21	From BCD format data to floating decimal point (EREXP(P)) .....	7 - 248
7.12	Special function instructions	7 - 250
7.12.1	SIN operation on floating-point data (Single precision) (SIN(P)).....	7 - 250
7.12.2	SIN operation on floating-point data (Double precision) (SIND(P)).....	7 - 252
7.12.3	COS operation on floating-point data (Single precision) (COS(P)) .....	7 - 254
7.12.4	COS operation on floating-point data (Double precision) (COSD(P)) .....	7 - 256
7.12.5	TAN operation on floating-point data (Single precision) (TAN(P)).....	7 - 258
7.12.6	TAN operation on floating-point data (Double precision) (TAND(P)).....	7 - 260
7.12.7	$\text{SIN}^{-1}$ operation on floating point data (Single precision) (ASIN(P)) .....	7 - 262
7.12.8	$\text{SIN}^{-1}$ operation on floating-point data (Double precision) (ASIND(P)) .....	7 - 265
7.12.9	$\text{COS}^{-1}$ operation on floating-point data (Single precision) (ACOS(P)) .....	7 - 267
7.12.10	$\text{COS}^{-1}$ operation on floating-point data (Double precision) (ACOSD(P)) .....	7 - 269
7.12.11	$\text{TAN}^{-1}$ operation on floating-point data (Single precision) (ATAN(P)).....	7 - 271
7.12.12	$\text{TAN}^{-1}$ operation on floating-point data (Double precision) (ATAND(P)).....	7 - 273
7.12.13	Conversion from floating-point angle to radian (Single precision) (RAD(P)) .....	7 - 275
7.12.14	Conversion from floating-point angle to radian (Double precision) (RADD(P)) .....	7 - 277
7.12.15	Conversion from floating-point radian to angle (Single precision) (DEG(P)).....	7 - 279
7.12.16	Conversion from floating-point radian to angle (Double precision) (DEGD(P)) .....	7 - 281
7.12.17	Exponentiation operation on floating-point data (Single precision) (POW(P)).....	7 - 283
7.12.18	Exponentiation operation on floating-point data (Single precision) (POWD(P)).....	7 - 285
7.12.19	Square root operation for floating-point data (Single precision) (SQR(P)) .....	7 - 287
7.12.20	Square root operation for floating-point data (Double precision) (SQRD(P)) .....	7 - 289
7.12.21	Exponent operation on floating-point data (Single precision) (EXP(P)).....	7 - 291
7.12.22	Exponent operation on floating-point data (Double precision) (EXPD(P)).....	7 - 294
7.12.23	Natural logarithm operation on floating-point data (Single precision) (LOG(P)) .....	7 - 296
7.12.24	Natural logarithm operation on floating-point data (Double precision) (LOGD(P))... ..	7 - 298

7.12.25 Common logarithm operation on floating-point data (Single precision) (LOG10(P)).....	7 - 300
7.12.26 Common logarithm operation on floating-point data (Double precision) (LOG10D(P)).....	7 - 302
7.12.27 Random number generation and series updates (RND(P),SRND(P)) .....	7 - 304
7.12.28 BCD 4-digit and 8-digit square roots (BSQR(P),BDSQR(P)) .....	7 - 306
7.12.29 BCD type SIN operation (BSIN(P)).....	7 - 309
7.12.30 BCD type COS operations (BCOS(P)) .....	7 - 311
7.12.31 BCD type TAN operation (BTAN(P)) .....	7 - 313
7.12.32 BCD type SIN <sup>-1</sup> operations (BASIN(P)).....	7 - 315
7.12.33 BCD type COS <sup>-1</sup> operation (BACOS(P)).....	7 - 317
7.12.34 BCD type TAN <sup>-1</sup> operations (BATAN(P)) .....	7 - 319
<b>7.13 Data Control Instructions</b>	<b>7 - 321</b>
7.13.1 Upper and lower limit controls for BIN 16-bit and BIN 32-bit data (LIMIT(P),DLIMIT(P)) .....	7 - 321
7.13.2 BIN 16-bit and 32-bit dead band controls (BAND(P),DBAND(P)) .....	7 - 324
7.13.3 Zone control for BIN 16-bit and BIN 32-bit data (ZONE(P),DZONE(P)).....	7 - 327
7.13.4 Scaling (Point-by-point coordinate data) (SCL(P),DSCL(P)).....	7 - 330
7.13.5 Scaling (Point-by-point coordinate data) (SCL2(P),DSCL2(P)).....	7 - 334
<b>7.14 File register switching instructions</b>	<b>7 - 337</b>
7.14.1 Switching file register numbers (RSET(P)).....	7 - 337
7.14.2 Setting files for file register use (QDRSET(P)) .....	7 - 339
7.14.3 File setting for comments (QCDSSET(P)) .....	7 - 342
<b>7.15 Clock instructions</b>	<b>7 - 344</b>
7.15.1 Reading clock data (DATERD(P)).....	7 - 344
7.15.2 Writing clock data (DATEWR(P)) .....	7 - 346
7.15.3 Clock data addition operation (DATE+(P)).....	7 - 348
7.15.4 Clock data subtraction operation (DATE-(P)).....	7 - 350
7.15.5 Time data conversion (from Hour/Minute/Second to Second) (SECOND(P)).....	7 - 352
7.15.6 Time data conversion (from Second to Hour/Minute/Second ) (HOUR(P)).....	7 - 354
7.15.7 Date comparison (DT=,DT<>,DT>,DT<=,DT<,DT>=) .....	7 - 356
7.15.8 Clock comparison (TM=,TM<>,TM>,TM<=,TM<,TM>=).....	7 - 361
<b>7.16 Expansion Clock Instructions</b>	<b>7 - 366</b>
7.16.1 Reading expansion clock data (S(P).DATERD) .....	7 - 366
7.16.2 Expansion clock data addition operation (S(P).DATE+).....	7 - 369
7.16.3 Expansion clock data subtraction operation (S(P).DATE-).....	7 - 372
<b>7.17 Program control instructions</b>	<b>7 - 375</b>
7.17.1 Program standby instruction (PSTOP(P)) .....	7 - 377
7.17.2 Program output OFF standby instruction (POFF(P)).....	7 - 378
7.17.3 Program scan execution registration instruction (PSCAN(P)) .....	7 - 380
7.17.4 Program low speed execution registration instruction (PLOW(P)) .....	7 - 382
7.17.5 Program execution status check instruction (PCHK).....	7 - 384
<b>7.18 Other instructions</b>	<b>7 - 386</b>
7.18.1 Resetting watchdog timer (WDT(P)).....	7 - 386
7.18.2 Timing pulse generation (DUTY) .....	7 - 388
7.18.3 Time check instruction (TIMCHK).....	7 - 390
7.18.4 Direct 1-byte read from file register (ZRRDB(P)).....	7 - 391

7.18.5	File register direct 1-byte write (ZRWRB(P))	7 - 393
7.18.6	Indirect address read operations (ADRSET(P))	7 - 395
7.18.7	Numerical key input from keyboard (KEY)	7 - 396
7.18.8	Batch save or recovery of index register (ZPUSH(P),ZPOP(P))	7 - 400
7.18.9	Reading Module Information (UNIRD(P))	7 - 402
7.18.10	Reading module model name(TYPERD(P))	7 - 406
7.18.11	Trace Set/Reset (TRACE,TRACER)	7 - 411
7.18.12	Writing Data to Designated File (SP.FWRITE)	7 - 413
7.18.13	Reading Data from Designated File (SP.FREAD)	7 - 424
7.18.14	Writing Data to Standard ROM (SP.DEVST)	7 - 436
7.18.15	Read Data from Standard ROM (S(P).DEVLD)	7 - 438
7.18.16	Load Program from Memory Card (PLOADP)	7 - 440
7.18.17	Unload Program from Program Memory (PUNLOADP)	7 - 443
7.18.18	Load + Unload (PSWAPP)	7 - 445
7.18.19	High-speed Block Transfer of File Register (RBMOV(P))	7 - 448

## Common Instructions 2/2

<b>8. INSTRUCTIONS FOR DATA LINK</b>	<b>8 - 1 to 8 - 10</b>
8.1 Network refresh instructions	8 - 2
8.1.1 Refresh instruction for the designated module (S(P)/J(P)/G(P).ZCOM)	8 - 2
8.2 Reading/Writing Routing Information	8 - 6
8.2.1 Reading routing information (S(P)/Z(P).RTREAD)	8 - 6
8.2.2 Registering routing information (S(P)/Z(P).RTWRITE)	8 - 8
<b>9. Multiple CPU dedicated instruction</b>	<b>9 - 1 to 9 - 18</b>
9.1 Writing to the CPU Shared Memory of Host CPU	9 - 2
9.1.1 Write to Host CPU Shared Memory (S(P).TO)	9 - 4
9.1.2 Writing to host station CPU shared memory (TO(P), DTO(P))	9 - 7
9.2 Reading from the CPU Shared Memory of another CPU	9 - 11
9.2.1 Reading from Other CPU Shared Memory (FROM(P), DFRO(P))	9 - 12
<b>10. QCPU INSTRUCTIONS</b>	<b>10 - 1 to 10 - 20</b>
10.1 Overview	10 - 2
10.2 Writing Devices to Another CPU (D(P).DDWR)	10 - 13
10.3 Reading Devices from Another CPU (D(P).DDRDR)	10 - 17
<b>11. QCPU INSTRUCTIONS</b>	<b>11 - 1 to 11 - 4</b>
11.1 System Switching Instruction (SP.CONTSW)	11 - 2
<b>12. ERROR CODES</b>	<b>12 - 1 to 12 - 84</b>
12.1 Error Code List	12 - 2
12.1.1 Error codes	12 - 3
12.1.2 Reading an error code	12 - 3
12.1.3 Error code list (1000 to 1999)	12 - 4

12.1.4	Error code list (2000 to 2999)	12 - 16
12.1.5	Error code list (3000 to 3999)	12 - 34
12.1.6	Error code list (4000 to 4999)	12 - 51
12.1.7	Error code list (5000 to 5999)	12 - 66
12.1.8	Error code list (6000 to 6999)	12 - 68
12.1.9	Error code list (7000 to 10000)	12 - 78
12.2	Canceling of Errors	12 - 83

<b>APPENDICES</b>	<b>App - 1 to App - 198</b>
-------------------	-----------------------------

<b>Appendix 1 OPERATION PROCESSING TIME</b>		<b>App - 2</b>
Appendix 1.1	Definition	App - 2
Appendix 1.2	Operation Processing Time of Basic Model QCPU	App - 3
Appendix 1.3	Operation Processing Time of High Performance Model QCPU/Process CPU/ Redundant CPU	App - 21
Appendix 1.4	Operation Processing Time of Universal Model QCPU	App - 50
Appendix 1.4.1	Subset instruction processing time	App - 50
Appendix 1.4.2	Processing time of instructions other than subset instruction	App - 66
<b>Appendix 2 CPU PERFORMANCE COMPARISON</b>		<b>App - 114</b>
Appendix 2.1	Comparison of Q with AnNCP, AnACPU, and AnUCPU	App - 114
Appendix 2.1.1	Usable devices	App - 114
Appendix 2.1.2	I/O control mode	App - 115
Appendix 2.1.3	Data that can be used by instructions	App - 115
Appendix 2.1.4	Timer comparison	App - 116
Appendix 2.1.5	Comparison of counters	App - 117
Appendix 2.1.6	Comparison of display instructions	App - 117
Appendix 2.1.7	Instructions whose designation format has been changed (Except dedicated instructions for AnACPU and AnUCPU)	App - 118
Appendix 2.1.8	AnACPU and AnUCPU dedicated instructions	App - 119
<b>Appendix 3 SPECIAL RELAY LIST</b>		<b>App - 120</b>
<b>Appendix 4 SPECIAL REGISTER LIST</b>		<b>App - 146</b>
<b>Appendix 5 APPLICATION PROGRAM EXAMPLES</b>		<b>App - 198</b>
Appendix 5.1	Concept of Programs which Perform Operations of $X^n$ , $\sqrt[n]{X}$	App - 198

<b>INDEX</b>	<b>Index - 1 to Index - 12</b>
--------------	--------------------------------

# MANUALS

To understand the main specifications, functions, and usage of the CPU module, refer to the basic manuals.  
Read other manuals as well when using a different type of CPU module and its functions.  
Order each manual as needed, referring to the following list.

The numbers in the "CPU module" and the respective modules are as follows.

Number	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU
5)	Universal model QCPU

○:Basic manual, ●:Other CPU module manuals

Manual name < Manual number (model code) >	Description	CPU module				
		1)	2)	3)	4)	5)
<b>■User's manual</b>						
QCPU User's Manual (Hardware design, Maintenance and Inspection) < SH-080483ENG (13JR73) >	Specifications of the hardware (CPU modules, power supply modules, base units, extension cables, and memory cards), system maintenance and inspection, troubleshooting, and error codes	●	●	●	●	●
QnUCPU User's Manual (Function Explanation, Program Fundamentals) < SH-080807ENG (13JZ27) >	Functions, methods, and devices for programming					●
Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals) < SH-080808ENG (13JZ28) >	Functions, methods, and devices for programming	●	●	●	●	
QnUCPU User's Manual (Communication via Built-in Ethernet Port) < SH-080811ENG (13JZ29) >	Functions for the communication via built-in Ethernet port of the CPU module					○
<b>■Programming Manual</b>						
QCPU Programming Manual (Common Instructions) < SH-080809ENG (13JW10) >	How to use sequence instructions, basic instructions, and application instructions	●	●	●	●	●
QCPU (Q Mode)/QnACPU Programming Manual (SFC) < SH-080041 (13JF60) >	System configuration, performance specifications, functions, programming, debugging, and error codes for SFC (MELSAP3) programs	○	○	○	○	○
QCPU (Q Mode) Programming Manual (MELSAP-L) < SH-080072 (13JC03) >	Programming methods, specifications, and functions for SFC (MELSAP-L) programs	○	○	○	○	○
QCPU (Q Mode) Programming Manual (Structured Text) < SH-080366E (13JF68) >	Programming methods using structured languages	○	○	○	○	○
QCPU (Q Mode) / QnACPU Programming Manual (PID Control Instructions) < SH-080040 (13JF59) >	Dedicated instructions for PID control	○	○		○	○
QnPH/QnPRHCPU Programming Manual (Process Control Instructions) < SH-080316E (13JF59) >	Describes the dedicated instructions for performing process control.			○	○	



Related Manuals

Manual name < Manual number (model code) >	Description
CC-Link IE Controller Network Reference Manual < SH-080668ENG (13JV16) >	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of the CC-Link IE controller network module
Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network) < SH-080049 (13JF92) >	Explains the specifications for a MELSECNET/H network system for PLC to PLC network. It explains the procedures and settings up to operation, setting the parameters, programming and troubleshooting.
Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network) < SH-080124 (13JF96) >	Explains the specifications for a MELSECNET/H network system for remote I/O network. It explains the procedures and settings up to operation, setting the parameters, programming and troubleshooting.
Type MELSECNET, MELSECNET/B Data Link System Reference Manual < IB-66530 (13JF70) >	Describes the general concept, specifications, and part names and settings for MELSECNET (II) and MELSECNET/B.
Q Corresponding Ethernet Interface Module User's Manual (Application) < SH-080010 (13JF70) >	Describes various functions of the Ethernet module: e-mail function, PLC CPU status monitoring, communication via MELSECNET/H or MELSECNET/10 network system, communication using data link instructions, file transfer (using FTP) and other functions.



# 1

## GENERAL DESCRIPTION

---

This manual explains the common instructions required for programming of the QCPU.

The common instructions refer to all instructions except those dedicated to special function modules (such as AJ71QC24 and AJ71PT32-S3) and to AD57 models, as well as PID control instructions, SFC instructions and ST instructions.

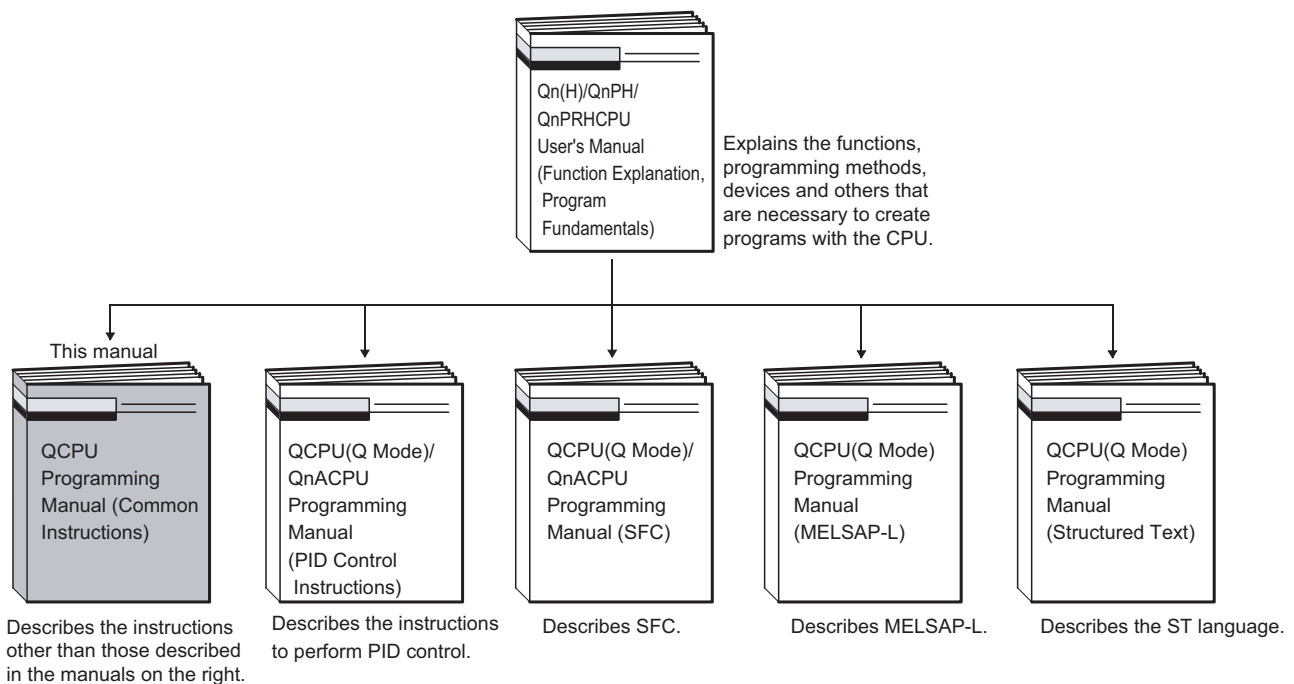
## 1.1 Related Programming Manuals

---

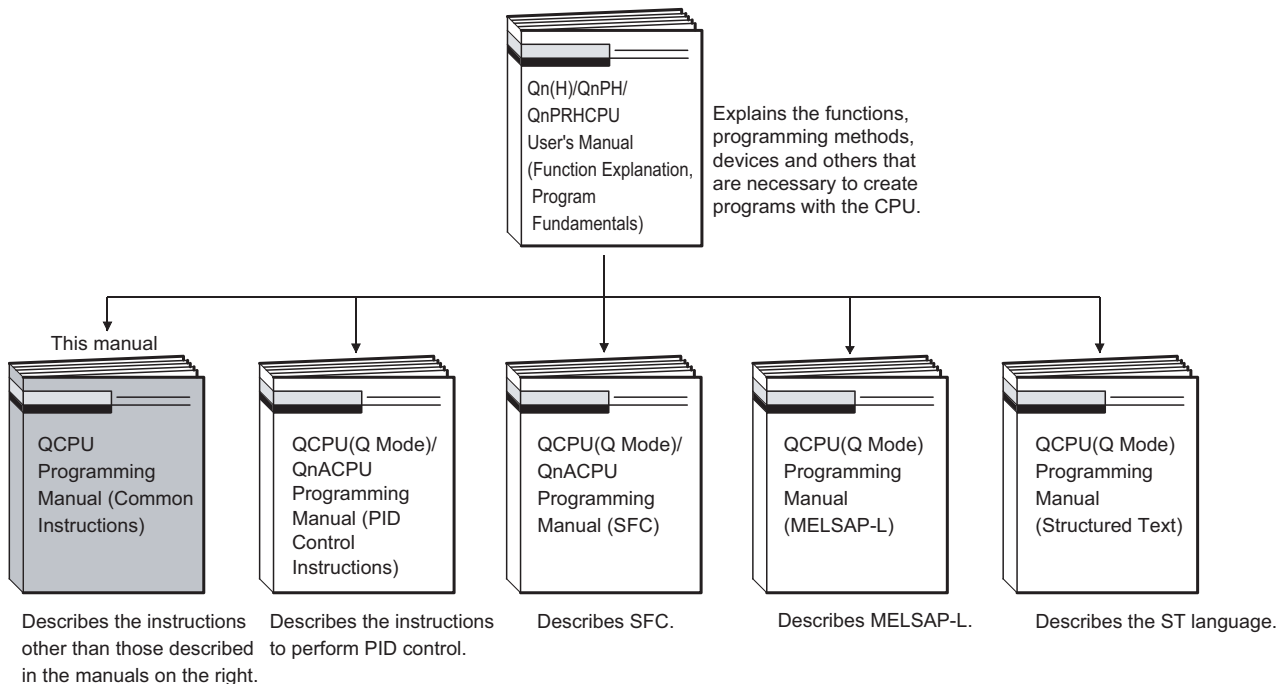
Before reading this manual, check the functions, programming methods, devices and others that are necessary to create programs with the CPU in the manuals below:

- QnUCPU User's Manual (Function Explanation, Program Fundamentals)
- Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)

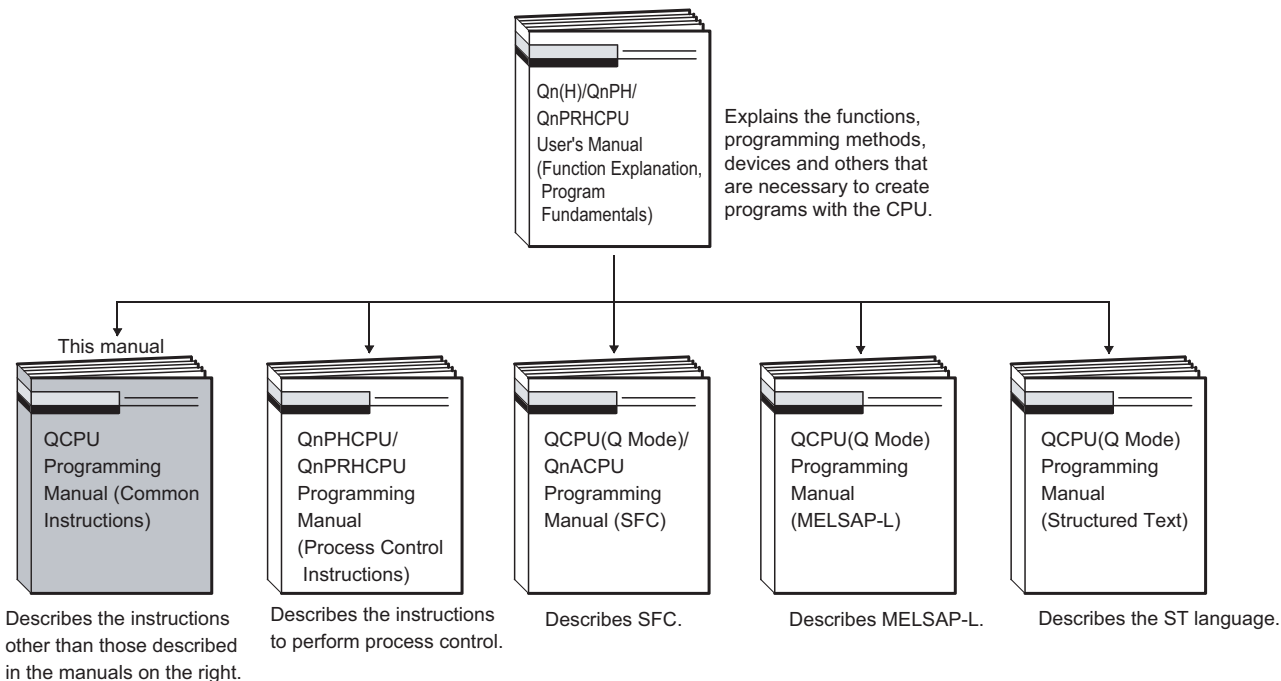
(1) High Performance model QCPU



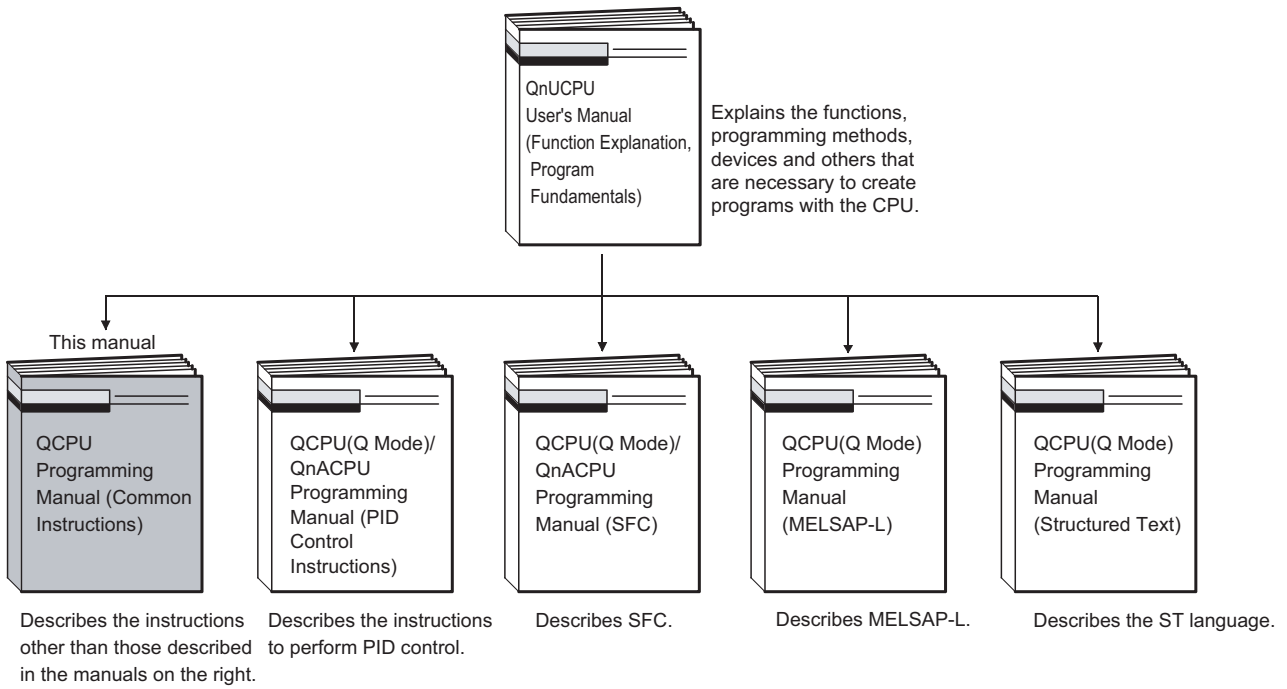
(2) Basic model QCPU



(3) Process CPU and Redundant CPU



#### (4) Universal model QCPU



## 1.2 Abbreviations and Generic Names

This manual uses the generic names and abbreviations shown below to refer to Q series CPU modules, unless otherwise specified.

\* □ indicates a part of the model or version.

Generic term/Abbreviation	Description of Generic Name/Abbreviation
<b>■ Series</b>	
Q series	Abbreviation for Mitsubishi MELSEC-Q series programmable controller
<b>■ CPU module type</b>	
CPU module	Generic term for Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU and Universal model QCPU
Basic model QCPU	Generic term for Q00JCPU, Q00CPU and Q01CPU
High Performance model QCPU	Generic term for Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU and Q25HCPU
Process CPU	Generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU and Q25PHCPU
Redundant CPU	Generic term for Q12PRHCPU and Q25PRHCPU
Universal model QCPU	Generic term for Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU and Q26UDEHCPU
<b>■ CPU module model</b>	
QnCPU	Generic term for Q00JCPU, Q00CPU, Q01CPU and Q02CPU
QnHCPU	Generic term for Q02HCPU, Q06HCPU, Q12HCPU and Q25HCPU
QnPHCPU	Generic term for Q02PHCPU, Q06PHCPU, Q12PHCPU and Q25PHCPU
QnPRHCPU	Generic term for Q12PRHCPU and Q25PRHCPU
QnUCPU	Generic term for Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU and Q26UDEHCPU
QnU(D)(H)CPU	Generic term for Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU and Q26UDHCPU
QnUD(H)CPU	Generic name for Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU and Q26UDHCPU
QnUDE(H)CPU	Generic name for Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU and Q26UDEHCPU

(Continued)

Generic Name/Abbreviation	Description of Generic Name/Abbreviation
<b>■ Base unit model</b>	
Q3 □ B	Generic term for Q33B, Q35B, Q38B and Q312B main base units on which CPU module (except Q00JCPU), Q series power supply module, Q series I/O module, and intelligent function module can be mounted.
Q3 □ SB	Generic term for Q32SB, Q33SB and Q35SB slim type main base units on which Basic model QCPU (except Q00JCPU), High Performance model QCPU, slim type power supply module, Q series I/O module, and intelligent function module can be mounted.
Q3 □ RB	Other name for Q38RB redundant power supply main base unit on which CPU module (except Q00JCPU), redundant power supply module, Q series I/O module, and intelligent function module can be mounted.
Q3 □ DB	Generic term for the Q32DB and Q312DB type Multiple CPU high speed main base unit on which CPU module (except the Q00JCPU), Q series power supply module, Q series I/O module, and intelligent function module can be mounted.
Q5 □ B	Generic term for Q52B and Q55B extension base unit on which the Q Series I/O and intelligent function module can be mounted.
Q6 □ B	Generic term for Q63B, Q65B, Q68B and Q612B extension base unit on which Q Series power supply module, I/O module, intelligent function module can be mounted.
Q6 □ RB	Other name for Q68RB redundant power supply extension base unit on which redundant power supply module, Q series I/O module, and intelligent function module can be mounted.
Q6 □ WRB	Other name for Q65WRB extension base unit for redundant system on which redundant power supply module, Q series I/O module, and intelligent function module can be mounted.
QA1S6 □ B	Generic term for QA1S65B and QA1S68B extension base units on which AnS Series power supply module, I/O module, special function module can be mounted.
QA6 □ B	Generic term for QA65B and QA68B extension base units on which the A series power supply module, A series I/O modules and special function modules can be mounted.
A5 □ B	Generic term for A52B, A55B, and A58B extension base units on which A series I/O module and special function module can be mounted without power supply.
A6 □ B	Generic term for A62B, A65B, and A68B extension base units on which A series I/O module and special function module can be mounted.
QA6ADP	Abbreviation for QA6ADP QA conversion adapter module.
QA6ADP+A5 □ B/A6 □ B	Abbreviation for A large type extension base unit on which QA6ADP is mounted.
<b>■ Network</b>	
MELSECNET/H	Abbreviation for MELSECNET/H network system
MELSECNET/10	Abbreviation for MELSECNET/10 network system
MELSECNET(II,B)	Abbreviation for MELSECNET and MELSECNET/B data link system
Ethernet	Abbreviation for Ethernet network system
CC-Link	Abbreviation for Control & Communication Link



(Continued)

Generic Name/Abbreviation	Description of Generic Name/Abbreviation
■ Others	
GX Developer	Product name of Q series Corresponding SW <input type="checkbox"/> D5C-GPPW-type GPP function software package <input type="checkbox"/> : Version of the software Check the GX Developer versions that can be used for each CPU module in "System Configuration," QCPU User's Manual (Hardware Design, Maintenance and Inspection).
Intelligent function module	Generic name for intelligent function modules and special function modules
Intelligent function module device	Generic name for intelligent function module devices and special function module devices



# 2

## INSTRUCTION TABLES

---

## 2.1 Types of Instructions

The major types of CPU module instructions consist of sequence instructions, basic instructions, application instructions, data link instructions, QCPU instructions and redundant system instructions. These types of instructions are listed in Table 2.1 below.

Table 2.1 Types of Instructions

Types of Instruction	Meaning	Reference Chapter	
Sequence instruction	Contact instruction	Operation start, series connection, parallel connection	5
	Association instruction	Ladder block connection, store/read operation results, creation of pulses from operation results	
	Output instruction	Bit device output, pulse output, output reversal	
	Shift instruction	Bit device shift	
	Master control instruction	Master control	
	Termination instruction	Program termination	
	Other instruction	Program stop, instructions such as no operation which do not fit in the above categories	
Basic instruction	Comparison operation instruction	Comparisons such as =, >, <	6
	Arithmetic operation instruction	Addition, subtraction, multiplication or division of BIN or BCD	
	BCD ↔ BIN conversion instruction	Conversion from BCD to BIN and from BIN to BCD	
	Data transfer instruction	Transmits designated data	
	Program branch instruction	Program jumps	
	Program run control instruction	Enables or inhibits interrupt programs	
	I/O refresh	Executes partial refresh	
Application instruction	Other convenient instruction	Instructions for: Counter increment/decrement, teaching timer, special function timer, rotary table shortest direction control, etc.	7
	Logical operation instruction	Logical operations such as logical sum, logical product, etc.	
	Rotation instruction	Rotation of designated data	
	Shift instruction	Shift of designated data	
	Bit processing instruction	Bit set and reset, bit test, batch reset of bit devices	
	Data processing instruction	16-bit data searches, data processing such as decoding and encoding	
	Structure creation instruction	Repeated operation, subroutine program calls, indexing in ladder units	
	Table operation instruction	Data table read/write	
	Buffer memory access instruction	Data read/write from/to an intelligent function module	
	Display instruction	Print ASCII code, LED character display, etc.	
	Debugging and failure diagnosis instruction	Check, status latch, sampling trace, program trace	
	Character string processing instruction	Conversion between BIN/BCD and ASCII; conversion between BIN and character string; conversion between floating decimal point data and character strings, character string processing, etc.	
	Special function instruction	Trigonometric functions, conversion between angles and radians, exponential operations, automatic logarithms, square roots	
	Data control instruction	Upper and lower limit controls, dead band controls, zone controls	
	Switching instruction	File register block No. switches, designation of file registers and comment files	
	Clock instruction	Reading/writing of the values of year, month, day, hour, minute, second, and day of the week; addition/subtraction of the values of hour, minute, and second; conversion of the values of hour, minute, and second into second; comparison between the values of year, month, and day; and comparison between the values of hour, minute, and second	
	Expansion clock instruction	Reading of the values of year, month, day, hour, minute, second, millisecond, and day of the week; addition/subtraction of the values of hour, minute, second, and millisecond	
Peripheral device instruction	I/O to peripheral devices		
Program control instruction	Instructions to switch program execution conditions		
Other instruction	Instructions that do not fit in the above categories, such as watchdog timer reset instructions and timing clock instructions		

Table 2.1 Types of Instructions (Continued)

Types of Instruction		Meaning	Reference Chapter
Instruction for Data Link	Link refresh instruction	Designated network refresh	8
	Routing information read/write instruction	Reads, writes, and registers routing information	
Multiple CPU dedicated instruction	Multiple CPU dedicated instruction	Writing to host CPU shared memory, Reading from other CPU shared memory	9
Multiple CPU high-speed transmission dedicated instruction	Multiple CPU device write/read instruction	Writes/reads devices to/from another CPU.	10
Redundant system instruction	Instruction for Redundant CPU	System switching	11

# 2.2 How to Read Instruction Tables

The instruction tables found from Section 2.3 to 2.5 have been made according to the following format:

Table 2.2 How to Read Instruction Tables

Category	Instruction Symbols	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN 16-bit addition and subtraction operations	+		· (D)+(S) →(D)		3	●	6-16
	+P						
	+		· (S1)+(S2) →(D)		4	●	6-20
	+P						

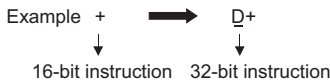


### Description

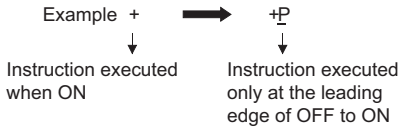
- 1) .....Classifies instructions according to their application.
- 2) .....Indicates the instruction symbol added to the instruction in a program.

Instruction code is built around the 16-bit instruction. The following notations are used to mark 32-bit instructions, instructions executed only at the leading edge of OFF to ON, real number instructions, and character string instructions:

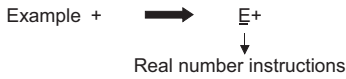
- 32-bit instruction..... The letter "D" is added to the first line of the instruction.



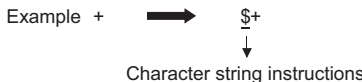
- Instructions executed only at the leading edge of OFF to ON  
..... The letter "P" is added to the end of the instruction.



- Real number instructions  
..... The letter "E" is added to the first line of the instruction.



- Character string instructions  
..... A dollar sign \$ is added to the first line of the instruction.



3) .....Shows symbol diagram on the ladder.

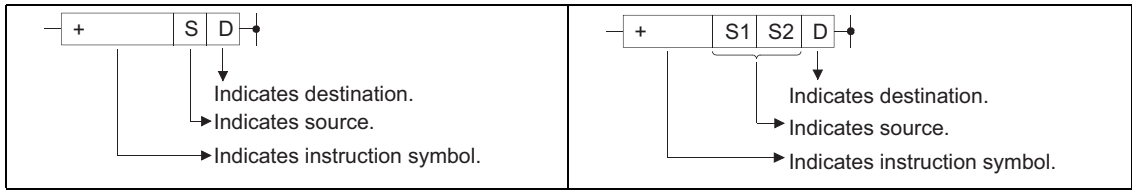


Fig. 2.1 Symbol Diagram on the Ladder

Destination..... Indicates where data will be sent after operation.

Source ..... Stores data prior to operation.

4) .....Indicates the type of processing that is performed by individual instructions.

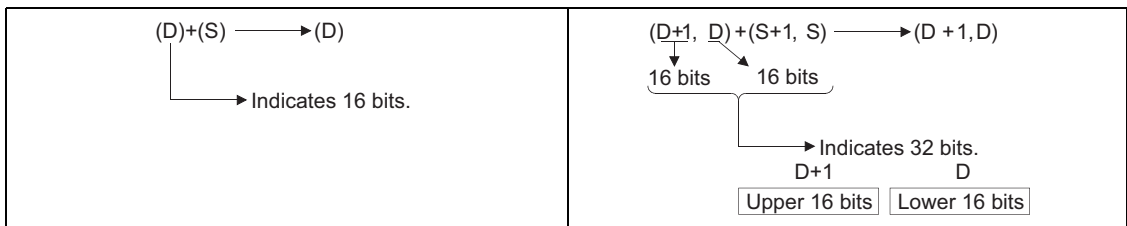


Fig. 2.2 Type of Processing Performed by Individual Instructions

5) .....The details of conditions for the execution of individual instructions are as follows:

Symbol	Execution Condition
No symbol recorded	Instruction executed under normal circumstances, with no regard to the ON/OFF status of conditions prior to the instruction. If the precondition is OFF, the instruction will conduct OFF processing.
	Executed during ON; instruction is executed only while the precondition is ON. If the preconditions is OFF, the instruction is not executed, and no processing is conducted.
	Executed once at ON; instruction executed only at leading edge when precondition goes from OFF to ON. Following execution, instruction will not be executed and no processing conducted even if condition remains ON.
	Executed during OFF; instruction is executed only while the precondition is OFF. If the precondition is ON, the instruction is not executed, and no processing is conducted.
	Executed once at OFF; instruction executed only at trailing edge when precondition goes from ON to OFF. Following execution, instruction will not be executed and no processing conducted even if condition remains OFF.

6) .....Indicates the basic number of steps for individual instructions.

See Section 3.8 for a description of the number of steps.

7) .....The ● mark indicates instructions for which subset processing is possible.

See Section 3.5 for details on subset processing.

8) .....Indicates the page numbers where the individual instructions are explained.

## 2.3 Sequence Instructions

### 2.3.1 Contact instructions

Table 2.3 Contact Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Contact	LD		• Starts logic operation (Starts a contact logic operation)		*1	●	5-2
	LDI		• Starts logical NOT operation (Starts b contact logic operation)				
	AND		• Logical product (a contact series connection)				
	ANI		• Logical product NOT (b contact series connection)				
	OR		• Logical sum (a contact parallel connection)				
	ORI		• Logical sum NOT (b contact parallel connection)				
	LDP		• Starts leading edge pulse operation		*2	●	5-5
	LDF		• Starts trailing edge pulse operation				
	ANDP		• Leading edge pulse series connection				
	ANDF		• Trailing edge pulse series connection				
	ORP		• Leading edge pulse parallel connection				
	ORF		• Trailing edge pulse parallel connection				
	LDPI		• Starts leading edge pulse NOT operation		3	●	5-7
	LDFI		• Starts trailing edge pulse NOT operation		3		
	ANDPI		• Leading edge pulse NOT series connection		4		
	ANDFI		• Trailing edge pulse NOT series connection		4		
	ORPI		• Leading edge pulse NOT parallel connection		4		
	ORFI		• Trailing edge pulse NOT parallel connection		4		

\*1: The number of steps may vary depending on the device being used.

Device	Number of Steps
Internal device, file register (R0 to R32767)	1
Direct access input (DX)	2
Devices other than above	3



\*2: The number of steps may vary depending on the device and type of CPU module being used.

Device	Number of Steps
	QCPU
Internal device, file register (R0 to R32767)	1
Direct access input (DX)	2
Devices other than above	3

## 2.3.2 Association instructions

Table 2.4 Association Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Connection	ANB		<ul style="list-style-type: none"> <li>AND between logical blocks (Series connection between logical blocks)</li> </ul>		1	-	5-10
	ORB		<ul style="list-style-type: none"> <li>OR between logical blocks (Series connection between logical blocks)</li> </ul>		1	-	5-10
	MPS		<ul style="list-style-type: none"> <li>Memory storage of operation results</li> </ul>		1	-	5-12
	MRD		<ul style="list-style-type: none"> <li>Read of operation results stored with MPS instruction</li> </ul>				
	MPP		<ul style="list-style-type: none"> <li>Read and reset of operation results stored with MPS instruction</li> </ul>				
	INV		<ul style="list-style-type: none"> <li>Inversion of operation result</li> </ul>		1	-	5-15
	MEP		<ul style="list-style-type: none"> <li>Conversion of operation result to leading edge pulse</li> </ul>		1	-	5-17
	MEF		<ul style="list-style-type: none"> <li>Conversion of operation result to trailing edge pulse</li> </ul>				
	EGP		<ul style="list-style-type: none"> <li>Conversion of operation result to leading edge pulse (Stored at Vn)</li> </ul>		1	-	5-18
	EGF		<ul style="list-style-type: none"> <li>Conversion of operation result to trailing edge pulse (Stored at Vn)</li> </ul>		*1		

\*1: The number of steps may vary depending on the device and type of CPU module being used.

Component	Number of Basic Steps
High Performance model QCPU	1
Process CPU	
Redundant CPU	
Universal model QCPU	2
Basic model QCPU	

## 2.3.3 Output instructions

Table 2.5 Output Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Output	OUT		• Device output		*1	-	5-20 5-22 5-26 5-28
	SET		• Sets device		*1	-	5-30 5-35
	RST		• Resets device		*1	-	5-32 5-35
	PLS		• Generates 1 cycle program pulse at leading edge of input signal.		2	-	5-37
	PLF		• Generates 1 cycle program pulse at trailing edge of input signal.				
	FF		• Reversal of device output		2	-	5-40
	DELTA		• Pulse conversion of direct output		2	-	5-42
	DELTAP						

\*1: The number of steps may vary depending on the device being used. See description pages of individual instructions for number of steps.

\*2: The execution condition applies only when an annunciator (F) is in use.

## 2.3.4 Shift instructions

Table 2.6 Shift Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Shift	SFT		• 1-bit shift of device		2	-	5-44
	SFTP						

## 2.3.5 Master control instructions

Table 2.7 Master Control Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Master control	MC		• Starts master control		2	-	5-47
	MCR		• Resets master control		1		

## 2.3.6 Termination instructions

Table 2.8 Termination Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Termination	FEND		• Termination of main program		1	-	5-51
	END		• Termination of sequence program				5-53

## 2.3.7 Other instructions

Table 2.9 Other Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Stop	STOP		<ul style="list-style-type: none"> <li>• Terminates sequence operation after input condition has been met.</li> <li>• Sequence program is executed by placing the RUN/STOP key switch back in the RUN position.</li> </ul>		1	-	5-55
Ignored	NOP		• Ignored (For program deletion or space)		1	-	5-57
	NOPLF		• Ignored (To change pages during printouts)				
	PAGE		• Ignored (Subsequent programs will be controlled from step 0 of page n)				

## 2.4 Basic instructions

### 2.4.1 Comparison operation instructions

Table 2.10 Comparison Operation Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN 16-bit data comparisons	LD=		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1) = (S2)</math></li> <li>• Non-conductive status when <math>(S1) \neq (S2)</math></li> </ul>		3	●	6-2
	AND=						
	OR=						
	LD<>		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1) \neq (S2)</math></li> <li>• Non-conductive status when <math>(S1) = (S2)</math></li> </ul>		3	●	
	AND<>						
	OR<>						
	LD>		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1) &gt; (S2)</math></li> <li>• Non-conductive status when <math>(S1) \leq (S2)</math></li> </ul>		3	●	
	AND>						
	OR>						
	LD<=		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1) \leq (S2)</math></li> <li>• Non-conductive status when <math>(S1) &gt; (S2)</math></li> </ul>		3	●	
	AND<=						
	OR<=						
	LD<		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1) &lt; (S2)</math></li> <li>• Non-conductive status when <math>(S1) \geq (S2)</math></li> </ul>		3	●	
	AND<						
	OR<						
LD>=		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1) \geq (S2)</math></li> <li>• Non-conductive status when <math>(S1) &lt; (S2)</math></li> </ul>		3	●		
AND>=							
OR>=							

Table 2.10 Comparison Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN 32-bit data comparisons	LDD=		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+1, S1) = (S2+1, S2)</math></li> <li>Non-Conductive status when <math>(S1+1, S1) \neq (S2+1, S2)</math></li> </ul>		*1	●	6-4
	ANDD=						
	ORD=						
	LDD<>		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+1, S1) \neq (S2+1, S2)</math></li> <li>Non-Conductive status when <math>(S1+1, S1) = (S2+1, S2)</math></li> </ul>		*1	●	
	ANDD<>						
	ORD<>						
	LDD>		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+1, S1) &gt; (S2+1, S2)</math></li> <li>Non-Conductive status when <math>(S1+1, S1) \leq (S2+1, S2)</math></li> </ul>		*1	●	
	ANDD>						
	ORD>						
	LDD<=		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+1, S1) \leq (S2+1, S2)</math></li> <li>Non-Conductive status when <math>(S1+1, S1) &gt; (S2+1, S2)</math></li> </ul>		*1	●	
	ANDD<=						
	ORD<=						
	LDD<		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+1, S1) &lt; (S2+1, S2)</math></li> <li>Non-Conductive status when <math>(S1+1, S1) \geq (S2+1, S2)</math></li> </ul>		*1	●	
	ANDD<						
	ORD<						
	LDD>=		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+1, S1) \geq (S2+1, S2)</math></li> <li>Non-Conductive status when <math>(S1+1, S1) &lt; (S2+1, S2)</math></li> </ul>		*1	●	
ANDD>=							
ORD>=							

\*1: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>Word device: Internal device (except for file register ZR)</li> <li>Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no Indexing.</li> <li>Constant: No limitations</li> </ul>	5 Note 1)
	Devices other than above	3 Note 2)
Basic model QCPU Universal model QCPU	All devices that can be used	3 Note 2)

Note 1) When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.

Note 2) The number of steps may increase due to the conditions described in Section 3.8.

Table 2.10 Comparison Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Floating decimal point data comparisons (Single precision)	LDE=		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1+1, S1) = (S2+1, S2)</math></li> <li>• Non-Conductive status when <math>(S1+1, S1) \neq (S2+1, S2)</math></li> </ul>		3	-	6-6
	ANDE=						
	ORE=						
	LDE<>		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1+1, S1) \neq (S2+1, S2)</math></li> <li>• Non-Conductive status when <math>(S1+1, S1) = (S2+1, S2)</math></li> </ul>		3		
	ANDE<>						
	ORE<>						
	LDE>		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1+1, S1) &gt; (S2+1, S2)</math></li> <li>• Non-Conductive status when <math>(S1+1, S1) \leq (S2+1, S2)</math></li> </ul>		3		
	ANDE>						
	ORE>						
	LDE<=		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1+1, S1) \leq (S2+1, S2)</math></li> <li>• Non-Conductive status when <math>(S1+1, S1) &gt; (S2+1, S2)</math></li> </ul>		3		
	ANDE<=						
	ORE<=						
	LDE<		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1+1, S1) &lt; (S2+1, S2)</math></li> <li>• Non-Conductive status when <math>(S1+1, S1) \geq (S2+1, S2)</math></li> </ul>		3		
	ANDE<						
	ORE<						
LDE>=		<ul style="list-style-type: none"> <li>• Conductive status when <math>(S1+1, S1) \geq (S2+1, S2)</math></li> <li>• Non-Conductive status when <math>(S1+1, S1) &lt; (S2+1, S2)</math></li> </ul>	3				
ANDE>=							
ORE>=							

Table 2.10 Comparison Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Floating decimal point data comparisons (Double precision)	LDED=		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+3, S1+2, S1+1, S1) = (S2+3, S2+2, S2+1, S2)</math></li> </ul>		3	-	6-8
	ANDED=		<ul style="list-style-type: none"> <li>Non-Conductive status when <math>(S1+3, S1+2, S1+1, S1) \neq (S2+3, S2+2, S2+1, S2)</math></li> </ul>				
	ORED=						
	LDED<>		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+3, S1+2, S1+1, S1) \neq (S2+3, S2+2, S2+1, S2)</math></li> </ul>		3	-	
	ANDED<>		<ul style="list-style-type: none"> <li>Non-Conductive status when <math>(S1+3, S1+2, S1+1, S1) = (S2+3, S2+2, S2+1, S2)</math></li> </ul>				
	ORED<>						
	LDED>		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+3, S1+2, S1+1, S1) &gt; (S2+3, S2+2, S2+1, S2)</math></li> </ul>		3	-	
	ANDED>		<ul style="list-style-type: none"> <li>Non-Conductive status when <math>(S1+3, S1+2, S1+1, S1) \leq (S2+3, S2+2, S2+1, S2)</math></li> </ul>				
	ORED>						
	LDED<=		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+3, S1+2, S1+1, S1) \leq (S2+3, S2+2, S2+1, S2)</math></li> </ul>		3	-	
	ANDED<=		<ul style="list-style-type: none"> <li>Non-Conductive status when <math>(S1+3, S1+2, S1+1, S1) &gt; (S2+3, S2+2, S2+1, S2)</math></li> </ul>				
	ORED<=						
	LDED<		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+3, S1+2, S1+1, S1) &lt; (S2+3, S2+2, S2+1, S2)</math></li> </ul>		3	-	
	ANDED<		<ul style="list-style-type: none"> <li>Non-Conductive status when <math>(S1+3, S1+2, S1+1, S1) \geq (S2+3, S2+2, S2+1, S2)</math></li> </ul>				
	ORED<						
LDED>=		<ul style="list-style-type: none"> <li>Conductive status when <math>(S1+3, S1+2, S1+1, S1) \geq (S2+3, S2+2, S2+1, S2)</math></li> </ul>		3	-		
ANDED>=		<ul style="list-style-type: none"> <li>Non-Conductive status when <math>(S1+3, S1+2, S1+1, S1) &lt; (S2+3, S2+2, S2+1, S2)</math></li> </ul>					
ORED>=							

Table 2.10 Comparison Operation Instructions (Continued)





Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Character string data comparisons	LD\$=		<ul style="list-style-type: none"> <li>• Compares character string S1 and character string S2 one character at a time. *2</li> <li>• Conductive status when (character string S1) = (character string S2)</li> <li>• Non-Conductive status when (character string S1) ≠ (character string S2)</li> </ul>		3	-	6-8
	AND\$=						
	OR\$=						
	LD\$<>		<ul style="list-style-type: none"> <li>• Compares character string S1 and character string S2 one character at a time. *2</li> <li>• Conductive status when (character string S1) ≠ (character string S2)</li> <li>• Non-Conductive status when (character string S1) = (character string S2)</li> </ul>		3		
	AND\$<>						
	OR\$<>						
	LD\$>		<ul style="list-style-type: none"> <li>• Compares character string S1 and character string S2 one character at a time. *2</li> <li>• Conductive status when (character string S1) &gt; (character string S2)</li> <li>• Non-Conductive status when (character string S1) ≤ (character string S2)</li> </ul>		3		
	AND\$>						
	OR\$>						
	LD\$<=		<ul style="list-style-type: none"> <li>• Compares character string S1 and character string S2 one character at a time. *2</li> <li>• Conductive status when (character string S1) ≤ (character string S2)</li> <li>• Non-Conductive status when (character string S1) &gt; (character string S2)</li> </ul>		3		
	AND\$<=						
	OR\$<=						
	LD\$<		<ul style="list-style-type: none"> <li>• Compares character string S1 and character string S2 one character at a time. *2</li> <li>• Conductive status when (character string S1) &lt; (character string S2)</li> <li>• Non-Conductive status when (character string S1) ≥ (character string S2)</li> </ul>		3		
	AND\$<						
	OR\$<						
LD\$>=		<ul style="list-style-type: none"> <li>• Compares character string S1 and character string S2 one character at a time. *2</li> <li>• Conductive status when (character string S1) ≥ (character string S2)</li> <li>• Non-Conductive status when (character string S1) &lt; (character string S2)</li> </ul>		3			
AND\$>=							
OR\$>=							

\*2: The conditions under which character string comparisons can be made are as shown below:

- Match: All characters in the strings must match
- Larger string: If character strings are different, determines the string with the largest number of character codes. If the lengths of the character strings are different, determines the longest character string.
- Smaller string: If the character strings are different, determines the string with the smallest number of character codes. If the lengths of the character strings are different, determines the shortest character string.



Table 2.10 Comparison Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN 16-bit Block data comparisons	BKCMP=	$\text{—BKCMP= S1 S2 D n—}$	<ul style="list-style-type: none"> <li>This instruction compares BIN 16-bit data stored in n-point devices starting from the device specified by S1 with BIN 16-bit data stored in n-point devices starting from the device specified by S2, and then stores the result into the nth device specified by (D) and up.</li> </ul>		5	-	6-15
	BKCMP<>	$\text{—BKCMP<> S1 S2 D n—}$					
	BKCMP>	$\text{—BKCMP> S1 S2 D n—}$					
	BKCMP<=	$\text{—BKCMP<= S1 S2 D n—}$					
	BKCMP<	$\text{—BKCMP< S1 S2 D n—}$					
	BKCMP>=	$\text{—BKCMP>= S1 S2 D n—}$					
	BKCMP=P	$\text{—BKCMP=P S1 S2 D n—}$					
	BKCMP<>P	$\text{—BKCMP<>P S1 S2 D n—}$					
	BKCMP>P	$\text{—BKCMP>P S1 S2 D n—}$					
	BKCMP<=P	$\text{—BKCMP<=P S1 S2 D n—}$					
	BKCMP<P	$\text{—BKCMP<P S1 S2 D n—}$					
	BKCMP>=P	$\text{—BKCMP>=P S1 S2 D n—}$					
BIN 32-bit block data comparisons	DBKCMP=	$\text{—DBKCMP= S1 S2 D n—}$	<ul style="list-style-type: none"> <li>This instruction compares BIN 32-bit data stored in n-point devices starting from the device specified by S1 with BIN 32-bit data stored in n-point devices starting from the device specified by a constant and S2, and then stores the result into the nth device specified by (D) and up.</li> </ul>		5	-	6-18
	DBKCMP<>	$\text{—DBKCMP<> S1 S2 D n—}$					
	DBKCMP>	$\text{—DBKCMP> S1 S2 D n—}$					
	DBKCMP<=	$\text{—DBKCMP<= S1 S2 D n—}$					
	DBKCMP<	$\text{—DBKCMP< S1 S2 D n—}$					
	DBKCMP>=	$\text{—DBKCMP>= S1 S2 D n—}$					
	DBKCMP=P	$\text{—DBKCMP=P S1 S2 D n—}$					
	DBKCMP<>P	$\text{—DBKCMP<>P S1 S2 D n—}$					
	DBKCMP>P	$\text{—DBKCMP>P S1 S2 D n—}$					
	DBKCMP<=P	$\text{—DBKCMP<=P S1 S2 D n—}$					
	DBKCMP<P	$\text{—DBKCMP<P S1 S2 D n—}$					
	DBKCMP>=P	$\text{—DBKCMP>=P S1 S2 D n—}$					

## 2.4.2 Arithmetic operation instructions

Table 2.11 Arithmetic Operation Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN 16-bit addition and subtraction operations	+		• $(D)+(S) \rightarrow (D)$		3	●	6-22
	+P						
	+		• $(S1)+(S2) \rightarrow (D)$		4	●	6-24
	+P						
	-		• $(D)-(S) \rightarrow (D)$		3	●	6-22
	-P						
	-		• $(S1)-(S2) \rightarrow (D)$		4	●	6-24
	-P						
BIN 32-bit addition and subtraction operations	D+		• $(D+1, D)+(S+1, S) \rightarrow (D+1, D)$		*1	●	6-26
	D+P						
	D+		• $(S1+1, S1)+(S2+1, S2) \rightarrow (D+1, D)$		*2	●	6-28
	D+P						
	D-		• $(D+1, D)-(S+1, S) \rightarrow (D+1, D)$		*1	●	6-26
	D-P						
	D-		• $(S1+1, S1)-(S2+1, S2) \rightarrow (D+1, D)$		*2	●	6-28
	D-P						
BIN 16-bit multiplication and division operations	*		• $(S1) \times (S2) \rightarrow (D+1, D)$		*3	●	6-30
	*P						
	/		• $(S1) / (S2)$		*4	●	
	/P		$\rightarrow$ Quotient(D), Remainder (D+1)				
BIN 32-bit multiplication and division operations	D*		• $(S1+1, S1) \times (S2+1, S2) \rightarrow (D+3, D+2, D+1, D)$		*4	●	6-32
	D*P						
	D/		• $(S1+1, S1) / (S2+1, S2)$		*4	●	
	D/P		$\rightarrow$ Quotient (D+1, D), Remainder (D+3, D+2)				

\*1: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	5 Note 1)
	Devices other than above	3 Note 2)
Basic model QCPU Universal model QCPU	All devices that can be used	3 Note 2)

Note 1) When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.

Note 2) The number of steps may increase due to the conditions described in Section 3.8.

\*2: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	6 Note 1)
	Devices other than above	4 Note 2)
Basic model QCPU	All devices that can be used	4 Note 2)
Universal model QCPU		3 Note 2)

Note 1) When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.

Note 2) The number of steps may increase due to the conditions described in Section 3.8.

\*3: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
QCPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	3
	Devices other than above	4 Note 1)

Note 1) The number of steps may increase due to the conditions described in Section 3.8.

\*4: The number of basic steps is three for the Universal model QCPU only.

Table 2.11 Arithmetic Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BCD 4-digit addition and subtraction operations	B+	$\overline{B+} \quad S \quad D$	• $(D)+(S) \rightarrow (D)$		3	●	6-34
	B+P	$\overline{B+P} \quad S \quad D$					
	B+	$\overline{B+} \quad S1 \quad S2 \quad D$	• $(S1)+(S2) \rightarrow (D)$		4	-	6-36
	B+P	$\overline{B+P} \quad S1 \quad S2 \quad D$					
	B-	$\overline{B-} \quad S \quad D$	• $(D)-(S) \rightarrow (D)$		3	●	6-34
	B-P	$\overline{B-P} \quad S \quad D$					
	B-	$\overline{B-} \quad S1 \quad S2 \quad D$	• $(S1)-(S2) \rightarrow (D)$		4	-	6-36
	B-P	$\overline{B-P} \quad S1 \quad S2 \quad D$					
BCD 8-digit addition and subtraction operations	DB+	$\overline{DB+} \quad S \quad D$	• $(D+1, D)+(S+1, S) \rightarrow (D+1, D)$		3	-	6-38
	DB+P	$\overline{DB+P} \quad S \quad D$					
	DB+	$\overline{DB+} \quad S1 \quad S2 \quad D$	• $(S1+1, S1)+(S2+1, S2) \rightarrow (D+1, D)$		4	-	6-40
	DB+P	$\overline{DB+P} \quad S1 \quad S2 \quad D$					
	DB-	$\overline{DB-} \quad S \quad D$	• $(D+1, D)-(S+1, S) \rightarrow (D+1, D)$		3	-	6-38
	DB-P	$\overline{DB-P} \quad S \quad D$					
	DB-	$\overline{DB-} \quad S1 \quad S2 \quad D$	• $(S1+1, S1)-(S2+1, S2) \rightarrow (D+1, D)$		4	-	6-40
	DB-P	$\overline{DB-P} \quad S1 \quad S2 \quad D$					
BCD 4-digit multiplication and division operations	B*	$\overline{B*} \quad S1 \quad S2 \quad D$	• $(S1) \times (S2) \rightarrow (D+1, D)$		4	●	6-42
	B*P	$\overline{B*P} \quad S1 \quad S2 \quad D$					
	B/	$\overline{B/} \quad S1 \quad S2 \quad D$	• $(S1) / (S2)$ → Quotient(D), Remainder (D+1)		4	●	
	B/P	$\overline{B/P} \quad S1 \quad S2 \quad D$					
BCD 8-digit multiplication and division operations	DB*	$\overline{DB*} \quad S1 \quad S2 \quad D$	• $(S1+1, S1) \times (S2+1, S2)$ → (D+3, D+2, D+1, D)		4	-	6-44
	DB*P	$\overline{DB*P} \quad S1 \quad S2 \quad D$					
	DB/	$\overline{DB/} \quad S1 \quad S2 \quad D$	• $(S1+1, S1) / (S2+1, S2)$ → Quotient (D+1, D), Remainder (D+3, D+2)		4	●	
	DB/P	$\overline{DB/P} \quad S1 \quad S2 \quad D$					

Table 2.11 Arithmetic Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Floating decimal point data addition and subtraction operations (Single precision)	E+	$\overline{E+} \quad S \quad D$	• (D+1, D)+(S+1, S)→(D+1, D)		3	● *6	6-46
	E+P	$\overline{E+P} \quad S \quad D$					
	E+	$\overline{E+} \quad S1 \quad S2 \quad D$	• (S1+1, S1)+(S2+1, S2)→(D+1, D)		4	● *6	6-48
	E+P	$\overline{E+P} \quad S1 \quad S2 \quad D$					
	E-	$\overline{E-} \quad S \quad D$	• (D+1, D)−(S+1, S)→(D+1, D)		3	● *6	6-46
	E-P	$\overline{E-P} \quad S \quad D$					
	E-	$\overline{E-} \quad S1 \quad S2 \quad D$	• (S1+1, S1)−(S2+1, S2)→(D+1, D)		4	● *6	6-48
	E-P	$\overline{E-P} \quad S1 \quad S2 \quad D$					
Floating decimal point data addition and subtraction operations (Double precision)	ED+	$\overline{ED+} \quad S \quad D$	• (D+3, D+2, D+1, D)+(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	●	6-50
	ED+P	$\overline{ED+P} \quad S \quad D$					
	ED+	$\overline{ED+} \quad S1 \quad S2 \quad D$	• (S1+3, S1+2, S1+1, S1)+(S2+3, S2+2, S2+1, S2)→(D+3, D+2, D+1, D)		4	●	6-52
	ED+P	$\overline{ED+P} \quad S1 \quad S2 \quad D$					
	ED-	$\overline{ED-} \quad S \quad D$	• (D+3, D+2, D+1, D)−(S+3, S+2, S+1, S)→(D+3, D+2, D+1, D)		3	●	6-50
	ED-P	$\overline{ED-P} \quad S \quad D$					
	ED-	$\overline{ED-} \quad S1 \quad S2 \quad D$	• (S1+3, S1+2, S1+1, S1)−(S2+3, S2+2, S2+1, S2)→(D+3, D+2, D+1, D)		4	●	6-52
	ED-P	$\overline{ED-P} \quad S1 \quad S2 \quad D$					
Floating decimal point data multiplication and division operations (Single precision)	E*	$\overline{E*} \quad S1 \quad S2 \quad D$	• (S1+1, S1) × (S2+1, S2)→(D+1, D)		3	● *6	6-54
	E*P	$\overline{E*P} \quad S1 \quad S2 \quad D$					
	E/	$\overline{E/} \quad S1 \quad S2 \quad D$	• (S1+1, S1) / (S2+1, S2)→Quotient (D+1, D)		4	● *6	
	E/P	$\overline{E/P} \quad S1 \quad S2 \quad D$					
Floating decimal point data multiplication and division operations (Double precision)	ED*	$\overline{ED*} \quad S1 \quad S2 \quad D$	• (S1+3, S1+2, S1+1, S1) × (S2+3, S2+2, S2+1, S2)→(D+3, D+2, D+1, D)		4	● *6	6-56
	ED*P	$\overline{ED*P} \quad S1 \quad S2 \quad D$					
	ED/	$\overline{ED/} \quad S1 \quad S2 \quad D$	• (S1+3, S1+2, S1+1, S1) / (S2+3, S2+2, S2+1, S2)→Quotient (D+3, D+2, D+1, D)		4	● *6	
	ED/P	$\overline{ED/P} \quad S1 \quad S2 \quad D$					

\*5: The number of basic steps is three for the Universal model QCPU only.

\*6: The subset is effective only with Universal model QCPU.

Table 2.11 Arithmetic Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN 16-bit data block addition and subtraction operations	BK+	$\boxed{\text{BK+}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$	• This instruction adds BIN 16-bit data stored in n-point devices starting from the device specified by (S1) to the n-point data stored in the devices starting from the device specified by (S2) in batch.		5	-	6-59
	BK+P	$\boxed{\text{BK+P}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$					
	BK-	$\boxed{\text{BK-}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$	• This instruction subtracts BIN 16-bit data stored in the n-point devices starting from the devices specified by (S2) from BIN 16-bit data stored in n-point devices starting from the device specified by (S1) in batch.		5	-	
	BK-P	$\boxed{\text{BK-P}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$					
BIN 32-bit data block addition and subtraction operations	DBK+	$\boxed{\text{DBK+}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$	• Adds BIN 32-bit data stored in the n-point devices starting from the device specified by (S1) and a constant to BIN 32-bit data stored in the n-point devices starting from the device specified by (S2) and stores the result into the nth device specified by (D) and up.		5	-	6-62
	DBK+P	$\boxed{\text{DBK+P}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$					
	DBK-	$\boxed{\text{DBK-}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$	• Subtracts BIN 32-bit data stored in the n-point devices starting from the device specified by (S2) or a constant from BIN 32-bit data stored in n-point devices starting from the device specified by (S1) and stores the operation result into the nth device specified by (D) and up.		5	-	
	DBK-P	$\boxed{\text{DBK-P}} \quad \boxed{S1} \boxed{S2} \boxed{D} \boxed{n}$					
Character string data Connection	\$+	$\boxed{\text{\$+}} \quad \boxed{S} \boxed{D}$	• Links character string designated with (S) to character string designated with (D), and stores the result from (D) onward.		3	-	6-65
	\$+P	$\boxed{\text{\$+P}} \quad \boxed{S} \boxed{D}$					
	\$+	$\boxed{\text{\$+}} \quad \boxed{S1} \boxed{S2} \boxed{D}$	• Links character string designated with (S2) to character string designated with (S1), and stores the result from (D) onward.		4	-	6-67
	\$+P	$\boxed{\text{\$+P}} \quad \boxed{S1} \boxed{S2} \boxed{D}$					
BIN data increment	INC	$\boxed{\text{INC}} \quad \boxed{D}$	• (D)+1→(D)		2	●	6-69
	INCP	$\boxed{\text{INCP}} \quad \boxed{D}$					
	DINC	$\boxed{\text{DINC}} \quad \boxed{D}$	• (D+1, D)+1→(D+1, D)		*7	●	6-71
	DINCP	$\boxed{\text{DINCP}} \quad \boxed{D}$					
	DEC	$\boxed{\text{DEC}} \quad \boxed{D}$	• (D)-1→(D)		2	●	6-69
	DECP	$\boxed{\text{DECP}} \quad \boxed{D}$					
	DDEC	$\boxed{\text{DDEC}} \quad \boxed{D}$	• (D+1, D)-1→(D+1, D)		*7	●	6-71
	DDECP	$\boxed{\text{DDECP}} \quad \boxed{D}$					

\*7: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	3 Note 1)
	Devices other than above	2 Note 2)
Basic model QCPU Universal model QCPU	All devices that can be used	2 Note 2)

Note 1) When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.

Note 2) The number of steps may increase due to the conditions described in Section 3.8.

## 2.4.3 Data conversion instructions

Table 2.12 Data Conversion Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BCD conversions	BCD	$\boxed{\text{BCD}} \quad \boxed{S} \quad \boxed{D}$	BCD conversions · (S) $\xrightarrow{\quad}$ (D) ↑ BIN (0 to 9999)		3 *1	●	6-73
	BCDP	$\boxed{\text{BCDP}} \quad \boxed{S} \quad \boxed{D}$					
	DBCDCD	$\boxed{\text{DBCDCD}} \quad \boxed{S} \quad \boxed{D}$	BCD conversions · (S+1, S) $\xrightarrow{\quad}$ (D+1, D) ↑ BIN (0 to 99999999)		3 *1	●	
	DBCDCP	$\boxed{\text{DBCDCP}} \quad \boxed{S} \quad \boxed{D}$					
BIN conversions	BIN	$\boxed{\text{BIN}} \quad \boxed{S} \quad \boxed{D}$	BIN conversions · (S) $\xrightarrow{\quad}$ (D) ↑ BCD (0 to 9999)		3 *1	●	6-75
	BINP	$\boxed{\text{BINP}} \quad \boxed{S} \quad \boxed{D}$					
	DBIN	$\boxed{\text{DBIN}} \quad \boxed{S} \quad \boxed{D}$	BIN conversions · (S+1, S) $\xrightarrow{\quad}$ (D+1, D) ↑ BCD (0 to 99999999)		3 *1	●	
	DBINP	$\boxed{\text{DBINP}} \quad \boxed{S} \quad \boxed{D}$					
BIN ↓ Floating point conversions (Single precision)	FLT	$\boxed{\text{FLT}} \quad \boxed{S} \quad \boxed{D}$	Conversion to real number · (S) $\xrightarrow{\quad}$ (D+1, D) ↑ BIN( -32768 to 32767)		3 *1	● *2	6-78
	FLTP	$\boxed{\text{FLTP}} \quad \boxed{S} \quad \boxed{D}$					
	DFLT	$\boxed{\text{DFLT}} \quad \boxed{S} \quad \boxed{D}$	Conversion to real number · (S+1, S) $\xrightarrow{\quad}$ (D+1, D) ↑ BIN(-2147483648 to 2147483647)		3 *1	● *2	
	DFLTP	$\boxed{\text{DFLTP}} \quad \boxed{S} \quad \boxed{D}$					
BIN ↓ Floating point conversions (Double precision)	FLTD	$\boxed{\text{FLTD}} \quad \boxed{S} \quad \boxed{D}$	Conversion to real number · (S) $\xrightarrow{\quad}$ (D+3, D+2, D+1, D) ↑ BIN( -32768 to 32767)		4	● *2	6-81
	FLTDP	$\boxed{\text{FLTDP}} \quad \boxed{S} \quad \boxed{D}$					
	DFLTD	$\boxed{\text{DFLTD}} \quad \boxed{S} \quad \boxed{D}$	Conversion to real number · (S+1, S) $\xrightarrow{\quad}$ (D+3, D+2, D+1, D) ↑ BIN(-2147483648 to 2147483647)		4	● *2	
	DFLTDP	$\boxed{\text{DFLTDP}} \quad \boxed{S} \quad \boxed{D}$					
Floating point ↓ BIN conversions (Single precision)	INT	$\boxed{\text{INT}} \quad \boxed{S} \quad \boxed{D}$	Conversion to BIN · (S+1, S) $\xrightarrow{\quad}$ (D) ↑ Real number (-32768 to 32767)		3 *1	● *2	6-83
	INTP	$\boxed{\text{INTP}} \quad \boxed{S} \quad \boxed{D}$					
	DINT	$\boxed{\text{DINT}} \quad \boxed{S} \quad \boxed{D}$	Conversion to BIN · (S+1, S) $\xrightarrow{\quad}$ (D+1, D) ↑ Real number (-2147483648 to 2147483647)		3 *1	● *2	
	DINTP	$\boxed{\text{DINTP}} \quad \boxed{S} \quad \boxed{D}$					
Floating point ↓ BIN conversions (Double precision)	INTD	$\boxed{\text{INTD}} \quad \boxed{S} \quad \boxed{D}$	Conversion to BIN · (S+3, S+2, S+1, S) $\xrightarrow{\quad}$ (D) ↑ Real number ( -32768 to 32767)		3	● *2	6-86
	INTDP	$\boxed{\text{INTDP}} \quad \boxed{S} \quad \boxed{D}$					
	DINTD	$\boxed{\text{DINTD}} \quad \boxed{S} \quad \boxed{D}$	Conversion to BIN · (S+3, S+2, S+1, S) $\xrightarrow{\quad}$ (D+1, D) ↑ Real number ( -2147483648 to 2147483647)		3	● *2	
	DINTDP	$\boxed{\text{DINTDP}} \quad \boxed{S} \quad \boxed{D}$					

\*1: The number of basic steps is two for the Universal model QCPU only.

\*2: The subset is effective only with Universal model QCPU.



Table 2.12 Data Conversion Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description				
BIN 16-bit ↓ 32-bit conversion	DBL		Conversion (S) → (D+1, D) ↑ BIN (-32768 to 32767)		3	-	6-88				
	DBLP										
	WORD		Conversion (S+1, S) → (D) ↑ BIN (-32768 to 32767)		3	-	6-89				
	WORDP										
BIN ↓ Gray code conversions	GRY		Conversion to gray code (S) → (D) ↑ BIN (-32768 to 32767)		3	-	6-90				
	GRYP										
	DGRY		Conversion to gray code (S+1, S) → (D+1, D) ↑ BIN (-2147483648 to 2147483647)		3	-					
	DGRYP										
Gray code ↓ BIN conversions	GBIN		Conversion to BIN data (S) → (D) ↑ Gray code (-32768 to 32767)		3	-	6-92				
	GBINP										
	DGBIN		Conversion to BIN data (S+1, S) → (D+1, D) ↑ Gray code (-2147483648 to 2147483647)		3	-					
	DGBINP										
Complement to 2	NEG		(D) → (D) ↑ BIN data		2	-	6-94				
	NEGP										
	DNEG		(D+1, D) → (D+1, D) ↑ BIN data		2	-					
	DNEGP										
	ENEG		(D+1, D) → (D+1, D) ↑ Real number data		2	-	6-96				
	ENEGP										
	EDNEG		(D+3, D+2, D+1, D) → (D+3, D+2, D+1, D) ↑ Real number data		3	-	6-97				
	EDNEGP										
Block conversion	BKBCD		• Batch converts BIN data n points from (S) to BCD data and stores the result from (D) onward.		4	-	6-98				
	BKBCDP										
	BKBIN			• Batch converts BCD data n points from (S) to BIN data and stores the result from (D) onward.					4	-	6-100
	BKBINP										
Floating-point Single precision ↓ Double precision	ECON		Conversion to double precision (S+1, S) → (D+3, D+2, D+1, D) ↑ 32-bit floating-point real number		3	-	6-102				
	ECONP										
Floating-point Double precision ↓ Single precision	EDCON		Conversion to single precision (S+3, S+2, S+1, S) → (D+1, D) ↑ 64-bit floating-point real number		3	-	6-104				
	EDCONP										

## 2.4.4 Data transfer instructions

Table 2.13 Data Transfer Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
16-bit data transfer	MOV		$(S) \longrightarrow (D)$		*4	●	6-106
	MOVP				*1		
32-bit data transfer	DMOV		$(S+1,S) \longrightarrow (D+1,D)$		*2	●	
	DMOVP						
Floating decimal point data transfer (Single precision)	EMOV		$(S+1, S) \longrightarrow (D+1, D)$ Real number data		*2	● *3	6-108
	EMOVP						
Floating decimal point data transfer (Double precision)	EDMOV		$(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$ Real number data		2	● *3	6-110
	EDMOVP						
Character string data transfer	\$MOV		• Transfers character string designated by (S) to device designated by (D) onward.		3	-	6-112
	\$MOVP						
16-bit data negation transfer	CML		$(\bar{S}) \longrightarrow (D)$		*1	●	6-114
	CMLP						
32-bit data negation transfer	DCML		$(\bar{S+1}, S) \longrightarrow (D+1, D)$		*2	●	
	DCMLP						
Block transfer	BMOV		$(S) \longrightarrow (D)$ 		4	●	6-117
	BMOVP						
Identical 16-bit data block transfers	FMOV		$(S) \longrightarrow (D)$ 		4	●	6-120
	FMOVP						
Identical 32-bit data block transfers	DFMOV		$(S+1, S) \longrightarrow (D+1, D)$ 		4	●	
	DFMOVP						
16-bit data exchange	XCH		$(D1) \longleftrightarrow (D2)$		3	●	6-122
	XCHP						
32-bit data exchange	DXCH		$(D1+1, D1) \longleftrightarrow (D2+1, D2)$		3	●	
	DXCHP						

Table 2.13 Data Transfer Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Block data exchange	BXCH				4	-	6-126
	BXCHP						
Exchange of upper and lower bytes	SWAP				3	-	6-128
	SWAPP						

\*1: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
QCPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K4, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	2
	Devices other than above	3 Note 1)

Note 1) The number of steps may increase due to the conditions described in Section 3.8.

\*2: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	3
	Devices other than above	3 Note 1)
Basic model QCPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations (The number of steps is 3 when the above device + constant are used.)</li> </ul>	2
	Devices other than above	3 Note 1)
Universal model QCPU	All devices that can be used	2 Note 1)

Note 1) The number of steps may increase due to the conditions described in Section 3.8.

\*3: The subset is effective only with QCPU.

\*4: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
QCPU	<ul style="list-style-type: none"><li>• Word device: Internal device (except for file register ZR)</li><li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K4, and which use no indexing.</li><li>• Constant: No limitations</li></ul>	2
	Devices other than above	3 Note 1)

Note 1) The number of steps may increase due to the conditions described in Section 3.8.

## 2.4.5 Program branch instructions

Table 2.14 Program Branch Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Jump	CJ		• Jumps to Pn when input conditions are met.		2	●	6-129
	SCJ		• Jumps to Pn from the scan after the meeting of input condition.		2	●	
	JMP		• Jumps unconditionally to Pn.		2	●	
	GOEND		• Jumps to END instruction when input condition is met.		1	-	6-132

## 2.4.6 Program execution control instructions

Table 2.15 Program Execution Control Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Disable interrupts	DI		• Prohibits the running of an interrupt program.		1	-	6-133
Enable interrupts	EI		• Resets interrupt program execution prohibition.		1	-	
Interrupt disable/enable setting	IMASK		• Inhibits or permits interrupts for each interrupt program.		2	-	
Return	IRET		• Returns to sequence program from an interrupt program.		1	-	6-139

## 2.4.7 I/O refresh instructions

Table 2.16 I/O Refresh Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
I/O Refresh	RFS		• Refreshes the relevant I/O area during scan.		3	-	6-141
	RFSP						

## 2.4.8 Other convenient instructions

Table 2.17 Other convenient instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Up/Down counter	UDCNT1				4	-	6-143
	UDCNT2				4	-	6-146
Teaching timer	TTMR		<ul style="list-style-type: none"> <li>• (Time that TTMR is ON) <math>\times</math> n <math>\rightarrow</math> (D)</li> <li>n=0:1, n=1:10n, n=2:100</li> </ul>		3	-	6-149
Special timer	STMR		<ul style="list-style-type: none"> <li>• The 4 points from the bit device designated by (D) operate as shown below, depending on the ON/OFF status of the input conditions for the STMR instruction:</li> <li>(D)+0: Off delay timer output</li> <li>(D)+1: One shot after off timer output</li> <li>(D)+2: One shot after on timer output</li> <li>(D)+3: On delay and off delay timer output</li> </ul>		3	-	6-151
Shortest direction control	ROTC		<ul style="list-style-type: none"> <li>• Rotates a rotary table with n1 divisions from the stop position to the position designated by (S+1) in the shortest direction.</li> </ul>		5	-	6-154
Ramp signal	RAMP		<ul style="list-style-type: none"> <li>• Changes device data designated by D1 from n1 to n2 in n3 scans.</li> </ul>		6	-	6-157
Pulse density	SPD		<ul style="list-style-type: none"> <li>• Counts the pulse input from the device designated by (S) for the duration of time designated by n, and stores the count in the device designated by (D).</li> </ul>		4	-	6-160
Pulse output	PLSY		<ul style="list-style-type: none"> <li>• (n1)Hz <math>\rightarrow</math> (D)</li> <li>Output n2 times</li> </ul>		4	-	6-162
Pulse width modulation	PWM				4	-	6-164
Matrix input	MTR		<ul style="list-style-type: none"> <li>• Reads data of 16 points <math>\times</math> n rows from the devices starting from the one specified by (S), and stores them to the devices starting from the one specified by (D2).</li> </ul>		5	-	6-166

# 2.5 Application Instructions

## 2.5.1 Logical operation instructions

Table 2.18 Logical Operation Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Logical product	WAND		$(D) \wedge (S) \rightarrow (D)$		3	●	7-3
	WANDP		$(D) \wedge (S) \rightarrow (D)$		3	●	7-3
	WAND		$(S1) \wedge (S2) \rightarrow (D)$		4	●	7-6
	WANDP		$(S1) \wedge (S2) \rightarrow (D)$		4	●	7-6
	DAND		$(D+1, D) \wedge (S+1, S) \rightarrow (D+1, D)$		2	●	7-3
	DANDP		$(D+1, D) \wedge (S+1, S) \rightarrow (D+1, D)$		2	●	7-3
	DAND		$(S1+1, S1) \wedge (S2+1, S2) \rightarrow (D+1, D)$		3	●	7-6
	DANDP		$(S1+1, S1) \wedge (S2+1, S2) \rightarrow (D+1, D)$		3	●	7-6
	BKAND				5	-	7-9
	BKANDP				5	-	7-9
Logical sum	WOR		$(D) \vee (S) \rightarrow (D)$		3	●	7-11
	WORP		$(D) \vee (S) \rightarrow (D)$		3	●	7-11
	WOR		$(S1) \vee (S2) \rightarrow (D)$		4	●	7-14
	WORP		$(S1) \vee (S2) \rightarrow (D)$		4	●	7-14
	DOR		$(D+1, D) \vee (S+1, S) \rightarrow (D+1, D)$		2	●	7-11
	DORP		$(D+1, D) \vee (S+1, S) \rightarrow (D+1, D)$		2	●	7-11
	DOR		$(S1+1, S1) \vee (S2+1, S2) \rightarrow (D+1, D)$		3	●	7-14
	DORP		$(S1+1, S1) \vee (S2+1, S2) \rightarrow (D+1, D)$		3	●	7-14
	BKOR				5	-	7-17
	BKORP				5	-	7-17
Exclusive OR	WXOR		$(D) \veebar (S) \rightarrow (D)$		3	●	7-19
	WXORP		$(D) \veebar (S) \rightarrow (D)$		3	●	7-19
	WXOR		$(S1) \veebar (S2) \rightarrow (D)$		4	●	7-22
	WXORP		$(S1) \veebar (S2) \rightarrow (D)$		4	●	7-22
	DXOR		$(D+1, D) \veebar (S+1, S) \rightarrow (D+1, D)$		2	●	7-19
	DXORP		$(D+1, D) \veebar (S+1, S) \rightarrow (D+1, D)$		2	●	7-19

Table 2.18 Logical Operation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Exclusive OR	DXOR	$\overline{\text{DXOR}} \quad \text{S1} \quad \text{S2} \quad \text{D}$	$(\text{S1+1,S1}) \vee (\text{S2+1,S2}) \rightarrow (\text{D+1,D})$		*3	●	7-22
	DXORP	$\overline{\text{DXORP}} \quad \text{S1} \quad \text{S2} \quad \text{D}$					
	BKXOR	$\overline{\text{BKXOR}} \quad \text{S1} \quad \text{S2} \quad \text{D} \quad \text{n}$			5	-	7-25
	BKXORP	$\overline{\text{BKXORP}} \quad \text{S1} \quad \text{S2} \quad \text{D} \quad \text{n}$					
NON exclusive logical sum	WXNR	$\overline{\text{WXNR}} \quad \text{S} \quad \text{D}$	$\overline{(\text{D})} \vee (\text{S}) \rightarrow (\text{D})$		3	●	7-27
	WXNRP	$\overline{\text{WXNRP}} \quad \text{S} \quad \text{D}$					
	WXNR	$\overline{\text{WXNR}} \quad \text{S1} \quad \text{S2} \quad \text{D}$	$(\text{S1}) \vee (\text{S2}) \rightarrow (\text{D})$		4	●	7-30
	WXNRP	$\overline{\text{WXNRP}} \quad \text{S1} \quad \text{S2} \quad \text{D}$					
	DXNR	$\overline{\text{DXNR}} \quad \text{S} \quad \text{D}$	$\overline{(\text{D+1,D})} \vee (\text{S+1,S}) \rightarrow (\text{D+1,D})$		*2	●	7-27
	DXNRP	$\overline{\text{DXNRP}} \quad \text{S} \quad \text{D}$					
	DXNR	$\overline{\text{DXNR}} \quad \text{S1} \quad \text{S2} \quad \text{D}$	$(\text{S1+1,S1}) \vee (\text{S2+1,S2}) \rightarrow (\text{D+1,D})$		*3	●	7-30
	DXNRP	$\overline{\text{DXNRP}} \quad \text{S1} \quad \text{S2} \quad \text{D}$					
	BKXNR	$\overline{\text{BKXNR}} \quad \text{S1} \quad \text{S2} \quad \text{D} \quad \text{n}$			5	-	7-33
	BKXNRP	$\overline{\text{BKXNRP}} \quad \text{S1} \quad \text{S2} \quad \text{D} \quad \text{n}$					



\*1: The number of basic steps is three for the Universal model QCPU only.

\*2: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	5 Note 1)
	Devices other than above	3 Note 2)
Basic model QCPU Universal model QCPU	All devices that can be used	3 Note 2)

Note 1) When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.

Note 2) The number of steps may increase due to the conditions described in Section 3.8.

\*3: The number of steps may vary depending on the device and type of CPU module being used.

Component	Device	Number of Steps
High Performance model QCPU Process CPU Redundant CPU	<ul style="list-style-type: none"> <li>• Word device: Internal device (except for file register ZR)</li> <li>• Bit device: Devices whose device Nos. are multiples of 16, whose digit designation is K8, and which use no indexing.</li> <li>• Constant: No limitations</li> </ul>	6 Note 1)
	Devices other than above	4 Note 2)
Basic model QCPU	All devices that can be used	4 Note 2)
Universal model QCPU		3 Note 2)

Note 1) When using a High Performance model QCPU, Process CPU or Redundant CPU, the number of steps increases but the processing speed becomes faster.

Note 2) The number of steps may increase due to the conditions described in Section 3.8.

## 2.5.2 Rotation instructions

Table 2.19 Rotation Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Right rotation	ROR				3	●	7-35
	RORP				3	●	
	RORP				3	●	
	RORP				3	●	
Left rotation	ROL				3	●	7-38
	ROLP				3	●	
	ROLP				3	●	
	ROLP				3	●	
Right rotation	DROR				3	●	7-41
	DRORP				3	●	
	DRORP				3	●	
	DRORP				3	●	
Left rotation	DROL				3	●	7-44
	DROLP				3	●	
	DROLP				3	●	
	DROLP				3	●	

## 2.5.3 Shift instructions

Table 2.20 Shift Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
n-bit shift of 16-bit data	SFR				3	●	7-46
	SFRP						
	SFL				3	●	
	SFLP						
1-bit shift of n-bit data	BSFR				3	-	7-49
	BSFRP						
	BSFL				3	-	
	BSFLP						
n-bit shift of n-bit data	SFTBR				4	-	7-51
	SFTBRP						
	SFTBL				4	-	
	SFTBLP						
1-word shift of n-words data	DSFR				3	●	7-54
	DSFRP						
	DSFL				3	●	
	DSFLP						
n-words shift of n-words data	SFTWR				4	-	7-56
	SFTWRP						
	SFTWL				4	-	
	SFTWLP						

## 2.5.4 Bit processing instructions

Table 2.21 Bit Processing Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Bit set/reset	BSET				3	●	7-59
	BSETP						
	BRST				3	●	
	BRSTP						
Bit tests	TEST				4	-	7-61
	TESTP						
	DTEST				4	-	
	DTESTP						
Batch reset of bit devices	BKRST				3	-	7-64
	BKRSTP						

## 2.5.5 Data processing instructions

Table 2.22 Data Processing Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Data searches	SER				5	-	7-66
	SERP						
	DSER				5	-	
	DSERP						
Bit checks	SUM				3	●	7-69
	SUMP						
	DSUM				3	●	
	DSUMP						
Decode	DECO				4	-	7-71
	DECOP						
Encode	ENCO				4	-	7-73
	ENCOP						
7-seg- ment decode	SEG				3	●	7-75
	SEGP						

Table 2.22 Data Processing Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Separating and linking	DIS		• Separates 16-bit data designated by (S) into 4-bit units, and stores at the lower 4 bits of n points from (D). ( $n \leq 4$ )		4	-	7-77
	DISP						
	UNI		• Links the lower 4 bits of n points from the device designated by (S) and stores at the device designated by (D). ( $n \leq 4$ )		4	-	7-79
	UNIP						
	NDIS		• Separates the data in the devices starting from the one specified by (S1) into bits specified by the devices from (S2), and stores them to the devices starting from the one specified by (D).		4	-	7-81
	NDISP						
	NUNI		• Links the data in the devices starting from the one specified by (S1) with bits specified by the devices from (S2), and stores them to the devices starting from the one specified by (D).		4	-	7-85
	NUNIP						
	WTOB		• Breaks n points of 16-bit data from the device designated by (S) into 8-bit units, and stores in sequence at the device designated by (D).		4	-	7-85
	WTOBP						
	BTOW		• Links the lower 8 bits of 16-bit data of n points from the device designated by (S) into 16-bit units, and stores in sequence at the device designated by (D).		4	-	7-85
	BTOWP						
Search	MAX		• Searches the data of n points from the device designated by (S) in 16-bit units, and stores the maximum value at the device designated by (D).		4	-	7-89
	MAXP						
	MIN		• Searches the data of n points from the device designated by (S) in 16-bit units, and stores the minimum value at the device designated by (D).		4	-	7-92
	MINP						
	DMAX		• Searches the data of 2n points from the device designated by (S) in 32-bit units, and stores the maximum value at the device designated by (D).		4	-	7-89
	DMAXP						
	DMIN		• Searches the data of 2n points from the device designated by (S) in 32-bit units, and stores the minimum value at the device designated by (D).		4	-	7-92
	DMINP						

Table 2.22 Data Processing Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Sort	SORT	<ul style="list-style-type: none"> <li>· S2: Number of comparisons to be made during a single run</li> <li>· D1: Device to be turned ON at the completion of sort</li> <li>· D2: For system use</li> </ul>	<ul style="list-style-type: none"> <li>· Sorts data of n points from device designated by (S1) in 16-bit units. (n x (n-1)/2 scans required)</li> </ul>		6	-	7-95
	DSORT	<ul style="list-style-type: none"> <li>· S2: Number of comparisons to be made during a single run</li> <li>· D1: Device to be turned ON at the completion of sort</li> <li>· D2: For system use</li> </ul>	<ul style="list-style-type: none"> <li>· Sorts data of 2n points from device designated by (S1) in 32-bit units. (n x (n-1)/2 scans required)</li> </ul>				
Total value calculations	WSUM		<ul style="list-style-type: none"> <li>· Adds 16 bit BIN data of n points from the device specified by (S), and stores it in the device specified by (D).</li> </ul>		4	-	7-99
	WSUMP						
	DWSUM		<ul style="list-style-type: none"> <li>· Adds 32 bit BIN data of n points from the device specified by (S), and stores it in the device specified by (D).</li> </ul>				7-101
	DWSUMP						
Calculation of averages	MEAN		<ul style="list-style-type: none"> <li>· Calculates the mean of n-point devices (in 16-bit units) starting from the device specified by (S), and then stores the result into the device specified by (D).</li> </ul>		4	-	7-103
	MEANP						
	DMEAN		<ul style="list-style-type: none"> <li>· Calculates the mean of n-point devices (in 32-bit units) starting from the device specified by (S), and then stores the result into the device specified by (D).</li> </ul>				
	DMEANP						

## 2.5.6 Structure creation instructions

Table 2.23 Structure Creation Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Number of repeats	FOR		• Executes n times between the <b>FOR</b> and <b>NEXT</b> .		2	-	7-105
	NEXT				1	-	
	BREAK		• Forcibly ends the execution of the <b>FOR</b> to <b>NEXT</b> cycle and jumps pointer Pn.		3	-	7-108
	BREAKP						
Subroutine program calls	CALL		• Executes subroutine program Pn when input condition is met. (S1 to Sn are arguments sent to subroutine program. n ≦ 5)		*1 2 +	*3	7-110
	CALLP						
	RET		• Returns from subroutine program		1	-	7-115
	FCALL		• Performs non-execution processing of subroutine program Pn if input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. n ≦ 5)		*1 2 +	-	7-116
	FCALLP						
	ECALL		• Executes subroutine program Pn from within designated program name when input condition is met. (S1 to Sn are arguments sent to subroutine program. n ≦ 5)		*2 3 +	-	7-120
ECALLP		n					









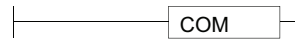


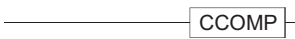

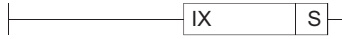
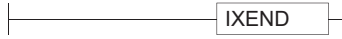


\*1: n indicates number of arguments for subroutine program.

\*2: n indicates the total of the number of arguments used in the subroutine program and the number of program name steps. The number of program name steps is calculated as "number of characters in the program/2" (decimal fraction is rounded up).

\*3: The subset is effective only with the Universal model QCPU.



Table 2.23 Structure Creation Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Subroutine program calls	EFCALL	  * :File name	<ul style="list-style-type: none"> <li>Performs non-execution processing of subroutine program Pn if input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. N ≦ 5)</li> </ul>		*2 3 + n	-	7-125
	EFCALLP	  * :File name					
	XCALL		<ul style="list-style-type: none"> <li>Executes subroutine program Pn when input condition is met.</li> <li>Performs non-execution processing of subroutine program Pn if input conditions have not been met. (S1 to Sn are arguments sent to subroutine program. N ≦ 5)</li> </ul>		*1 2 + n	-	7-129
Select refresh	COM		<ul style="list-style-type: none"> <li>Performs auto refresh of intelligent function modules, link refresh, auto refresh of CPU shared memory, and communications with peripherals.</li> </ul>		1	-	7-134
	CCOM		<ul style="list-style-type: none"> <li>Performs auto refresh of intelligent function modules, auto refresh of CPU shared memory, and communications with peripherals after the input conditions are met.</li> </ul>		1	-	7-141
	COM				1	-	7-137
Fixed indexing	IX		<ul style="list-style-type: none"> <li>Perform indexing for individual devices used in device indexing ladder.</li> </ul>		2	-	7-144
	IXEND			1	-		
	IXDEV		<ul style="list-style-type: none"> <li>Stores indexing value used for indexing performed between the [IX] and [IXEND] to the device designated by D or later.</li> </ul>		1	-	7-148
	IXSET	 Designates indexing value.		3	-		

\*1: n indicates number of arguments for subroutine program.

\*2: n indicates the total of the number of arguments used in the subroutine program and the number of program name steps. The number of program name steps is calculated as "number of characters in the program/2" (decimal fraction is rounded up).

## 2.5.7 Data table operation instructions

Table 2.24 Data table Operation Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Data table processing	FIFW		(S)  (D) Pointer Pointer + 1		3	-	7-151
	FIFWP		Device at pointer + 1				
	FIFR		(S) Pointer Pointer - 1 (D)		3	-	7-153
	FIFRP						
	FPOP		(S) Pointer Pointer - 1 (D)		3	-	7-155
	FPOPP		Device at pointer + 1				
	FDEL		(S) Pointer Pointer - 1 (D)		4	-	7-157
	FDELP		Designated by n				
	FINS		(S)  (D) Pointer Pointer + 1		4	-	
	FINSP		Designated by n				

## 2.5.8 Buffer memory access instructions

Table 2.25 Buffer Memory Access Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Data read	FROM		• Reads data in 16-bit units from an intelligent function module.		5	-	7-160
	FROMP						
	DFRO		• Reads data in 32-bit units from an intelligent function module.		5	-	
	DFROP						
Data write	TO		• Writes data in 16-bit units to an intelligent function module.		5	-	7-163
	TOP						
	DTO		• Writes data in 32-bit units to an intelligent function module.		5	-	
	DTOP						





## 2.5.9 Display instructions

Table 2.26 Display Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
ASCII print	PR	* When SM701 is OFF 	• Outputs ASCII code of 8 points (16 characters) from device designated by (S) to output module.		3	-	7-166
	PR	* When SM701 is ON 	• Outputs ASCII code from device designated by (S) to 00H to output module.				
	PRC		• Converts comments from device designated by (S) to ASCII code and outputs to output module.				7-169
Reset	LEDR		• Resets annunciator and LED indicator display.		1	-	7-172

## 2.5.10 Debugging and failure diagnosis instructions

Table 2.27 Debugging and Failure Diagnosis Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Checks	CHKST		<ul style="list-style-type: none"> <li>The CHK instruction is executed when CHKST is executable.</li> <li>Jumps to the step following the CHK instruction when CHKST is in a non-executable status.</li> </ul>		1	-	7-175
	CHK		<ul style="list-style-type: none"> <li>During normal conditions → SM80 : OFF, SD80 : 0</li> <li>During abnormal conditions → SM80 : ON, SD80 : Failure No.</li> </ul>				
	CHKCIR		<ul style="list-style-type: none"> <li>Starts update in ladder pattern being checked by the CHK instruction.</li> </ul>		1	-	7-179
	CHKEND		<ul style="list-style-type: none"> <li>Ends update in ladder pattern being checked by the CHK instruction.</li> </ul>				

## 2.5.11 Character string processing instructions

Table 2.28 Character String Processing Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
BIN ↓ Decimal ASCII	BINDA	— BINDA S D —	• Converts 1-word BIN value designated by (S) to a 5-digit, decimal ASCII value, and stores it at the word device designated by (D).		3	-	7-183
	BINDAP	— BINDAP S D —					
	DBINDA	— DBINDA S D —	• Converts 2-word BIN value designated by (S) to a 10-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	-	
	DBINDAP	— DBINDAP S D —					
BIN ↓ Hexadecimal ASCII	BINHA	— BINHA S D —	• Converts 1-word BIN value designated by (S) to a 4-digit, hexadecimal ASCII value, and stores it at a word device following the word device number designated by (D).		3	-	7-186
	BINHAP	— BINHAP S D —					
	DBINHA	— DBINHA S D —	• Converts 2-word BIN value designated by (S) to an 8-digit, hexadecimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	-	
	DBINHAP	— DBINHAP S D —					
BCD ↓ Decimal ASCII	BCDDA	— BCDDA S D —	• Converts 1-word BCD value designated by (S) to a 4-digit, decimal ASCII value, and stores it at a word device following the word device number designated by (D).		3	-	7-189
	BCDDAP	— BCDDAP S D —					
	DBCDDA	— DBCDDA S D —	• Converts 2-word BCD value designated by (S) to an 8-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).		3	-	
	DBCDDAP	— DBCDDAP S D —					
Decimal ASCII ↓ BIN	DABIN	— DABIN S D —	• Converts a 5-digit, decimal ASCII value designated by (S) to a 1-word BIN value, and stores it at a word device number designated by (D).		3	-	7-192
	DABINP	— DABINP S D —					
	DDABIN	— DDABIN S D —	• Converts a 10-digit, decimal ASCII value designated by (S) to a 2-word BIN value, and stores it at a word device number designated by (D).		3	-	
	DDABINP	— DDABINP S D —					
Hexadecimal ASCII ↓ BIN	HABIN	— HABIN S D —	• Converts a 4-digit, hexadecimal ASCII value designated by (S) to a 1-word BIN value, and stores it at a word device number designated by (D).		3	-	7-195
	HABINP	— HABINP S D —					
	DHABIN	— DHABIN S D —	• Converts an 8-digit, hexadecimal ASCII value designated by (S) to a 2-word BIN value, and stores it at a word device number designated by (D).		3	-	
	DHABINP	— DHABINP S D —					

Table 2.28 Character String Processing Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Decimal ASCII ↓ BCD	DABCD		• Converts a 4-digit, decimal ASCII value designated by (S) to a 1-word BCD value, and stores it at a device number designated by (D).		3	-	7-198
	DABCDP						
	DDABCD		• Converts a 8-digit decimal ASCII value designated by (S) to a 2-word BCD value, and stores it at the word device number designated by (D).		3	-	
	DDABCDP						
Device comment read operation	COMRD		• Stores comment from device designated by (S) at a device designated by (D).		3	-	7-201
	COMRDP						
Character string length detection	LEN		• Stores data length (number of characters) in character string designated by (S) at a device designated by (D).		3	-	7-204
	LENP						
BIN ↓ Decimal character string	STR		• Converts a 1-word BIN value designated by (S2) to a decimal character string with the total number of digits and the number of decimal fraction digits designated by (S1) and stores them at a device designated by (D).		4	-	7-206
	STRP						
	DSTR		• Converts a 2-word BIN value designated by (S2) to a decimal character string with the total number of digits and the number of decimal fraction digits designated by (S1) and stores them at a device designated by (D).		4	-	
	DSTRP						
Decimal character string ↓ BIN	VAL		• Converts a character string including decimal point designated by (S) to a 1-word BIN value and the number of decimal fraction digits, and stores them into devices designated by (D1) and (D2).		4	-	7-212
	VALP						
	DVAL		• Converts a character string including decimal point designated by (S) to a 2-word BIN value and the number of decimal fraction digits, and stores them into devices designated by (D1) and (D2).		4	-	
	DVALP						
Floating decimal point ↓ Character string	ESTR		• Converts the 32-bit floating decimal point data designated by (S) to a character string, and stores it in devices designated by (D).		4	-	7-217
	ESTRP						
Character string ↓ Floating decimal point	EVAL		• Converts the character string designated by (S) to a 32-bit floating decimal point data, and stores it in devices designated by (D).		3	-	7-224
	EVALP						

Table 2.28 Character String Processing Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Hexadecimal BIN ↓ ASCII	ASC	— ASC S D n —	• Converts the 1-word BIN value at the device numbers designated by (S) to hexadecimal ASCII, and stores n characters of them at the device numbers designated by (D) and after.		4	-	7-228
	ASCP	— ASCP S D n —					
ASCII ↓ Hexadecimal BIN	HEX	— HEX S D n —	• Converts n hexadecimal ASCII characters of the device numbers designated by (S) and after to BIN values, and stores them at the device numbers designated by (D).		4	-	7-230
	HEXP	— HEXP S D n —					
Character string	RIGHT	— RIGHT S D n —	• Stores n characters from the end of a character string designated by (S) at the device designated by (D).		4	-	7-232
	RIGHTP	— RIGHTP S D n —					
	LEFT	— LEFT S D n —	• Stores n characters from the beginning of a character string designated by (S) at the device designated by (D).		4	-	7-232
	LEFTP	— LEFTP S D n —					
	MIDR	— MIDR S1 D S2 —	• Stores the designated number of characters in the character string designated by (S1) from the position designated by (S2) at the device designated by (D).		4	-	7-235
	MIDRP	— MIDRP S1 D S2 —					
	MIDW	— MIDW S1 D S2 —	• Stores the character string of (S1) in the specified number to the character string of (D) at the position specified by (S2).		4	-	7-235
	MIDWP	— MIDWP S1 D S2 —					
	INSTR	— INSTR S1 S2 D n —	• Searches character string (S1) from the nth character of character string (S2), and stores matched positions at (D).		5	-	7-239
	INSTRP	— INSTRP S1 S2 D n —					
	STRINS	— STRINS S D n —	• Inserts the character string data specified by (S) to the (n)th character (insert position) from the initial character string data specified by (D).		4	-	7-241
	STRINSP	— STRINSP S D n —					
STRDEL	— STRDEL D n1 n2 —	• Deletes the (n2) characters data specified by (D) starting from the device(insert position) specified by n1.		4	-	7-243	
STRDELP	— STRDELP D n1 n2 —						
Floating decimal point ↓ BCD	EMOD	— EMOD S1 S2 D —	• Converts 32-bit floating decimal point data (S1) to BCD data with number of decimal fraction digits designated by (S2), and stores at device designated by (D).		4	-	7-245
	EMODP	— EMODP S1 S2 D —					
BCD ↓ Floating decimal point	EREXP	— EREXP S1 S2 D —	• Converts BCD data (S1) to 32-bit floating decimal point data with the number of decimal fraction digits designated by (S2), and stores at device designated by (D).		4	-	7-248
	EREXPP	— EREXPP S1 S2 D —					

## 2.5.12 Special function instructions

Table 2.29 Special Function Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Trigonometric functions (Floating-point single-precision)	SIN		• $\text{Sin}(S+1, S) \longrightarrow (D+1, D)$		3	-	7-250
	SINP						
	COS		• $\text{Cos}(S+1, S) \longrightarrow (D+1, D)$		3	-	7-254
	COSP						
	TAN		• $\text{Tan}(S+1, S) \longrightarrow (D+1, D)$		3	-	7-258
	TANP						
	ASIN		• $\text{Sin}^{-1}(S+1, S) \longrightarrow (D+1, D)$		3	-	7-262
	ASINP						
	ACOS		• $\text{Cos}^{-1}(S+1, S) \longrightarrow (D+1, D)$		3	-	7-267
	ACOSP						
	ATAN		• $\text{Tan}^{-1}(S+1, S) \longrightarrow (D+1, D)$		3	-	7-271
	ATANP						
Trigonometric functions (Floating-point double-precision)	SIND		$\text{Sin}(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$		3	-	7-252
	SINDP						
	COSD		$\text{Cos}(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$		3	-	7-256
	COSDP						
	TAND		$\text{Tan}(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$		3	-	7-260
	TANDP						
	ASIND		$\text{Sin}^{-1}(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$		3	-	7-265
	ASINDP						
	ACOSD		$\text{Cos}^{-1}(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$		3	-	7-269
	ACOSDP						
	ATAND		$\text{Tan}^{-1}(S+3, S+2, S+1, S) \longrightarrow (D+3, D+2, D+1, D)$		3	-	7-273
	ATANDP						



Table 2.29 Special Function Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Angles ↕ Radians conversion	RAD	$\boxed{\text{RAD}} \quad \boxed{S} \quad \boxed{D}$	• $(S+1, S) \rightarrow (D+1, D)$ Conversion from angles to radians		3	-	7-275
	RADP	$\boxed{\text{RADP}} \quad \boxed{S} \quad \boxed{D}$					
	RADD	$\boxed{\text{RADD}} \quad \boxed{S} \quad \boxed{D}$	• $(S+3, S+2, S+1, S) \rightarrow (D+3, D+2, D+1, D)$ Conversion from angle to radian		3	-	7-277
	RADDP	$\boxed{\text{RADDP}} \quad \boxed{S} \quad \boxed{D}$					
	DEG	$\boxed{\text{DEG}} \quad \boxed{S} \quad \boxed{D}$	• $(S+1, S) \rightarrow (D+1, D)$ Conversion from radians to angles		3	-	7-279
	DEGP	$\boxed{\text{DEGP}} \quad \boxed{S} \quad \boxed{D}$					
	DEGD	$\boxed{\text{DEGD}} \quad \boxed{S} \quad \boxed{D}$	• $(S+3, S+2, S+1, S) \rightarrow (D+3, D+2, D+1, D)$ Conversion from radian to angle		3	-	7-281
	DEGDP	$\boxed{\text{DEGDP}} \quad \boxed{S} \quad \boxed{D}$					
Square root	SQR	$\boxed{\text{SQR}} \quad \boxed{S} \quad \boxed{D}$	• $\sqrt{(S+1, S)} \rightarrow (D+1, D)$		3	-	7-287
	SQRP	$\boxed{\text{SQRP}} \quad \boxed{S} \quad \boxed{D}$					
	SQRD	$\boxed{\text{SQRD}} \quad \boxed{S} \quad \boxed{D}$	$\sqrt{(S+3, S+2, S+1, S)} \rightarrow (D+3, D+2, D+1, D)$		3	-	7-289
	SQRDP	$\boxed{\text{SQRDP}} \quad \boxed{S} \quad \boxed{D}$					
Exponent operations	EXP	$\boxed{\text{EXP}} \quad \boxed{S} \quad \boxed{D}$	• $e^{(S+1, S)} \rightarrow (D+1, D)$		3	-	7-291
	EXPP	$\boxed{\text{EXPP}} \quad \boxed{S} \quad \boxed{D}$					
	EXPD	$\boxed{\text{EXPD}} \quad \boxed{S} \quad \boxed{D}$	$e^{(S+3, S+2, S+1, S)} \rightarrow (D+3, D+2, D+1, D)$		3	-	7-294
	EXPDP	$\boxed{\text{EXPDP}} \quad \boxed{S} \quad \boxed{D}$					
Natural logarithms	LOG	$\boxed{\text{LOG}} \quad \boxed{S} \quad \boxed{D}$	• $\text{Log}_e(S+1, S) \rightarrow (D+1, D)$		3	-	7-296
	LOGP	$\boxed{\text{LOGP}} \quad \boxed{S} \quad \boxed{D}$					
	LOGD	$\boxed{\text{LOGD}} \quad \boxed{S} \quad \boxed{D}$	$\text{Log}_e(S+3, S+2, S+1, S) \rightarrow (D+3, D+2, D+1, D)$		3	-	7-298
	LOGDP	$\boxed{\text{LOGDP}} \quad \boxed{S} \quad \boxed{D}$					
Exponentiation	POW	$\boxed{\text{POW}} \quad \boxed{S1} \quad \boxed{S2} \quad \boxed{D}$	• $(S1+1, S1)^{(S2+1, S2)} \rightarrow (D+1, D)$		4	-	7-300
	POWP	$\boxed{\text{POWP}} \quad \boxed{S1} \quad \boxed{S2} \quad \boxed{D}$					
	POWD	$\boxed{\text{POWD}} \quad \boxed{S1} \quad \boxed{S2} \quad \boxed{D}$	• $(S1+3, S1+2, S1+1, S1)^{(S2+3, S2+2, S2+1, S2)} \rightarrow (D+3, D+2, D+1, D)$		4	-	7-302
	POWDP	$\boxed{\text{POWDP}} \quad \boxed{S1} \quad \boxed{S2} \quad \boxed{D}$					
Common logarithm	LOG10	$\boxed{\text{LOG10}} \quad \boxed{S} \quad \boxed{D}$	• $\log_{10}(S+1, S) \rightarrow (D+1, D)$		3	-	7-300
	LOG10P	$\boxed{\text{LOG10P}} \quad \boxed{S} \quad \boxed{D}$					
	LOG10D	$\boxed{\text{LOG10D}} \quad \boxed{S} \quad \boxed{D}$	$\log_{10}(S+3, S+2S+1, S) \rightarrow (D+3, D+2, D+1, D)$		3	-	7-302
	LOG10DP	$\boxed{\text{LOG10DP}} \quad \boxed{S} \quad \boxed{D}$					

Table 2.29 Special Function Instructions (Continued)

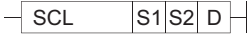

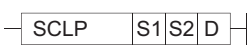

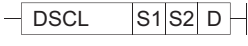

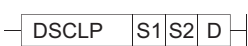

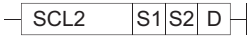

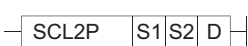

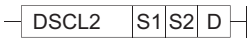



Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Random number generation	RND		• Generates a random number (from 0 to less than 32767) and stores it at the device designated by (D).		2	-	7-304
	RNDP						
Random number series update	SRND		• Updates random number series according to the 16-bit BIN data stored in the device designated by (S).		2	-	7-304
	SRNDP						
Square root	BSQR		• $\sqrt{S} \rightarrow (D)+0$ +1 Integer part +1 Decimal fraction part		3	-	7-306
	BSQRP						
	BDSQR		• $\sqrt{S+1, S} \rightarrow (D)+0$ +1 Integer part +1 Decimal fraction part		3	-	
	BDSQRP						
Trigonometric functions	BSIN		• $\sin(S) \rightarrow (D)+0$ +1 Sign +1 Integer part +2 Decimal fraction part		3	-	7-309
	BSINP						
	BCOS		• $\cos(S) \rightarrow (D)+0$ +1 Sign +1 Integer part +2 Decimal fraction part		3	-	7-311
	BCOSP						
	BTAN		• $\tan(S) \rightarrow (D)+0$ +1 Sign +1 Integer part +2 Decimal fraction part		3	-	7-313
	BTANP						
	BASIN		• $\sin^{-1}(S) \rightarrow (D)+0$ +1 Sign +1 Integer part +2 Decimal fraction part		3	-	7-315
	BASINP						
	BACOS		• $\cos^{-1}(S) \rightarrow (D)+0$ +1 Sign +1 Integer part +2 Decimal fraction part		3	-	7-317
	BACOSP						
	BATAN		• $\tan^{-1}(S) \rightarrow (D)+0$ +1 Sign +1 Integer part +2 Decimal fraction part		3	-	7-319
	BATANP						

## 2.5.13 Data control instructions

Table 2.30 Data Control Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Upper and lower limit controls	LIMIT		<ul style="list-style-type: none"> <li>When <math>(S3) &lt; (S1)</math> ..... Stores value of <math>(S1)</math> at <math>(D)</math></li> </ul>		5	-	7-321
	LIMITP		<ul style="list-style-type: none"> <li>When <math>(S1) \leq (S3) \leq (S2)</math> ..... Stores value of <math>(S3)</math> at <math>(D)</math></li> <li>When <math>(S2) &lt; (S3)</math> ..... Stores value of <math>(S2)</math> at <math>(D)</math></li> </ul>				
	DLIMIT		<ul style="list-style-type: none"> <li>When <math>((S3)+1, (S3)) &lt; ((S1)+1, S1)</math> .. Stores value of <math>((S1)+1, (S1))</math> at <math>((D)+1, (D))</math></li> <li>When <math>((S1)+1, (S1)) \leq ((S3)+1, (S3)) &lt; (S2+1, S2)</math> .. Stores value of <math>((S3)+1, (S3))</math> at <math>((D)+1, (D))</math></li> </ul>		5	-	
	DLIMITP		<ul style="list-style-type: none"> <li>When <math>((S2), (S2)+1) &lt; ((S3), (S3)+1)</math> .. Stores value of <math>((S2)+1, (S2))</math> at <math>((D)+1, (D))</math></li> </ul>				
Dead band controls	BAND		<ul style="list-style-type: none"> <li>When <math>(S1) \leq (S3) \leq (S2)</math>..... <math>0 \rightarrow (D)</math></li> <li>When <math>(S3) &lt; (S1)</math>..... <math>(S3) - (S1) \rightarrow (D)</math></li> <li>When <math>(S2) &lt; (S3)</math>..... <math>(S3) - (S2) \rightarrow (D)</math></li> </ul>		5	-	7-324
	BANDP						
	DBAND		<ul style="list-style-type: none"> <li>When <math>((S1)+1, (S1)) \leq ((S3)+1, (S3)) \leq ((S2)+1, (S2))</math>..... <math>0 \rightarrow ((D)+1, (D))</math></li> <li>When <math>((S3)+1, (S3)) &lt; ((S1)+1, (S1))</math>..... <math>((S3)+1, (S3)) - ((S1)+1, (S1)) \rightarrow ((D)+1, (D))</math></li> </ul>		5	-	
	DBANDP		<ul style="list-style-type: none"> <li>When <math>((S2)+1, (S2)) &lt; ((S3)+1, (S3))</math>..... <math>((S3)+1, (S3)) - ((S2)+1, (S2)) \rightarrow ((D)+1, (D))</math></li> </ul>				
Zone controls	ZONE		<ul style="list-style-type: none"> <li>When <math>(S3) = 0</math>..... <math>0 \rightarrow (D)</math></li> <li>When <math>(S3) &gt; 0</math>..... <math>(S3)+(S2) \rightarrow (D)</math></li> <li>When <math>(S3) &lt; 0</math>..... <math>(S3) - (S1) \rightarrow (D)</math></li> </ul>		5	-	7-327
	ZONEP						
	DZONE		<ul style="list-style-type: none"> <li>When <math>((S3)+1, (S3)) = 0</math> ..... <math>0 \rightarrow ((D)+1, (D))</math></li> <li>When <math>((S3)+1, (S3)) &gt; 0</math> ..... <math>((S3)+1, (S3)) + ((S2)+1, (S2)) \rightarrow ((D)+1, (D))</math></li> </ul>		5	-	
	DZONEP		<ul style="list-style-type: none"> <li>When <math>((S3)+1, (S3)) &lt; 0</math> ..... <math>((S3)+1, (S3)) + ((S1)+1, (S1)) \rightarrow ((D)+1, (D))</math></li> </ul>				

Table 2.30 Special Function Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Point-by-point coordinate data	SCL		<ul style="list-style-type: none"> <li>Executes scaling for the scaling conversion data (16-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.</li> </ul>		4	-	7-330
	SCLP						
	DSCL		<ul style="list-style-type: none"> <li>Executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.</li> </ul>		4	-	
	DSCLP						
X or Y coordinate data	SCL2		<ul style="list-style-type: none"> <li>Executes scaling for the scaling conversion data (16-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.</li> </ul>		4	-	7-334
	SCL2P						
	DSCL2		<ul style="list-style-type: none"> <li>Executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified by (S1), and then stores the result into the device specified by (D). The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.</li> </ul>		4	-	
	DSCL2P						

## 2.5.14 Switching instructions

Table 2.31 Switching Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Block number switching	RSET		<ul style="list-style-type: none"> <li>Converts extension file register block number to number designated by (S).</li> </ul>		2	-	7-337
	RSETP						
File set	QDRSET		<ul style="list-style-type: none"> <li>Sets file names used as file registers.</li> </ul>		*1	-	7-339
	QDRSETP				2 +		
	QCDSET		<ul style="list-style-type: none"> <li>Sets file names used as comment files.</li> </ul>		*1	-	7-342
	QCDSETP				2 +		

\*1:  $n$  ( $[(\text{number of file name characters}) / 2]$ ) indicates a step. (Decimal fractions are rounded up.)

## 2.5.15 Clock instructions

Table 2.32 Clock Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Read/ write clock data	DATERD	$\boxed{\text{DATERD}} \boxed{D}$	• (Clock elements) → (D)+0 +1 Year +2 Month +3 Day +4 Hour +5 Minute +6 Sec. +6 Day of the week		2	-	7-344
	DATERDP	$\boxed{\text{DATERDP}} \boxed{D}$					
	DATEWR	$\boxed{\text{DATEWR}} \boxed{S}$	• (D)+0 +1 Year +2 Month +3 Day +4 Hour +5 Minute +6 Sec. +6 Day of the week	$\rightarrow$ (Clock elements) 	2	-	7-346
	DATEWRP	$\boxed{\text{DATEWRP}} \boxed{S}$					
Clock data addition/ subtraction	DATE+	$\boxed{\text{DATE+}} \boxed{S1} \boxed{S2} \boxed{D}$	$\begin{matrix} (S1) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix} + \begin{matrix} (S2) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix} \rightarrow \begin{matrix} (D) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix}$		4	-	7-348
	DATE+P	$\boxed{\text{DATE+P}} \boxed{S1} \boxed{S2} \boxed{D}$					
	DATE-	$\boxed{\text{DATE-}} \boxed{S1} \boxed{S2} \boxed{D}$	$\begin{matrix} (S1) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix} - \begin{matrix} (S2) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix} \rightarrow \begin{matrix} (D) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix}$		4	-	7-350
	DATE-P	$\boxed{\text{DATE-P}} \boxed{S1} \boxed{S2} \boxed{D}$					
Clock data translation	SECOND	$\boxed{\text{SECOND}} \boxed{S} \boxed{D}$	$\begin{matrix} (S) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix} \rightarrow \begin{matrix} (D) \\ \text{Sec. (Lower 16 bits)} \\ \text{Sec. (Upper 16 bits)} \end{matrix}$		3	-	7-352
	SECONDP	$\boxed{\text{SECONDP}} \boxed{S} \boxed{D}$					
	HOUR	$\boxed{\text{HOUR}} \boxed{S} \boxed{D}$	$\begin{matrix} (S) \\ \text{Sec. (Lower 16 bits)} \\ \text{Sec. (Upper 16 bits)} \end{matrix} \rightarrow \begin{matrix} (D) \\ \text{Hour} \\ \text{Minute} \\ \text{Sec.} \end{matrix}$		3	-	7-354
	HOURP	$\boxed{\text{HOURP}} \boxed{S} \boxed{D}$					

Table 2.32 Character String Processing Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Date comparison	LDDT=				4	-	7-356
	ANDDT=						
	ORDT=						
	LDDT<>				4	-	
	ANDDT<>						
	ORDT<>						
	LDDT<				4	-	
	ANDDT<						
	ORDT<						
	LDDT<=				4	-	
	ANDDT<=						
	ORDT<=						
	LDDT>				4	-	
	ANDDT>						
	ORDT>						
LDDT>=				4	-		
ANDDT>=							
ORDT>=							

Table 2.32 Character String Processing Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Clock comparison	LDTM=		<p>→ Comparison operation result</p>		4	-	7-361
	ANDTM=						
	ORTM=						
	LDTM<>		<p>→ Comparison operation result</p>		4	-	
	ANDTM<>						
	ORTM<>						
	LDTM<		<p>→ Comparison operation result</p>		4	-	
	ANDTM<						
	ORTM<						
	LDTM<=		<p>→ Comparison operation result</p>		4	-	
	ANDTM<=						
	ORTM<=						
	LDTM>		<p>→ Comparison operation result</p>		4	-	
	ANDTM>						
	ORTM>						
	LDTM>=		<p>→ Comparison operation result</p>		4	-	
ANDTM>=							
ORTM>=							



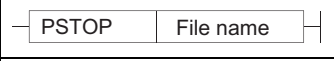
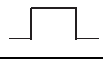



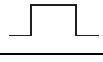
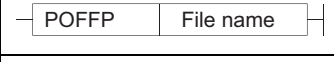
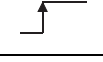
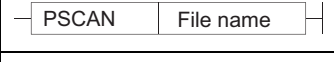
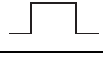


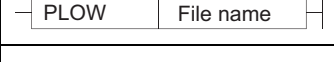

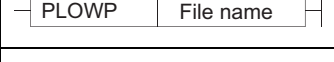

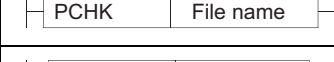


## 2.5.16 Expansion clock instruction

Table 2.33 Expansion clock instruction

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description				
Reading data of the expansion clock	S.DAT-ERD	$\boxed{\text{S.DATERD}} \boxed{D}$	·(Clock elements) →(D) +0 +1 Year +2 Month +3 Day +4 Hour +5 Minute +6 Sec. +7 Day of the week 1/1000 sec.		6	-					
	SP.DAT-ERD	$\boxed{\text{SP.DATERD}} \boxed{D}$									
Adding or subtracting data values of the expansion clock	S.DATE+	$\boxed{\text{S.DATE+}} \boxed{S1} \boxed{S2} \boxed{D}$	(S1) Hour Minute Sec. — 1/1000 sec.	+	(S2) Hour Minute Sec. — 1/1000 sec.	→	(D) Hour Minute Sec. — 1/1000 sec.		8	-	
	SP.DATE+	$\boxed{\text{SP.DATE+}} \boxed{S1} \boxed{S2} \boxed{D}$									
	S.DATE-	$\boxed{\text{S.DATE-}} \boxed{S1} \boxed{S2} \boxed{D}$	(S1) Hour Minute Sec. — 1/1000 sec.	-	(S2) Hour Minute Sec. — 1/1000 sec.	→	(D) Hour Minute Sec. — 1/1000 sec.		8	-	
	SP.DATE-	$\boxed{\text{SP.DATE-}} \boxed{S1} \boxed{S2} \boxed{D}$									

## 2.5.17 Program control instructions

Table 2.34 Program Control Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Program control instructions	PSTOP		• Places designated program in standby status.		*1	-	7-377
	PSTOPP				2 + n		
	POFF		• Turns OUT instruction coil of designated program OFF, and places program in standby status.		*1	-	7-378
	POFFP				2 + n		
	PSCAN		• Registers designated program as scan execution type.		*1	-	7-380
	PSCANP				2 + n		
	PLOW		• Registers designated program as low-speed execution type.		*1	-	7-382
	PLOWP				2 + n		
	LDPCHK		• In conduction when program of specified file name is being executed. • In non-conduction when program of specified file name is not executed.		*1	-	7-384
	ANDPCHK			2 + n			
ORPCHK		n					

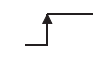
\*1:  $n$  ((number of file name characters] / 2) indicates a step. (Decimal fractions are rounded up.)

## 2.5.18 Other instructions

Table 2.35 Other Instructions

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description	
WDT reset	WDT		<ul style="list-style-type: none"> <li>Resets watchdog timer during sequence program.</li> </ul>		1	-	7-386	
	WDTP							
Timing clock	DUTY		 SM420 to SM424, SM430 to SM434		4	-	7-388	
Time check	TIMCHK		<ul style="list-style-type: none"> <li>Turns ON device specified by (D) if measured ON time of input condition is longer than preset time continuously.</li> </ul>		4	-	7-390	
Direct read/write operations in 1-byte units	ZRRDB				3	-	7-391	
	ZRRDBP							
	ZRWRB					3	-	7-393
	ZRWRBP							
	ADRSET					3	-	7-395
	ADRSETP							
Numerical key input from keyboard	KEY		<ul style="list-style-type: none"> <li>Takes in ASCII data for 8 points of input unit designated by (S), converts to hexadecimal value following device number designated by (D1), and stores.</li> </ul>		5	-	7-396	
Batch save of index register	ZPUSH		<ul style="list-style-type: none"> <li>Saves the contents of index registers to a location starting from the device designated by (D).</li> </ul>		2	-	7-400	
	ZPUSHP							
Batch recovery of index register	ZPOP		<ul style="list-style-type: none"> <li>Reads the data stored in the location starting from the device designated by (D) to index registers.</li> </ul>		2	-	7-400	
	ZPOPP							
Batch write operation to E <sup>2</sup> PROM file register	EROMWR		<ul style="list-style-type: none"> <li>Writes a batch of data to E<sup>2</sup>PROM file register.</li> </ul>		5	-	7-400	
	EROMWRP							
Reading module information	UNIRD		<ul style="list-style-type: none"> <li>Reads the module information stored in the area starting from the I/O No. designated by n by the points designated by n2, and stores it in the area starting from the device designated by (D).</li> </ul>		4	-		
	UNIRDP							

Table 2.35 Other Instructions (Continued)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Module model name read	TYPERD	$\text{---TYPERD } n \text{ D---}$	<ul style="list-style-type: none"> <li>This instruction reads the module information stored in the area starting from the I/O number specified by "n", and stores it in the area starting from the device specified by (D).</li> </ul>		3	-	
	TYPERDP	$\text{---TYPERDP } n \text{ D---}$					
Trace set	TRACE	$\text{---TRACE---}$	<ul style="list-style-type: none"> <li>Stores the trace data set with peripheral device by the number of times set when SM800, SM801 and SM802 turn on, to the sampling trace file.</li> </ul>		1	-	
Trace rset	TRACER	$\text{---TRACER---}$	<ul style="list-style-type: none"> <li>Resets the data set the TRACE instruction.</li> </ul>		1	-	
Writing data to the designated file	SP.FWRITE	$\text{---SP.FWRITE } U0 \text{ } S0 \text{ } D0 \text{ } S1 \text{ } S2 \text{ } D1 \text{---}$	<ul style="list-style-type: none"> <li>Writes data to the designated file.</li> </ul>		11	-	
Reading data from designated file	SP.FREAD	$\text{---SP.FREAD } U0 \text{ } S0 \text{ } D0 \text{ } S1 \text{ } S2 \text{ } D1 \text{---}$	<ul style="list-style-type: none"> <li>Reads data from the designated file.</li> </ul>		11	-	
Writing data to standard ROM	S.DEVST	$\text{---S.DEVST } n1 \text{ } S \text{ } n2 \text{ } D \text{---}$	<ul style="list-style-type: none"> <li>Writes data to the device data storage file in the standard ROM.</li> </ul>		9	-	
Reading data from standard ROM	S.DEVLD	$\text{---S.DEVLD } n1 \text{ } D \text{ } n2 \text{---}$	<ul style="list-style-type: none"> <li>Reads data from the device data storage file in the standard ROM.</li> </ul>		8	-	
	SP.DEVLD	$\text{---SP.DEVLD } n1 \text{ } D \text{ } n2 \text{---}$					
Loading program from memory	PLOADP	$\text{---PLOADP } S \text{ } D \text{---}$	<ul style="list-style-type: none"> <li>Transfers the program stored in a memory card or standard memory (other than drive 0) to drive 0 and places the program in standby status.</li> </ul>		3	-	
Unloading program from program memory	PUNLOADP	$\text{---PUNLOADP } S \text{ } D \text{---}$	<ul style="list-style-type: none"> <li>Deletes the standby program stored in standard memory (drive 0).</li> </ul>		3	-	
Load + Unload	PSWAPP	$\text{---PSWAPP } S1 \text{ } S2 \text{ } D \text{---}$	<ul style="list-style-type: none"> <li>Deletes standby program stored in standard memory (drive 0) designated by (S1). Then, transfers the program stored in a memory card or standard memory (other than drive 0) designated by (S2) to drive 0 and places it in standby status.</li> </ul>		4	-	
High-speed block transfer of file register	RBMOV	$\text{---RBMOV } S \text{ } D \text{ } n \text{---}$	<ul style="list-style-type: none"> <li>Transfers n points of 16-bit data from the device designated by (S) to the devices of n points starting from the one designated by (D).</li> </ul>		4	-	
	RBMOV P	$\text{---RBMOV P } S \text{ } D \text{ } n \text{---}$					

## 2.5.19 Instructions for Data Link

Table 2.36 Instructions for Data Link

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Q link instruction: Network refresh	S.ZCOM	$\overline{\text{S.ZCOM}} \text{ Jn}$	Refreshes the designated network.		5	-	8-2
	SP.ZCOM	$\overline{\text{SP.ZCOM}} \text{ Jn}$					
	S.ZCOM	$\overline{\text{S.ZCOM}} \text{ Un}$					
	SP.ZCOM	$\overline{\text{SP.ZCOM}} \text{ Un}$					
Reading routing information	S.RTREAD	$\overline{\text{S.RTREAD}} \text{ n D}$	Reads data set at routing parameters.		7	-	8-6
	SP.RTREAD	$\overline{\text{SP.RTREAD}} \text{ n D}$					
	Z.RTREAD	$\overline{\text{Z.RTREAD}} \text{ n D}$					
	ZP.RTREAD	$\overline{\text{ZP.RTREAD}} \text{ n D}$					
Registering routing information	S.RTWRITE	$\overline{\text{S.RTWRITE}} \text{ n S}$	Writes routing data to the area designated by routing parameters.		8	-	8-8
	SP.RTWRITE	$\overline{\text{SP.RTWRITE}} \text{ n S}$					
	Z.RTWRITE	$\overline{\text{Z.RTWRITE}} \text{ n S}$					
	ZP.RTWRITE	$\overline{\text{ZP.RTWRITE}} \text{ n S}$					

## 2.5.20 Multiple CPU dedicated instruction

Table 2.37 Multiple CPU dedicated instruction

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Write to host CPU shared memory	S.TO		• Writes device data of the host station to the host CPU shared memory.		5	-	9-4
	SP.TO						
	TO		• Writes device data of the host station to the host CPU shared memory.		5	-	9-7
	TOP						
	DTO		• Writes device data of the host station to the host CPU shared memory in 32-bit units.		5	-	
	DTOP						
Read from other CPU shared memory	FROM		• Reads device data from the other CPU shared memories, and stores the data in the host station.		5	-	9-12
	FROMP						
	DFRO		• Reads device data from the other CPU shared memories in 32-bit units, and stores the data in the host station.		5	-	
	DFROP						



## 2.5.21 Multiple CPU high-speed transmission dedicated instruction

Table 2.40 Multiple CPU high-speed transmission dedicated instruction

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
Writing Devices to Another CPU	D.DDWR		In multiple CPU system, data stored in a device specified by host CPU (S2) or later is stored by the number of write points specified by (D2+1) into a device specified by another CPU (n) (D1) or later		10	-	10-13
	DP.DDWR						
Reading Devices from Another CPU	D.DDRD		In multiple CPU system, data stored in a device specified by another CPU (n) (D1) or later is stored by the number of read points specified by (S1+1) into a device specified by host CPU (S2) or late		10	-	10-17
	DP.DDRD						

## 2.5.22 Redundant system instructions (For Redundant CPU)

Table 2.39 Redundant System Instructions (For Redundant CPU)

Category	Instruction Symbol	Symbol	Processing Details	Execution Condition	Number of Basic Steps	Subset	See for Description
System switching	SP.CONTSW		Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.		8	-	11-2





# 3

## CONFIGURATION OF INSTRUCTIONS

---

# 3.1 Configuration of Instructions

Most CPU module instructions consist of an instruction part and a device part.

Each part is used for the following purpose:

- Instruction part ..... indicates the function of the instruction.
- Device part ..... indicates the data that is to be used with the instruction.

The device part is classified into source data, destination data, and number of devices.

(1) Source (S)

(a) Source is the data used for operations.

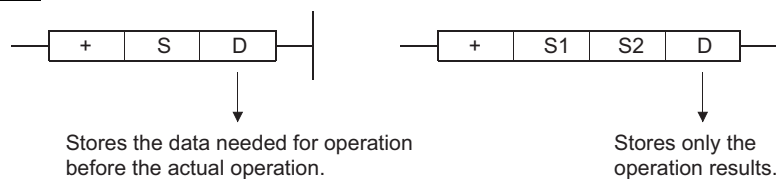
(b) The following source types are available, depending on the designated device:

- Constant ..... Designates a numeric value to be used in the operation.  
This is set when the program is created, and cannot be changed during the execution of the program.  
Constants should be indexed when used as variable data.
- Bit devices and word devices ..... Designates the device that stores the data to be used in the operation.  
Data must be stored in the designated device until the operation is executed.  
By changing the data stored in a designated device during program execution, the data to be used in the instruction can be changed.

(2) Destination (D)

(a) The destination stores the data after the operation has been conducted. However, some instructions require storing the data to be used in an operation at the destination prior to the operation execution.

**Example** An addition instruction involving BIN 16-bit data

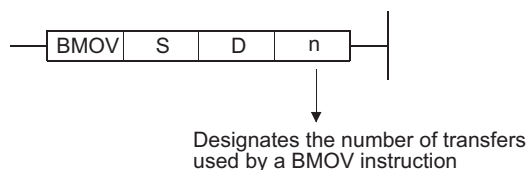


(b) A device for the data storage must always be set to the destination.

(3) Number of devices and number of transfers (n)

(a) The number of devices and number of transfers designate the numbers of devices and transfers used by instructions involving multiple devices.

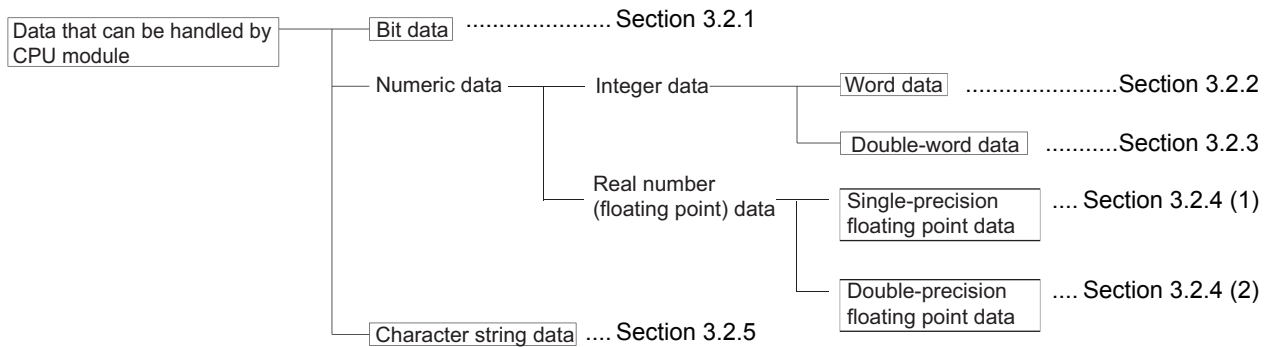
**Example** Block transfer instruction



(b) The number of devices or number of transfers can be set between 0 and 32767. However, if the number is 0, the instruction will be a no-operation instruction.

## 3.2 Designating Data

The following six types of data can be used with CPU module instructions.



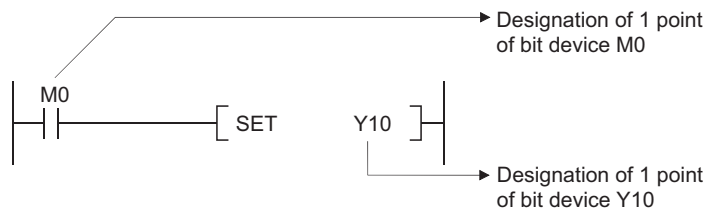
### 3.2.1 Using bit data

Bit data is data used in one-bit units, such as for contacts or coils.

"Bit devices" and "Bit designated word devices" can be used as bit data.

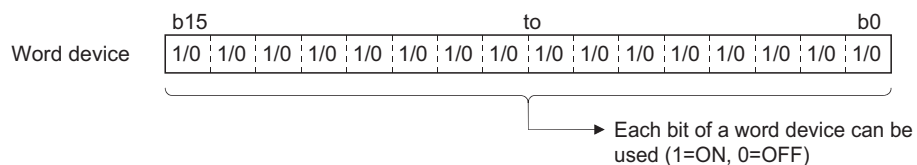
#### (1) When using bit devices

Bit devices are designated in one-point units.



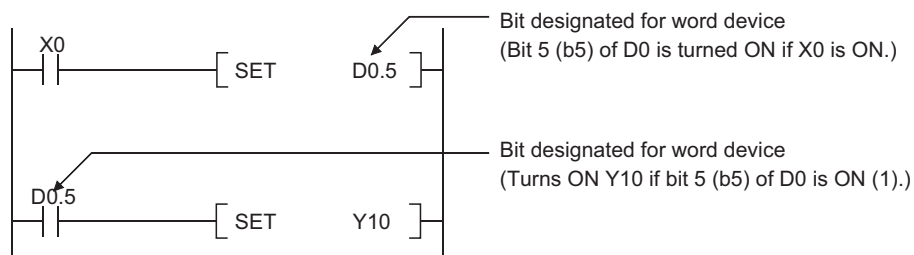
#### (2) Using word devices

(a) Word devices enable the use of a designated bit number 1/0 as bit data by the designation of that bit number.



(b) Word device bit designation is done by designating " **Word device** **Bit No.** ".  
(Designation of bit numbers is done in hexadecimal.)

For example, bit 5 (b5) of D0 is designated as D0.5, and bit 10 (b10) of D0 is designated as D0.A. However, there can be no bit designation for timers (T), retentive timers (ST), counters (C) or index register (Z). (Example Z0.0 is not available).



## 3.2.2 Using word (16 bits) data

Word data is 16-bit numeric data used by basic instructions and application instructions.

The following two types of word data can be used with CPU module:

- Decimal constants ..... K-32768 to K32767
- Hexadecimal constants ..... H0000 to HFFFF

Word devices and bit devices designated by digit can be used as word data.

For direct access input (DX) and direct access output (DY), word data cannot be designated by digit. (For details of direct access input and direct access output, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

### (1) When Using Bit Devices

- (a) Bit devices can deal with word data when digits are designated.

Digit designation of bit devices is done by designating

"Number of digits Head number of bit device". Digit designation of bit devices can be done in

4-point (4-bit) units, and designation can be made for K1 to K4. (For link direct devices, designation is done by "J Network No. \ Number of digits Head number of bit device".

When X100 to X10F are designated for Network No.2, it is done by J2\K4X100). For example, if X0 is designated for digit designation, the following points would be designated:

- K1X0 ..... The 4 points X0 to X3 are designated.
- K2X0 ..... The 8 points X0 to X7 are designated.
- K3X0 ..... The 12 points X0 to XB are designated.
- K4X0 ..... The 16 points X0 to XF are designated.

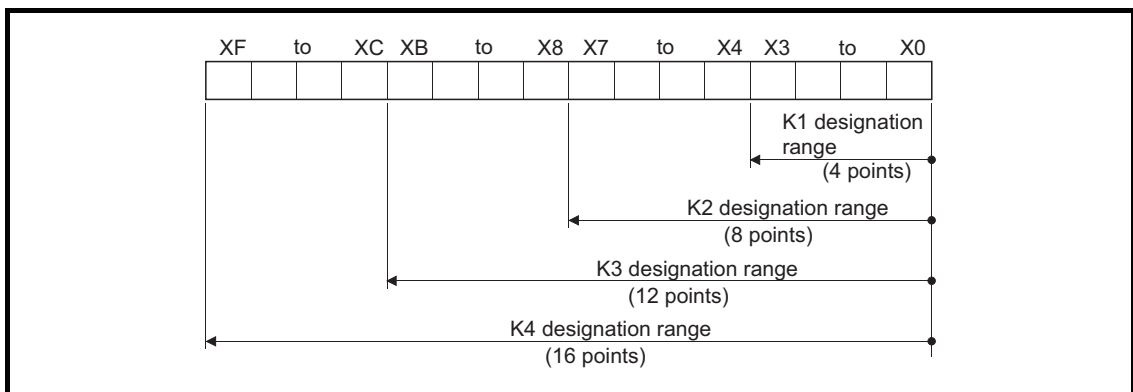


Fig 3.1 Digit Designation Setting Range for 16-Bit Instruction

- (b) In cases where digit designation has been made at the source (S), the numeric values shown in Table 3.1 are those which can be dealt with as source data.

Table 3.1 List of Numeric Values that Can Be Dealt with as Digit Designation

Number of Digits Designated	With 16-Bit Instruction
K1 (4 points)	0 to 15
K2 (8 points)	0 to 255
K3 (12 points)	0 to 4095
K4 (16 points)	-32768 to 32767

- (c) When destination (D) data is a word device  
The word device for the destination becomes 0 following the bit designated by digit designation at the source.

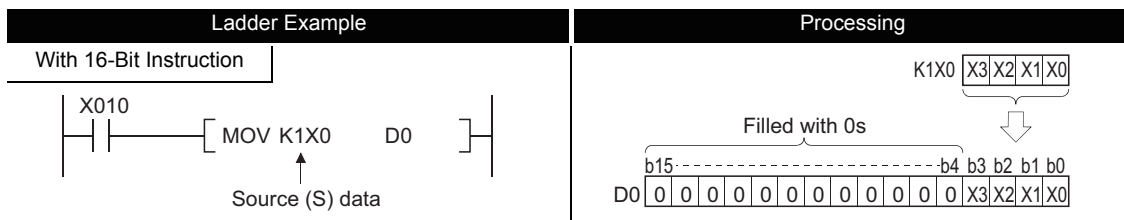


Fig 3.2 Ladder Example and Processing Conducted

- (d) In cases where digit designation is made at the destination (D), the number of points designated are used as the destination.  
Bit devices below the number of points designated as digits do not change.

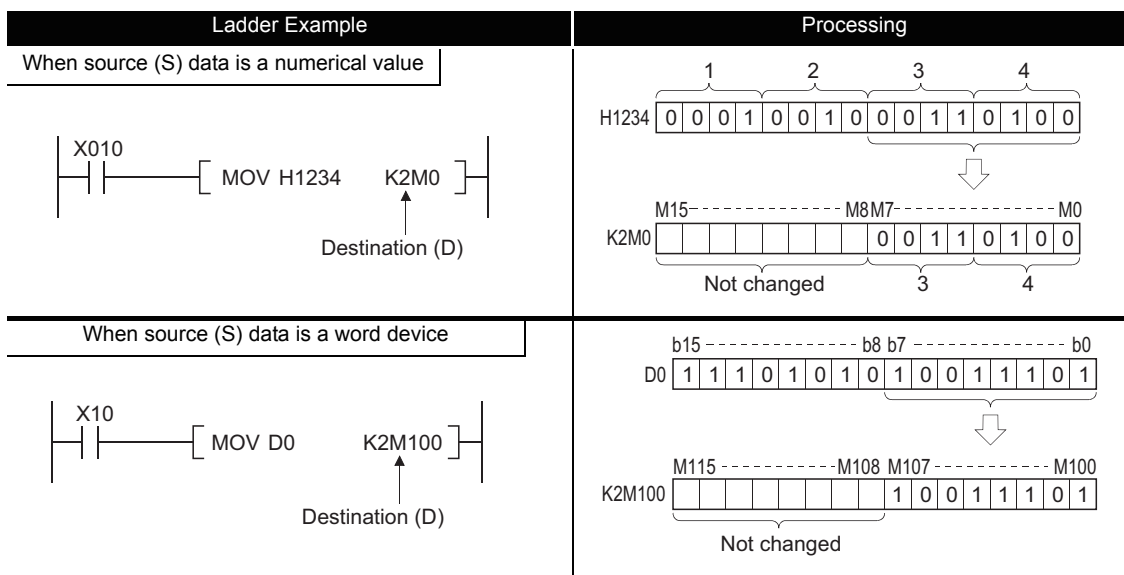
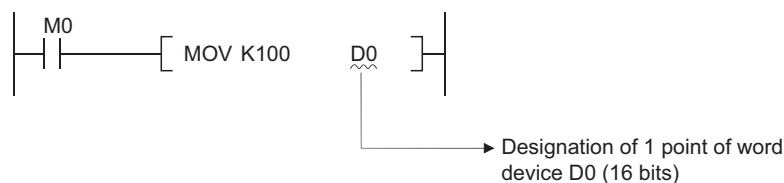


Fig 3.3 Ladder Example and Processing Conducted

(2) Using word devices

Word devices are designated in 1-point (16 bits) units.



**POINT**

1. When digit designation processing is conducted, a random value can be used for the bit device initial device number.
2. Digit designation cannot be made for the direct access I/O (DX, DY).

### 3.2.3 Using double word data (32 bits)

Double word data is 32-bit numerical data used by basic instructions and application instructions.

The two types of double word data that can be dealt with by CPU module are as follows:

- Decimal constants ..... K-2147483648 to K2147483647
- Hexadecimal constants ..... H00000000 to HFFFFFFF

Word devices and bit devices designated by digit designation can be used as double word data.

For direct access input (DX) and direct access output (DY), designation of double word data is not possible by digit designation.

#### (1) When Using Bit Devices

- (a) Digit designation can be used to enable a bit device to deal with double word data. Digit designation of bit devices is done by designating

" [Number of digits] [Head number of bit device] ". For link direct devices, designation is done by

"J [Network No.] \ [Number of digits] [Head number of bit device] ". When X100 to X11F are designated for Network No.2, it is done by J2\K8X100. Digit designation of bit devices can be done in 4-point (4-bit) units, and designation can be made for K1 to K8. For example, if X0 is designated for digit designation, the following points would be designated:

- K1X0 ..... The 4 points X0 to X3 are designated.
- K2X0 ..... The 8 points X0 to X7 are designated.
- K3X0 ..... The 12 points X0 to XB are designated.
- K4X0 ..... The 16 points X0 to XF are designated.
- K5X0 ..... The 20 points X0 to X13 are designated.
- K6X0 ..... The 24 points X0 to X17 are designated.
- K7X0 ..... The 28 points X0 to X1B are designated.
- K8X0 ..... The 32 points X0 to X1F are designated.

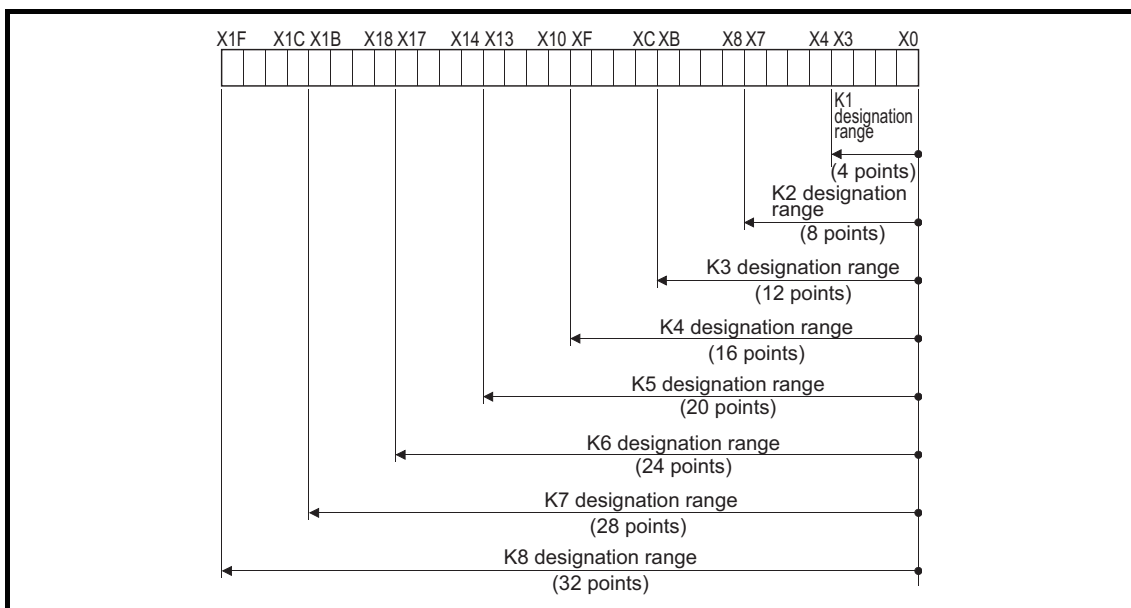


Fig 3.4 Digit Designation Setting Range for 32-Bit Instructions

- (b) In cases where digit designation has been made at the source (S), the numeric values shown in Table 3.2 are those which can be dealt with as source data.

Table 3.2 List of Numeric Values that Can Be Dealt with as Digit Designation

Number of Digits Designated	With 32 Bit Instructions	Number of Digits Designated	With 32 Bit Instructions
K1 (4 points)	0 to 15	K5 (20 points)	0 to 1048575
K2 (8 points)	0 to 255	K6 (24 points)	0 to 16777215
K3 (12 points)	0 to 4095	K7 (28 points)	0 to 268435455
K4 (16 points)	0 to 65535	K8 (32 points)	-2147483648 to 2147483647

- (c) When destination (D) data is a word device  
 The word device for the destination becomes 0 following the bit designated by digit designation at the source.

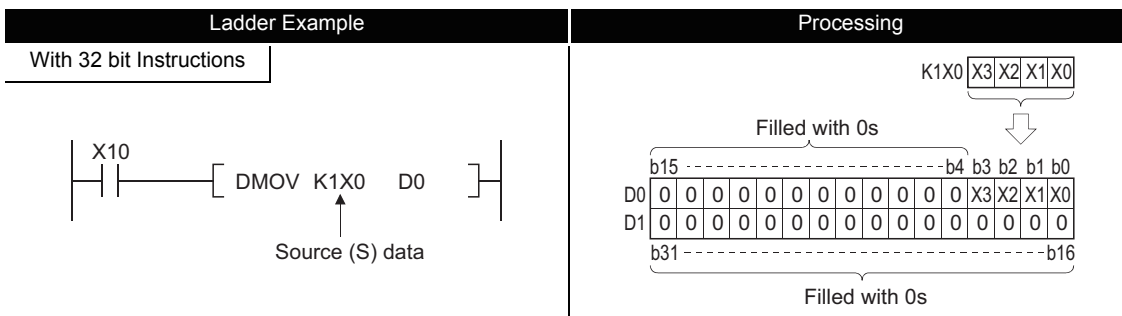


Fig 3.5 Ladder Example and Processing Conducted

- (d) In cases where digit designation is made at the destination (D), the number of points designated are used as the destination. Bit devices below the number of points designated as digits do not change.

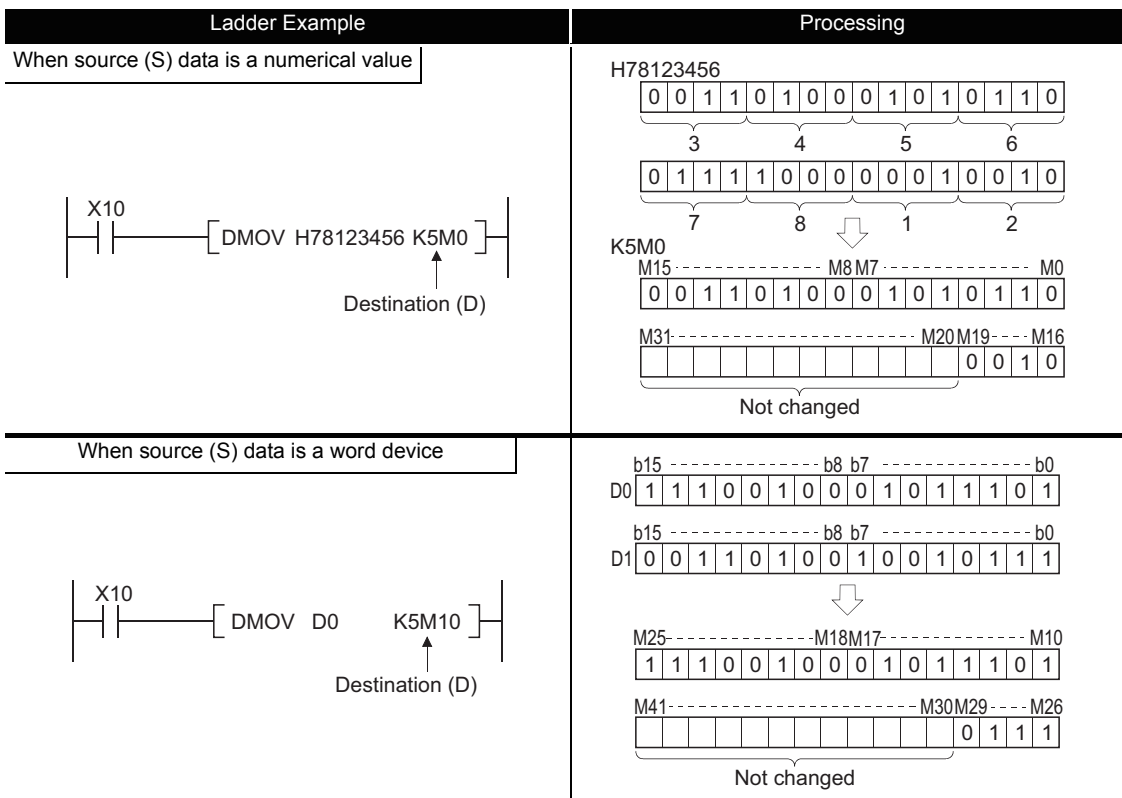


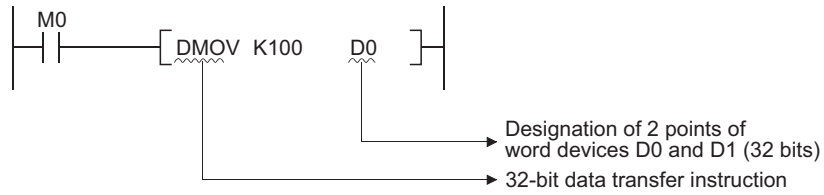
Fig 3.6 Ladder Example and Processing Conducted

**POINT**

1. When digit designation processing is conducted, a random value can be used for the bit device initial device number.
2. Digit designation cannot be made for the direct access I/O (DX, DY).

(2) Using word devices

A word device designates devices used by the lower 16 bits of data. A 32-bit instruction uses (designation device number) and (designation device number + 1).



### 3.2.4 Using real number data

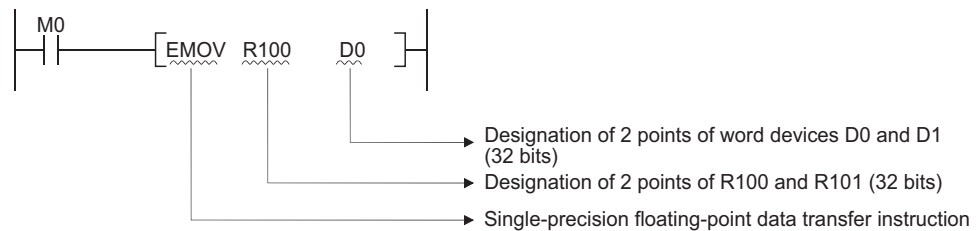
Real number data is floating decimal point data used with basic instructions and application instructions.

Only word devices are capable of storing real number data.

(1) Single-precision floating-point data

Instructions which deal with single-precision floating-point data designate devices which are used for the lower 16 bits of data.

Single-precision floating-point data are stored in the 32 bits which make up (designated device number) and (designated device number + 1).



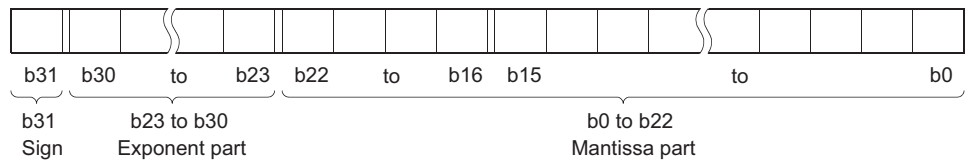
**Remark**

In sequence programs, floating decimal point data are designated by E□□.

Single-precision floating-point data uses two word devices and is expressed in the following manner:

$$[\text{Sign}] 1. [\text{Mantissa part}] \times 2^{[\text{Exponent part}]}$$

The bit configuration and meaning of the internal representation of single-precision floating-point data is as follows:



- **Sign** The sign is represented at b31.  
0: Positive  
1: Negative
- **Exponent part** The n of 2<sup>n</sup> is represented from b23 to b30.  
Depending on the BIN value of b23 to b30, the value of n is as follows:

b23 to b30	FFH	FEH	FDH		81	80	7FH	7EH		02	01	00
n	Not used	127	126		2	1	0	-1		-125	-126	Not used

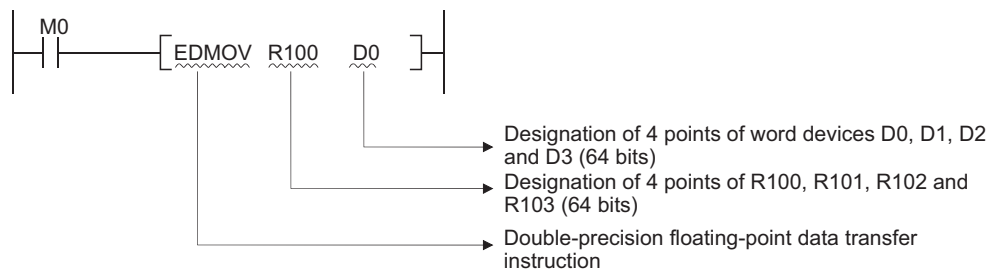


- Variable part The 23 bits from b0 to b22, represents the XXXXXX... at binary 1.XXXXXX....

(2) Double-precision floating-point data

Instructions which deal with double-precision floating-point data designate devices which are used for the lower 16 bits of data.

Double-precision floating-point data are stored in the 64 bits which make up (designated device number) to (designated device number + 3).



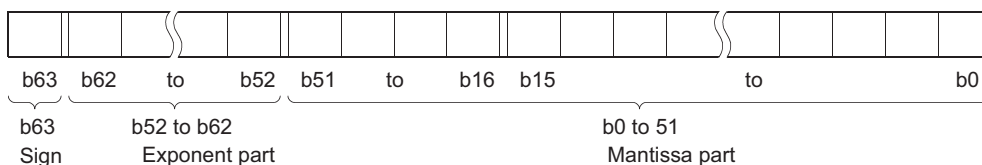
**Remark**

In sequence programs, floating decimal point data are designated by E□□.

Double-precision floating-point data uses four word devices and is expressed in the following manner:

$$[\text{Sign}] 1. [\text{Mantissa part}] \times 2^{[\text{Exponent part}]}$$

The bit configuration and meaning of the internal representation of double-precision floating-point data is as follows:



- Sign The sign is represented at b63.  
0: Positive  
1: Negative

- Exponent part The n of 2<sup>n</sup> is represented from b52 to b62. Depending on the BIN value of b52 to b62, the value of n is as follows:

b52 to b62	7FFH	7FEH	7FDH		400H	3FFH	3FEH	3FDH	3FCH		02H	01H	00H
n	Not used	1023	1022		2	1	0	-1	-2		-1021	-1022	Not used

- Variable part The 52 bits from b0 to b51, represents the XXXXXX... at binary 1.XXXXXX....

## POINT

1. The CPU module floating decimal point data can be monitored using the monitoring function of a peripheral device.
2. When floating-point data is used to express 0, all data in the following range are turned to 0.
  - (a) Single-precision floating-point data: b0 to b31
  - (b) Double-precision floating-point data: b0 to b63
3. The setting range of floating decimal point data is as follows. \*1
  - (a) Single-precision floating-point data  
 $-2^{128} < \text{Device data} \leq -2^{126}, 0, 2^{126} \leq \text{Device data} < 2^{128}$
  - (b) Double-precision floating-point data  
 $-2^{1024} < \text{Device data} \leq -2^{1022}, 0, 2^{1022} \leq \text{Device data} < 2^{1024}$
4. Do not specify  $-0$  in floating-point data (when only the most significant bit of the floating-point real number is 1). (An operation error will occur if floating-point operation is performed with  $-0$ .) The CPU module that performs the internal operation of floating-point operation with double precision does not result in operation error since it performs floating-point operation after converting  $-0$  into 0 in the CPU module when  $-0$  is specified. The CPU module that performs the internal operation of floating-point operation with single precision results in operation error since it gives priority to the processing speed and uses  $-0$  in floating-point operation without conversion when  $-0$  is specified.
  - (a) The following CPU modules will not result in operation error when  $-0$  is specified.
    - High Performance model QCPU where internal operation is set to double precision \*2  
(The internal operation of floating-point operation defaults to double precision.)
  - (b) The following CPU modules will result in operation error when  $-0$  is specified.
    - Basic model QCPU \*3
    - High Performance model QCPU where internal operation is set to single precision \*2
    - Process CPU
    - Redundant CPU
    - Universal model QCPU

---

\*1: For operations when a real number is out of range and operations when an invalid value is input, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

\*2: Switch between single precision and double precision of the internal operation of floating-point operation in the PLC system of the PLC parameter dialog box. For the single precision and double precision of floating-point operation, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

\*3: The Basic model QCPU can perform floating-point operation if its first five digits of serial No. are "04122 or later".

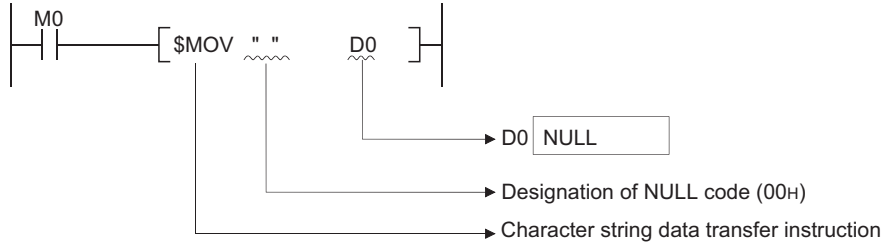
## 3.2.5 Using character string data

Character string data is character data used by basic instructions and application instructions.

The target ranges from the designated character to the NULL code (00H) that indicates the end of the character string.

### (1) When designated character is the NULL code

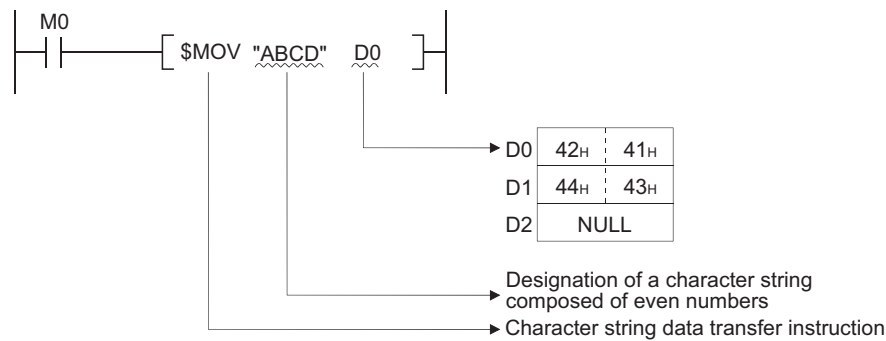
One word is used to store the NULL code.



### (2) When character string is even

Uses (number of characters/2 + 1) words, and stores character string and NULL code.

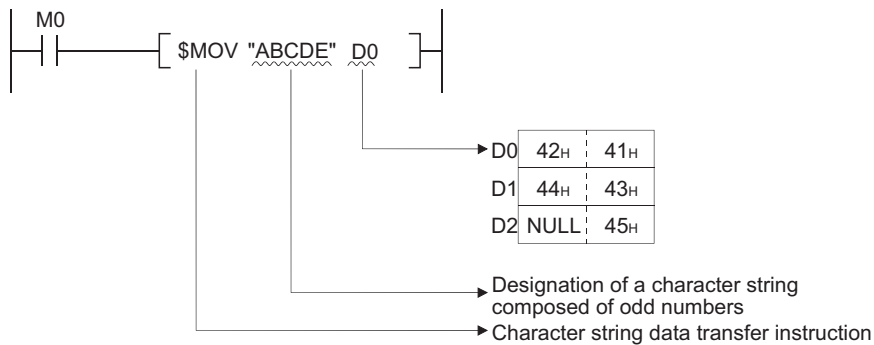
For example, if "ABCD" is transferred to D0, the character string ABCD is stored at D0 and D1, and the NULL code is stored at D2. (The NULL code is stored as the last one word.)



### (3) When number of characters is odd

Uses (number of characters/2) words (rounds up decimal fractions) and stores the character string and NULL code.

For example, if "ABCDE" is transferred to devices starting from D0, the character string (ABCDE) and the NULL code are stored from D0 to D2. (The NULL code is stored into the upper 8 bits of the last one word.)



## 3.3 Indexing

### (1) Overview of indexing

#### (a) Indexing is an indirect setting made by using an index register.

When an Indexing is used in a sequence program, the device to be used will become the device number specified directly plus the contents of the index register.

For example, if D2Z2 has been specified, the specified device is calculated as follows:  $D(2+3) = D5$  and the content of Z2 is 3 become the specified device.

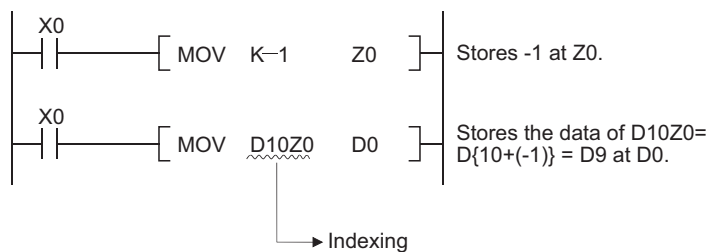
#### (b) Indexing with 16-bit index registers and indexing with 32-bit index registers are possible only for Universal model QCPU.

### (2) Indexing with 16-bit index registers

#### (a) Example of indexing

Each index register can be set between  $-32768$  and  $32767$ .

Indexing is performed in the way shown below:



#### (b) Devices to which indexing can be used

With the exception of the restrictions noted below, Indexing can be used with devices used with contacts, coils, basic instructions, and application instructions.

##### 1) Devices to which indexing can not be used

Device	Meaning
K, H	32-bit constant
E	Floating decimal point data
\$	Character string data
[ ]	Bit designated for word device
FX, FY, FD	Function devices
P	Pointers used as labels
I	Interrupt pointers used as labels
Z	Index register
S	Step relay
TR	SFC transfer devices*1
BL	SFC block devices*1

\*1: SFC transfer devices and SFC block devices are devices for SFC use.

Refer to the manual below for how to use these devices.

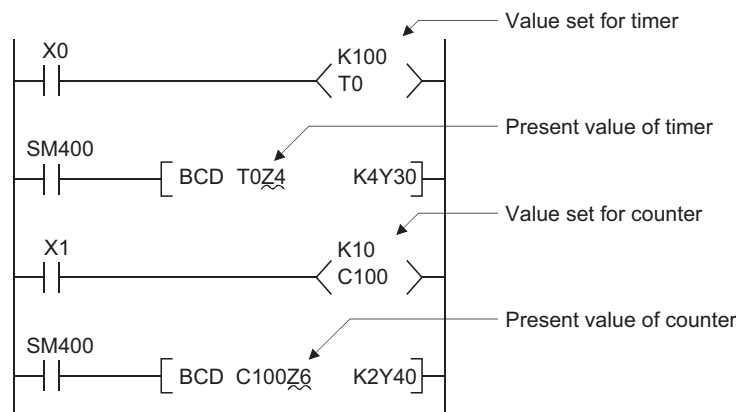
- QCPU (Q mode)/QnACPU Programming Manual (SFC)

2) Devices with limits for use with index registers

Device	Meaning	Application Example
T	• Only Z0 and Z1 can be used for timer contacts and coils.	
C	• Only Z0 and Z1 can be used for counter contacts and coils.	

**Remark**

For timer and counter present values, there are no limits on index register numbers used.



(c) A case where Indexing has been performed, and the actual process device, would be as follows:

(When Z0 = 20 and Z1 = -5)

Ladder Example	Actual Process Device
	<p>Description</p> <p><math>K2X50Z0 \dots\dots K2X(50 + 14) = K2X64</math></p> <p>Converts K20 into a hexadecimal number.</p> <p><math>K1M38Z1 \dots\dots K1M(38 - 5) = K1M33</math></p>
	<p>Description</p> <p><math>D0Z0 \dots\dots D(0 + 20) = D20</math></p> <p><math>K3Y12FZ1 \dots\dots K3Y(12F - 5) = K3Y12A</math></p> <p>Hexadecimal number</p>

Fig. 3.7 Ladder Example and Actual Process Device

(3) Indexing with 32-bit (only Universal model QCPU)

A method of specifying index registers in indexing with 32-bit can be selected from the following two methods.

- Specifying the index registers' range used for indexing with 32-bit.

- Specifying the 32-bit indexing using “ZZ” specification.

### POINT

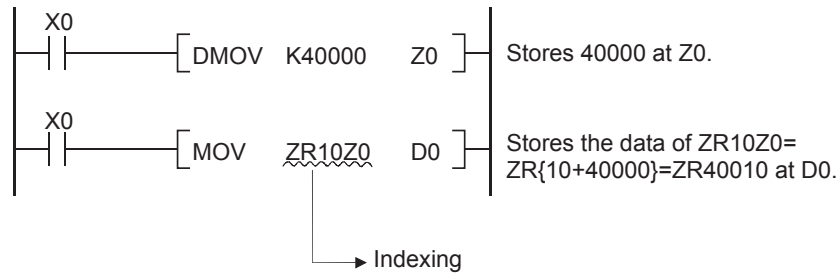
The 32-bit indexing with “ZZ” specification is available only for the following CPU modules that the version of GX Developer is 8.68W or later.

- The first five digits of the serial No. for QnU(D)(H)CPU is “10042” or higher.
- QnUDE(H)CPU

(a) Example of specifying the range of index registers for use of 32-bit indexing.

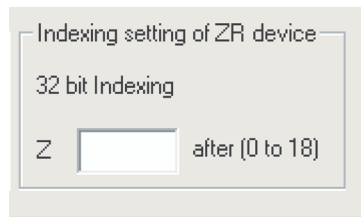
1) Each index register can be set between -2147483648 and 2147483647.

An example of indexing is shown below.

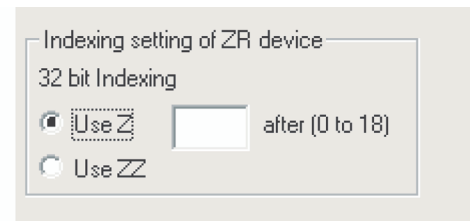


2) Specification method

For indexing with a 32-bit index register, specify the head number of an index register to be used on the Device tab of the Q parameter setting screen on GX Developer.



GX Developer 8.68R or earlier



GX Developer 8.68W or later

Fig. 3.8 Setting windows for ZR device indexing setting parameter

### POINT

When the head number of the index register used is changed on the Device tab of the Q parameter setting screen, do not change the parameters only or do not write only the parameters into the programmable controller. Be sure to write the parameter into the programmable controller with the program.

When the parameter is forced to be written into the programmable controller, an error of CAN'T EXE. PRG. occurs. (Error code: 2500)

3) Device that indexing can be used

Indexing can be used only for the device shown below.

Device	Meaning
ZR	Serial number access format file register
D	Extended data register (D)
W	Extended link register (W)

4) Usable range of index registers

The following table shows the usable range of index registers for indexing with 32-bit index registers.

For indexing with 32-bit index registers, the specified index register (Zn) and the next index register of the specified register (Zn+1) are used. Be sure not to overlap index registers to be used.

Setting Value	Index Registers to be Used	Setting Value	Index Registers to be Used
Z0	Z0, Z1	Z10	Z10, Z11
Z1	Z1, Z2	Z11	Z11, Z12
Z2	Z2, Z3	Z12	Z12, Z13
Z3	Z3, Z4	Z13	Z13, Z14
Z4	Z4, Z5	Z14	Z14, Z15
Z5	Z5, Z6	Z15	Z15, Z16
Z6	Z6, Z7	Z16	Z16, Z17
Z7	Z7, Z8	Z17	Z17, Z18
Z8	Z8, Z9	Z18	Z18, Z19
Z9	Z9, Z10	Z19	Cannot be specified

5) An example of indexing and the actual process device are as follows.

(When Z0 (32-bit) = 100000 and Z2 (16-bit) = -20)

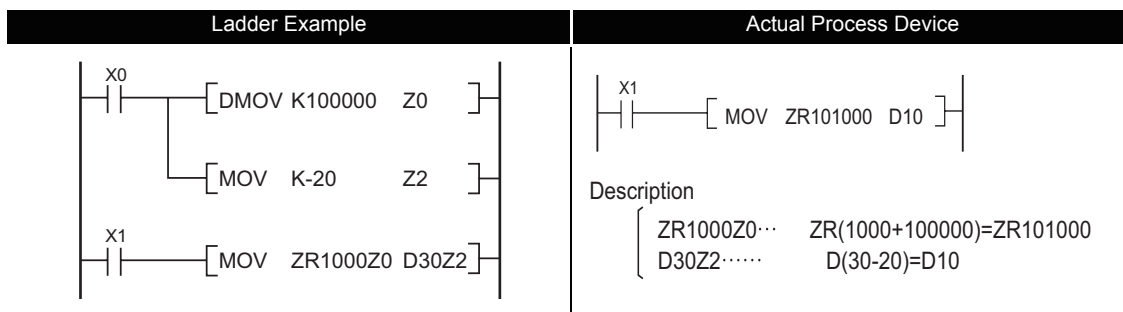
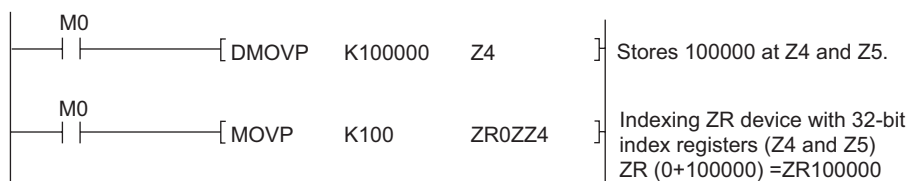


Fig. 3.9 Ladder Example and Actual Process Device

(b) Example of specifying 32-bit indexing with “ZZ” specification.

- 1) One index register can specify 32-bit indexing by using “ZZ” specification such as “ZR0ZZ4”.

The 32-bit indexing with “ZZ” specification is as follows.



- 2) Specification method

To perform 32-bit indexing by using “ZZ” specification, select “Use of ZZ” in “Indexing Setting for ZR Device” in PC parameter for GX Developer.

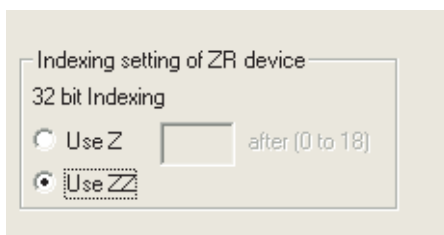


Fig. 3.10 Setting window for indexing setting parameter for ZR device

- 3) Device that indexing can be used

The following device is available for indexing.

Device	Meaning
ZR	Serial number access format file register
D	Extended data register (D)
W	Extended link register (W)

- 4) Usable range of index registers

The following table shows the usable range of index registers in 32-bit indexing used “ZZ” specification.

The 32-bit indexing with “ZZ” specification is specified as the format ZRmZZn. Specifying ZRmZZn enables Zn and Zn+1 of 32-bit values to index the device number, ZRm,

“ZZ” specification*1	Index Registers Used	“ZZ” specification*1	Index Registers Used
<input type="checkbox"/> ZZ0	Z0, Z1	<input type="checkbox"/> ZZ10	Z10, Z11
<input type="checkbox"/> ZZ1	Z1, Z2	<input type="checkbox"/> ZZ11	Z11, Z12
<input type="checkbox"/> ZZ2	Z2, Z3	<input type="checkbox"/> ZZ12	Z12, Z13
<input type="checkbox"/> ZZ3	Z3, Z4	<input type="checkbox"/> ZZ13	Z13, Z14
<input type="checkbox"/> ZZ4	Z4, Z5	<input type="checkbox"/> ZZ14	Z14, Z15
<input type="checkbox"/> ZZ5	Z5, Z6	<input type="checkbox"/> ZZ15	Z15, Z16
<input type="checkbox"/> ZZ6	Z6, Z7	<input type="checkbox"/> ZZ16	Z16, Z17
<input type="checkbox"/> ZZ7	Z7, Z8	<input type="checkbox"/> ZZ17	Z17, Z18
<input type="checkbox"/> ZZ8	Z8, Z9	<input type="checkbox"/> ZZ18	Z18, Z19
<input type="checkbox"/> ZZ9	Z9, Z10	<input type="checkbox"/> ZZ19	Not available

\*1: refers to device name (ZR) for indexing target.



5) The 32-bit indexing used “ZZ” specification and the actual processing device are as follows.

$$(Z0 \text{ (32-bit)} = 100000.Z2 \text{ (16-bit)} = -20)$$

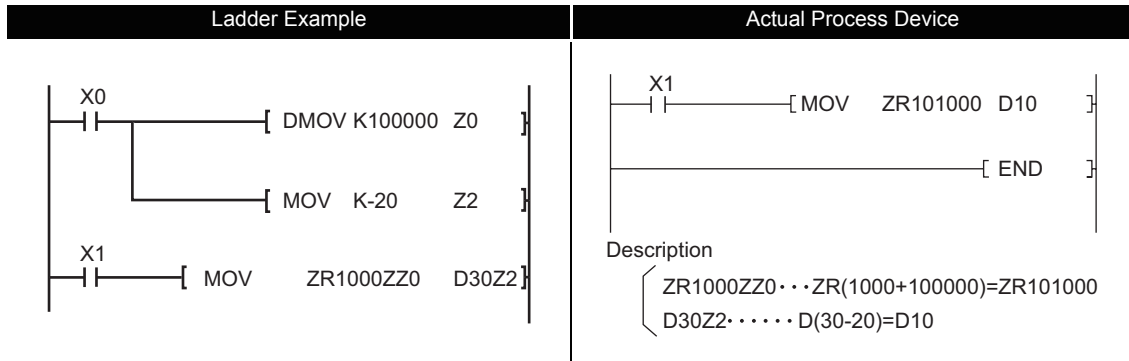


Fig.3.10 Ladder Example and Actual Process Device

6) Available functions for “ZZ” specification

The 32-bit indexing specification with “ZZ” specification applies in the following functions in GX Developer.

No.	Function Name and Description
1	Specifying devices in program instruction
2	Monitoring device registrations
3	Testing devices execution type
4	Testing devices with conditions
5	Setting monitor conditions
6	Tracing sampling (Trace point(specifying devices), Trace target device)

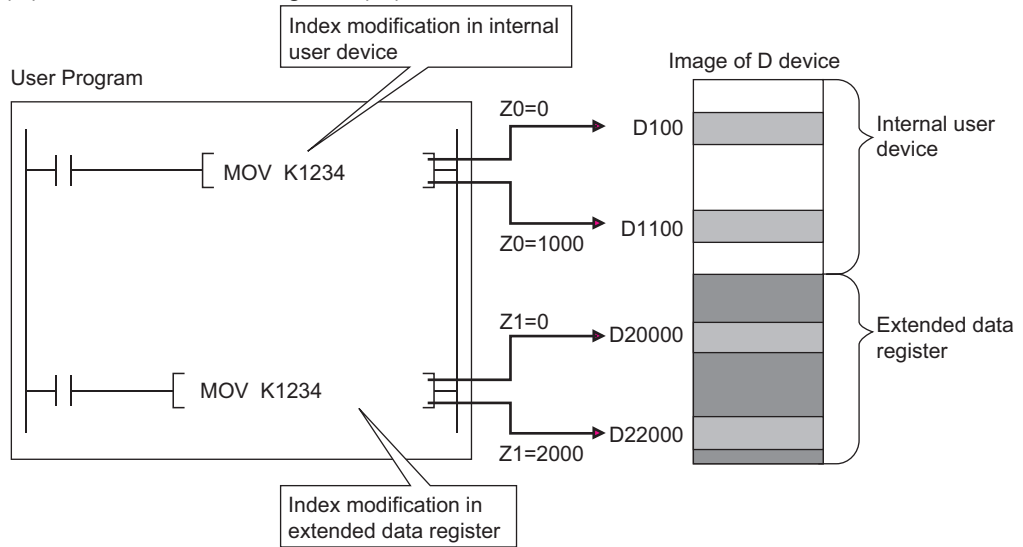
**POINT**

Single ZZn cannot be used as a device like “DMOV K100000 ZZ0”. When setting values of index registers to specify 32-bit indexing with “ZZ” specification, set the value of Zn (Z0~Z19).

Single ZZAn cannot be input to each function in GX Developer.

(4) Index modification using extended data register (D) and extended link register (W)  
(Universal model QCPU(except Q00UJCPU))

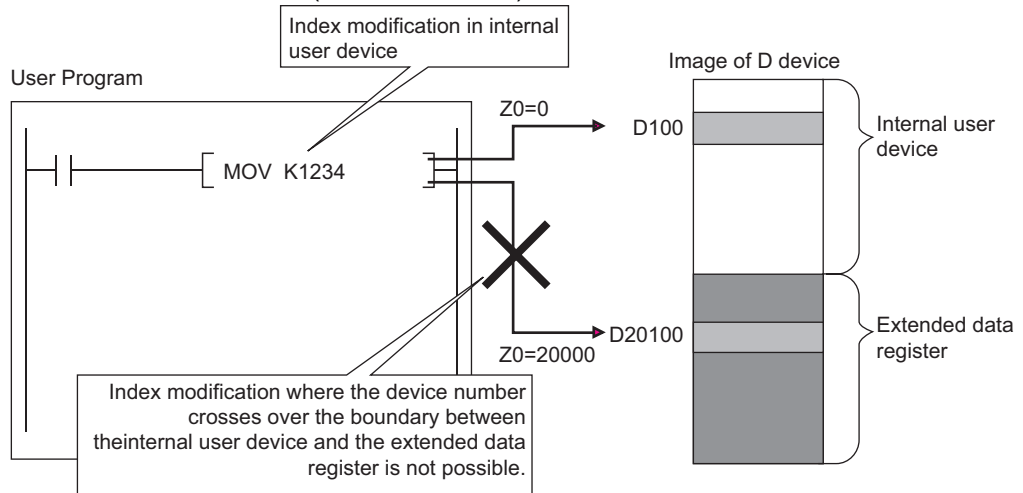
Like index modification using data register (D) and link register (W) of internal user device, a device can be specified by index modification within the range of the extended data register (D) and extended link register (W).



1) Index modification where the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W)

The specification of index modification where the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W) cannot be made.

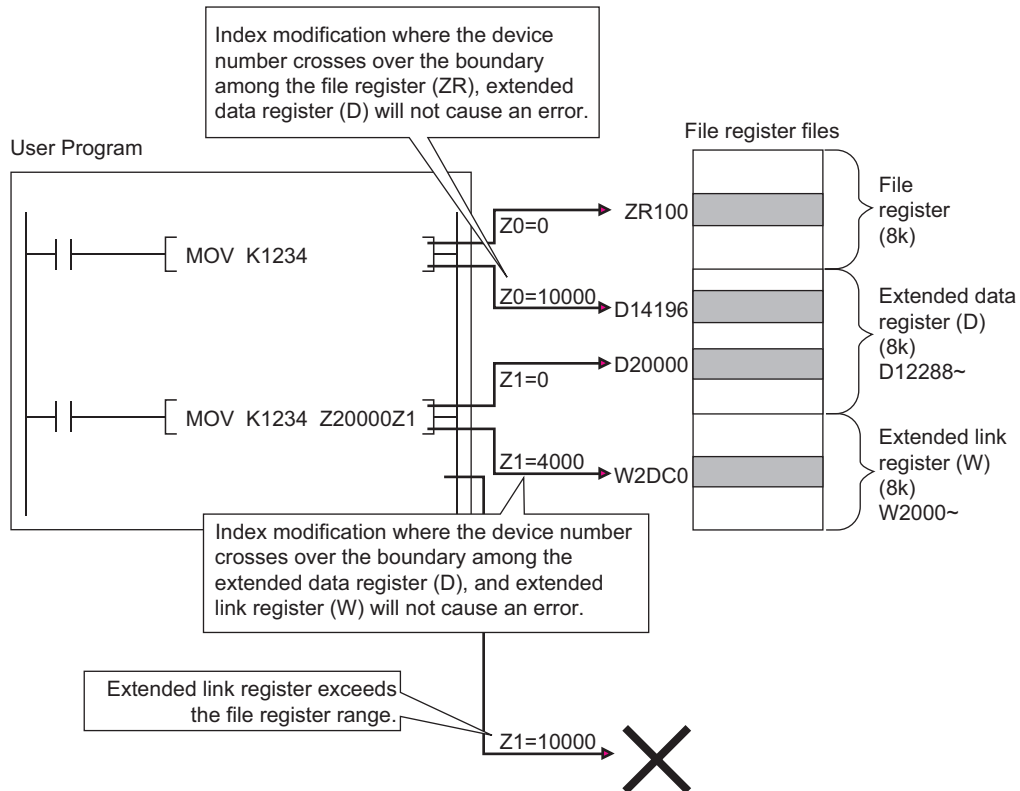
If doing so, an error occurs when the device range check is enabled at index modification (error code: 4101).



2) Index modification where the device number crosses over the boundary among the file register (ZR), extended data register (D), and extended link register (W)

Index modification where the device number crosses over the boundary among the file register (ZR), extended data register (D), and extended link register (W) will not cause an error.

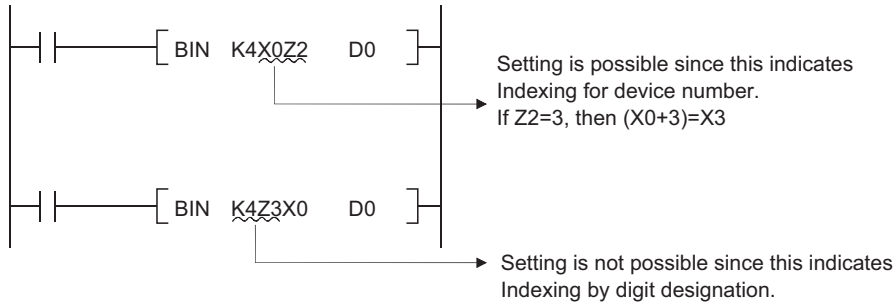
However, an error occurs if the index modification result of file register (ZR), extended data register (D), and extended link register exceeds the file register range (error code: 4101).



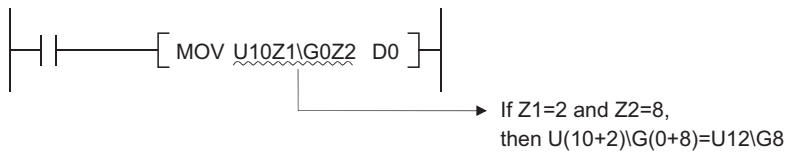
(5) Other index modifications

(a) Bit data

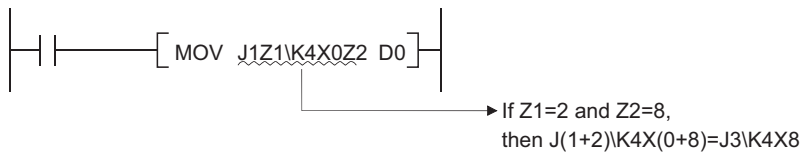
Device numbers can be index modified when performing digit designation. However, Indexing is not possible by digit designation.



(b) Both I/O numbers and buffer memory number can be performed indexing with intelligent function module devices\*1.

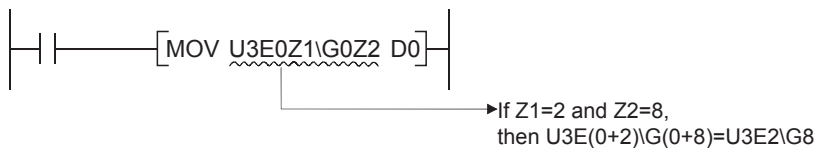


(c) Both network numbers and device numbers can be performed indexing with link direct devices\*1.



\*1: For the intelligent function module device, link direct devices, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)

(d) When indexing is used for multiple CPU shared devices\*2, indexing for the head I/O numbers of CPU modules and indexing for the CPU shared memory address are automatically executed.



\*2: For the multiple CPU shared device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)

- (e) Index modification using extended data register (D) and extended link register (W) by 32 bits (Universal model QCPU(except Q00UJCPU))  
 Like index modification using file register (ZR), index modification using extended data register (D) and extended link register (W) by 32 bits can be performed by the following two methods.
  - Specifying the index registers' range used for indexing with 32-bit.
  - Specifying the 32-bit indexing using "ZZ" specification.

**POINT**

The 32-bit indexing with "ZZ" specification is available only for the following CPU modules that the version of GX Developer is 8.68W or later.

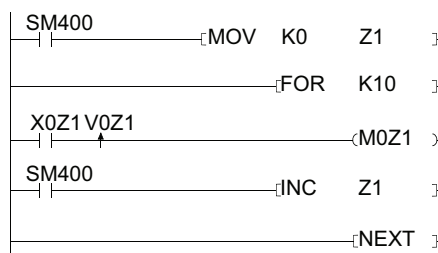
- The first five digits of the serial No. for QnU(D)(H)CPU is "10042" or higher. (except Q00UJCPU)
- QnUDE(H)CPU

(6) Cautions

- (a) Performing indexing between the FOR and NEXT instructions  
 Pulses can be output between the FOR and NEXT instructions by use of the edge relay (V). However, pulse output using the PLS/PLF/pulse (□ P) instruction is not allowed.

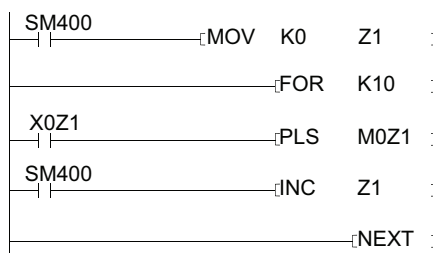
[When edge relay is used]

(M0Z1 provides normal pulse output.)



[When edge relay is not used]

(M0Z1 does not provide normal pulse output.)



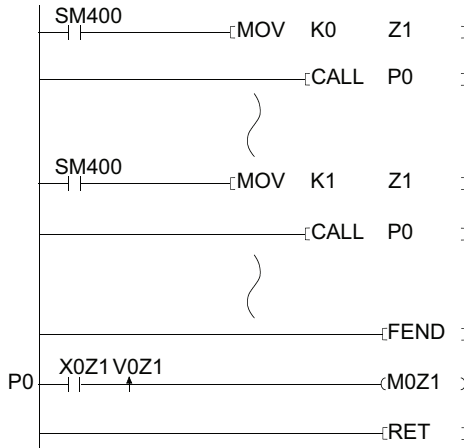
**Remark**

The ON/OFF data of X0Z1 is stored by the edge relay V0Z1.  
 For example, the ON/OFF data of X0 is stored by V0, and that of X1 by V1.

- (b) Performing indexing with the CALL instruction  
Pulses can be output with the CALL instruction by use of the edge relay (V). However, pulse output using the PLS/PLF/pulse (□ P) instruction is not allowed.

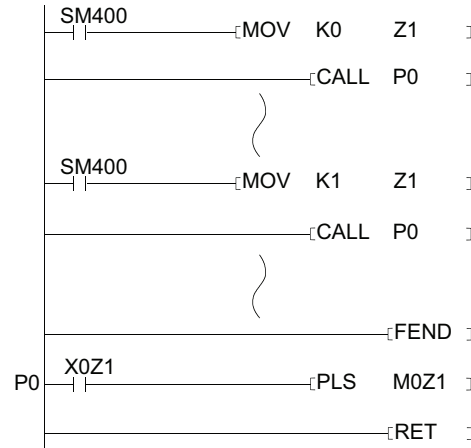
[When edge relay is used]

(M0Z1 provides normal pulse output.)



[When edge relay is not used]

(M0Z1 does not provide normal pulse output.)



- (c) Device range check during indexing

1) CPUs other than Universal model QCPU

Device range checks are not conducted during indexing.

Therefore, when the data after index modification exceed the user specified device range, the data is written to another device without causing an error. (Note, however, that when the data after index modification is written to the device for system use exceeding the user specified device range, an error occurs. (Error code: 1103))

Take extra precaution when using indexing in programming.

2) Universal model QCPU

The device range is checked for indexing.

With changing the settings of the PLC parameter on GX Developer, the device range is not checked.

- (d) Changing indexing with 16-bit index register for indexing with 32-bit index register

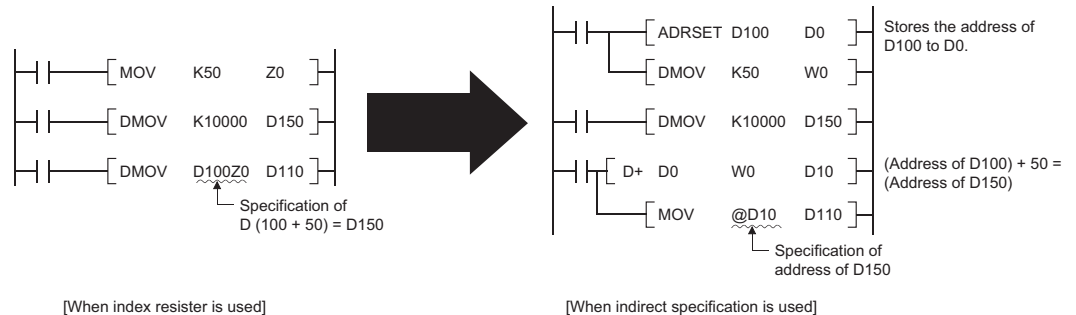
For changing indexing with 16-bit index register for indexing with 32-bit index register, check if the program has enough spaces for indexing.

For indexing with 32-bit index registers, the specified index register (Zn) and the next index register of the specified register (Zn+1) are used. Be sure not to overlap index registers to be used.

# 3.4 Indirect Specification

## (1) Indirect Specification

- (a) Indirect specification is a method that specifies address of the device to be used in a sequence program using two word devices (two points of word device). Use indirect specification as index modification when the index register is insufficient.



- (b) Specify the device to be used for specifying the address as "@ + (word device number)". For example, when @D100 is specified, the device address will be the contents of D101 and D100.
- (c) The address of the device specified indirectly can be confirmed with the ADRSET instruction.  
For the ADRSET instruction, refer to Section 7.18.6.

## (2) Indirect specification available devices

Table 3.3 shows that the CPU module devices can be specified indirectly.

Table 3.3 List of Indirect Specification Available Devices

Device Type		Availability of Indirect Specification	Example of Indirect Specification
Internal user device	Bit device *1	N/A	_____
	Word device *1	Available	• @D100 • @D100Z2 *2
Link direct device	Bit device *1	N/A	_____
	Word device *1	Available*3	• @J1W10 • @J1Z1W10Z2 *2
Intelligent function module device		Available*3	• @U10G0 • @U10Z1G0Z2 *2
Index register		N/A	_____
File register		Available	• @R0, @ZR20000 • @R0Z1, @ZR20000Z1 *2
Extended data register (D)		Available	• @D1000 • @W1000
Extended link register (W)			
Nesting		N/A	_____
Pointer			_____
Constants			_____
Other	SFC block device		_____
	SFC transition device		_____
	Network No. specification device	_____	
	I/O No. specification device	_____	

\*1: For the device names, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)

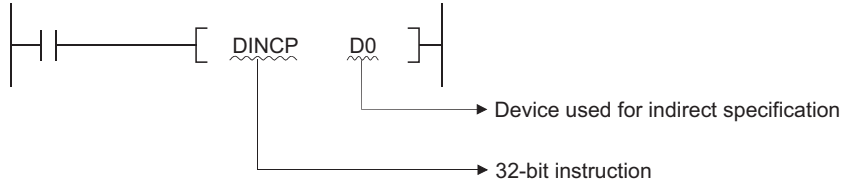
\*2: Indicates when index modification by an index register is performed.

\*3: Indirect specification is possible, but the address can not be written with the ADRSET instruction.

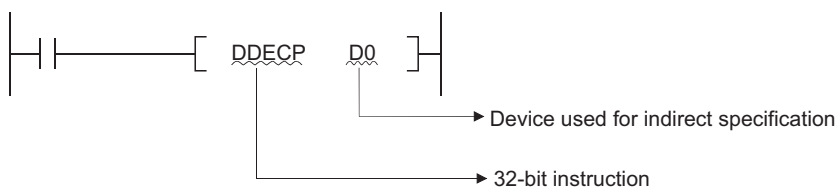
(3) Precautions

- (a) The address for indirect specification uses two words. Therefore, to substitute indirect specification for index modification, the addition/subtraction of 32-bit data is required. The following is the ladder used for the address addition/subtraction of the device stored in D1 and D0 for indirect specification.

[To add "1" to the address of the device for indirect specification]

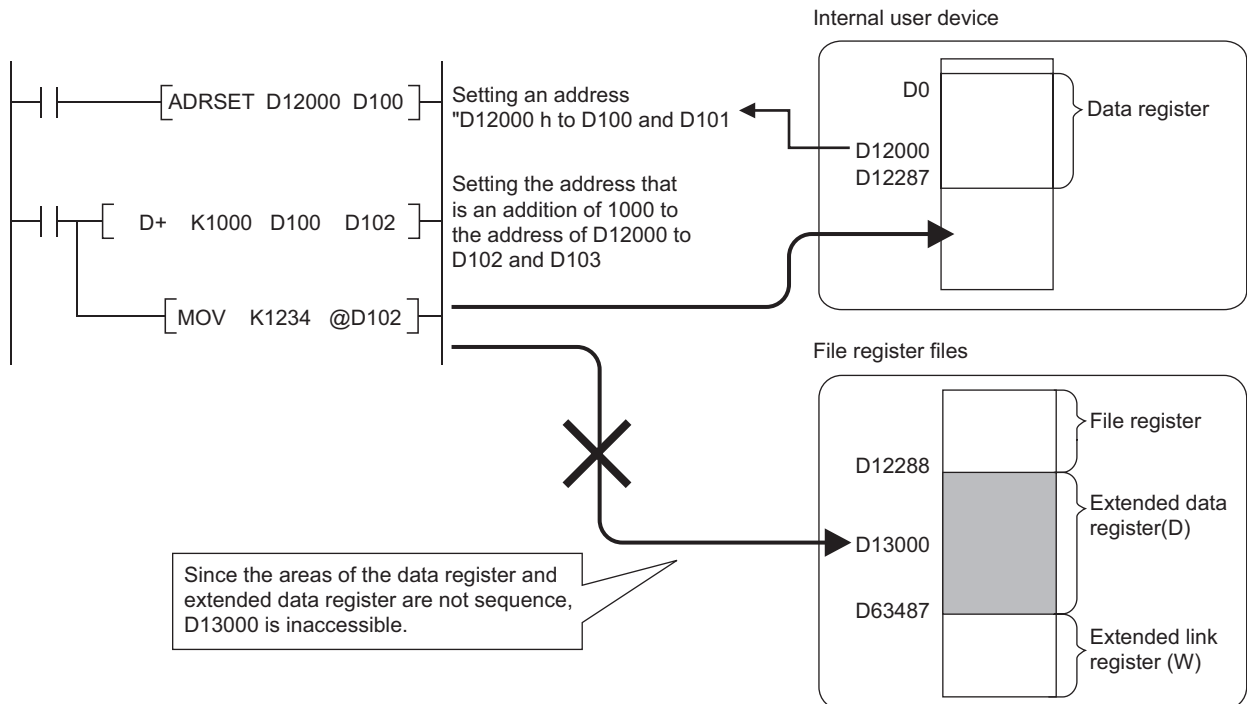


[To subtract "1" from the address of the device for indirect specification]



- (b) Indirect specification of extended data register (D) and extended link register (W)  
Indirect specification with indirect address can be performed in the extended data register (D) and extended link register (W).

Note that when indirect specification is performed to the extended data register (D) and data register (D) in internal device or to the extended link register (W) and link register (W) in internal device, the areas of the internal user device and extended data register (D) or extended link register (W) are not treated as a sequence.





## 3.5 Reducing Instruction Processing Time

### 3.5.1 Subset Processing

Subset processing is used to place limits on bit devices used by basic instructions and application instructions in order to increase processing speed.

However, the instruction symbol does not change.

To shorten scans, run instructions under the conditions indicated below.

(1) Conditions which each device must meet for subset processing

(a) When using word data

Device	Condition
Bit device	<ul style="list-style-type: none"> <li>• Designates a bit device number in a factor of 16.</li> <li>• Only K4 can be designated for digit designation.</li> <li>• Does not perform indexing.</li> </ul>
Word device	<ul style="list-style-type: none"> <li>• Internal user device.</li> <li>• File register (R, ZR <sup>*1</sup>)</li> <li>• Multiple CPU shared device <sup>*1, *2</sup></li> <li>• Index register (Z) / Standard device register (Z) <sup>*1</sup></li> </ul>
Constants	<ul style="list-style-type: none"> <li>• No limitations</li> </ul>

(b) When using double word data

Device	Condition
Bit device	<ul style="list-style-type: none"> <li>• Designates a bit device number in a factor of 16.</li> <li>• Only K8 can be designated for digit designation.</li> <li>• Does not perform indexing.</li> </ul>
Word device	<ul style="list-style-type: none"> <li>• Internal user device.</li> <li>• File register (R, ZR <sup>*1</sup>)</li> <li>• Multiple CPU shared device <sup>*1, *2</sup></li> <li>• Index register (Z) / Standard device register (Z) <sup>*1</sup></li> </ul>
Constants	<ul style="list-style-type: none"> <li>• No limitations</li> </ul>

(c) When using bit data

Device	Condition
Bit device	<ul style="list-style-type: none"> <li>• Internal user device (indexing possible)</li> </ul>
Word device	<ul style="list-style-type: none"> <li>• Bit specification of internal user device</li> <li>• Bit specification of file register (R, ZR <sup>*1</sup>)</li> <li>• Bit specification of multiple CPU shared device <sup>*1, *2</sup></li> </ul>

\*1: Only for Universal model QCPU

\*2: Valid only for the multiple CPU high speed transmission area (from U3En\G10000)

(Excluding the case that indexing is executed for the head I/O number of the CPU module (U3En\G10000))

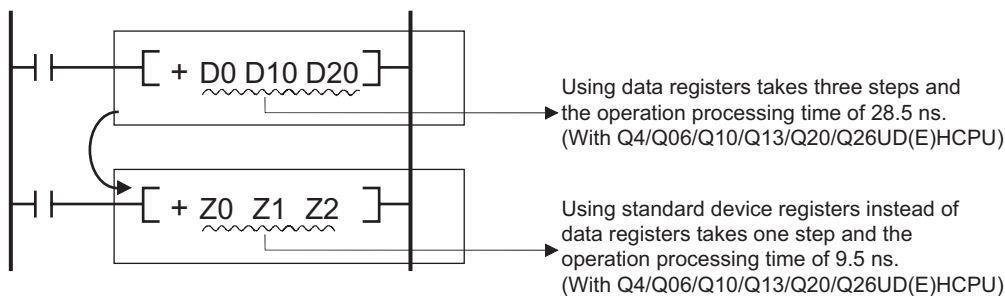
(2) Instructions for which subset processing can be used

Types of Instructions	Instruction Symbols
Contact instructions	LD,LDI,AND,ANI,OR,ORI,LDP,LDF,ANDP,ANDF,ORP,ORF,LDPI,ANDPI,ANDFI,ORPI,ORFI
Output instructions	OUT,SET,RST
Comparison operation instruction	• =, <>, <, <=, >, >=, D=, D<>, D<, D<=, D>, D>=
Arithmetic operation	• +, -, *, /, INC, DEC, D+, D-, D*, D/, DINC, DDEC • B+, B-, B*, B/, E+, E-, E*, E/
Data conversion instructions	• BCD, BIN, DBCD, DBIN, FLT, DFLT, INT, DINT
Data transfer instruction	• MOV, DMOV, CML, DCML, XCH, DXCH • FMOV, BMOV, EMOV
Program branch instruction	• CJ, SCJ, JMP
Logic operations	• WAND, DAND, WOR, DOR, WXOR, DXOR, WXNR, DXNR
Rotation instruction	• RCL, DRCL, RCR, DRCR, ROL, DROL, ROR, DROR
Shift instruction	• SFL, DSFL, SFR, DSFR
Data processing instructions	• SUM, SEG
Structure creation instructions	• FOR, CALL

### 3.5.2 Operation processing with standard device registers (Z) (only Universal model QCPU)

Operation processing time can be reduced with standard device registers (Z).

The following shows an example program with standard device registers.



Operation processing time is reduced with the instructions that the subset processing is possible.

For the number of steps, refer to Section 3.8.

For the operation time for each instruction, refer to Appendix 1.

#### POINT

Because standard device registers are the same devices as index registers, do not use device numbers of the standard device registers for the index registers.

## 3.6 Cautions on Programming (Operation Errors)

Operation errors are returned in the following cases when executing basic instructions and application instructions with CPU module:

- An error listed on the explanatory page for the individual instruction occurred.
- When an intelligent function module device is used, no intelligent function module is installed at the specified I/O number position.
- When an intelligent function module device is used, the specified buffer memory address does not exist.
- The relevant network does not exist when using a link device.
- When a link device is used, no network module is installed at the specified I/O number position.
- When a multiple CPU shared device is used, a CPU module is not installed at the head I/O number position of the specified CPU module.
- When a multiple CPU shared device is used, the specified shared memory address does not exist.
- The setting of the device number crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W).  
(Universal model QCPU only)

### POINT

When file register is set but a memory card is not installed or when file register is not set, writing/reading to/from file register is as follows:

- (1) For the High Performance model QCPU, Process CPU, and Redundant CPU  
An error does not occur even when writing/reading to/from file register is performed. However, "0H" is stored when reading from file register is performed.
- (2) For the Universal model QCPU  
The OPERATION ERROR (error code:4101) occurs when writing/reading to/from file register is performed.

#### (1) Device range check

Device range checks for the devices used by basic instructions and application instructions in CPU module are as indicated below:

##### (a) Instructions for specified each device, including MOV and DMOV

###### 1) CPUs other than Universal model QCPU

The device range is not checked. In cases where the corresponding device range is exceeded, data is written to other devices. \*1

For example, in a case where the data register has been allocated 12k points, there will be no error even if it exceeds D12287.



→ This designates D12287 and D12288 as the target devices for executing the DMOV instruction. However, since D12288 does not exist, data in another device is corrupted.

Device range checks are not conducted also in cases where indexing is being performed.

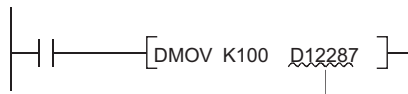
In cases where the corresponding device range is exceeded as the result of performing indexing, data is written to other devices.\*1

\*1: For the assignment order of internal user devices, refer to this Section (c) Character string data.

## 2) Universal model QCPU

The device range is checked. When the device number is outside the device range, an operation error occurs.

For example, when 12 k points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.



When D12287 is specified with the DMOV instruction, the target devices are D12287 and D12288. However, an operation error occurs because D12288 does not exist.

The device range is checked even though indexing is executed.

With changing the settings of the PLC parameter on GX Developer, the device range is not checked.\*2

- \*2: For changing the settings of the PLC parameter on GX Developer, refer to the following manual.
- QCPU User's Manual (Function Explanation, Program Fundamentals)

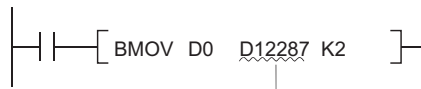
### (b) Instructions for a block of devices, including BMOV and FMOV

#### 1) CPUs other than Universal model QCPU

The device range is checked.

When the device number is outside the device range, an operation error occurs.

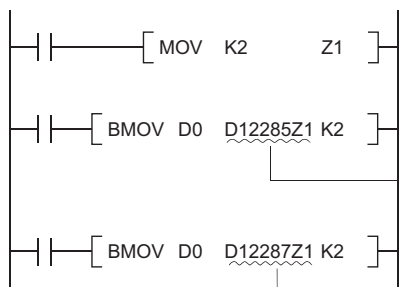
For example, when 12 k points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.



This designates D12287 and D12288 as the target devices for executing the BMOV instruction. However, since D12288 does not exist, an operation error occurs.

Device range checks are also conducted when indexing is performed.

However, if indexing has been conducted, there will be no error returned if the initial device number exceeds the relevant device range.



When D12287 is specified with the BMOV instruction, the target devices are D12287 and D12288. However, an operation error occurs because D12288 does not exist.

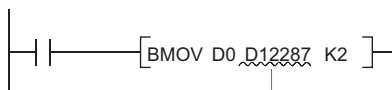
An operation error occurs since head device number is D12289 that exceeds the device range.

## 2) Universal model QCPU

The device range is checked.

When the device number is outside the device range, an operation error occurs.

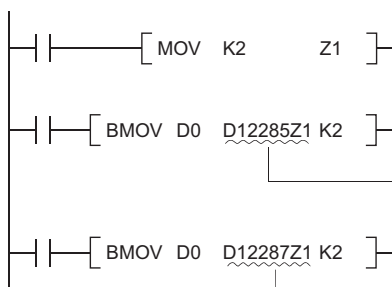
For example, when 12 k points are assigned to a data register, an error occurs if the device number of the data register exceeds D12287.



When D12287 is specified with the BMOV instruction, the target devices are D12287 and D12288. However, an operation error occurs because D12288 does not exist.

The device range is checked even though indexing is executed.

An error occurs when the head device number of the devices with indexing exceeds the device range.



When D12287 is specified with the BMOV instruction, the target devices are D12287 and D12288. However, an operation error occurs because D12288 does not exist.

An operation error occurs since head device number is D12289 that exceeds the device range.

With changing the settings of the PLC parameter on GX Developer, the device range is not checked.\*2

- \*2: For changing the settings of the PLC parameter on GX Developer, refer to the following manual.
- QCPU User's Manual (Function Explanation, Program Fundamentals)

### (c) Character string data

Because all character string data is of variable length, device range checks are performed.

In cases where the corresponding device range has been exceeded, an operation error will be returned.

For example, in a case where the data register has been allocated 12k points, there will be an error if it exceeds D12287.



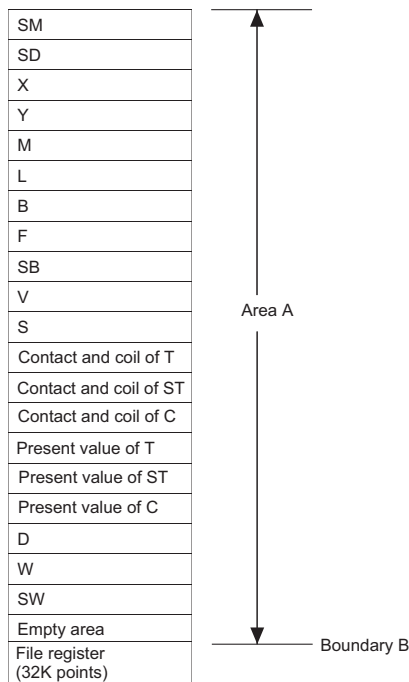
This designates D12287 and D12288 as the target devices for executing the \$MOV instruction. However, since D12288 does not exist, an operation error occurs.

However, with CPUs other than Universal model QCPU, when indexing is executed and the head device number is outside the device range, no error occurs and the other devices are accessed.

When performing the following access in Universal model QCPU, an error (error code: 4101) occurs.

- 1) Access crossing the boundary of devices caused by indexing (range of A area)

The allocation order of individual devices is shown below:



- 2) Access crossing the boundary of file registers caused by indexing
- 3) Access to file registers (R, ZR) without setting file register files
- 4) Access to file registers (R, ZR) exceeded the range of file register files

Presetting PC parameter not to check indexing device range enables the Universal model QCPU not to detect an error in the above accesses from 1) to 4).

Detecting an error in the above accesses from 1) to 4) , however, depends on the serial No. of Universal model QCPU.\*2

Setting device range in indexing	First 5 digits of serial No. for Universal model QCPU	
	Serial No."10021" or lower	Serial No."10022" or higher
Set	Detected errors in accesses 1) to 4)	
Not set	Detected errors in accesses 2) to 4)	Not detected

\*2: For changing the settings of the PLC parameter on GX Developer, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

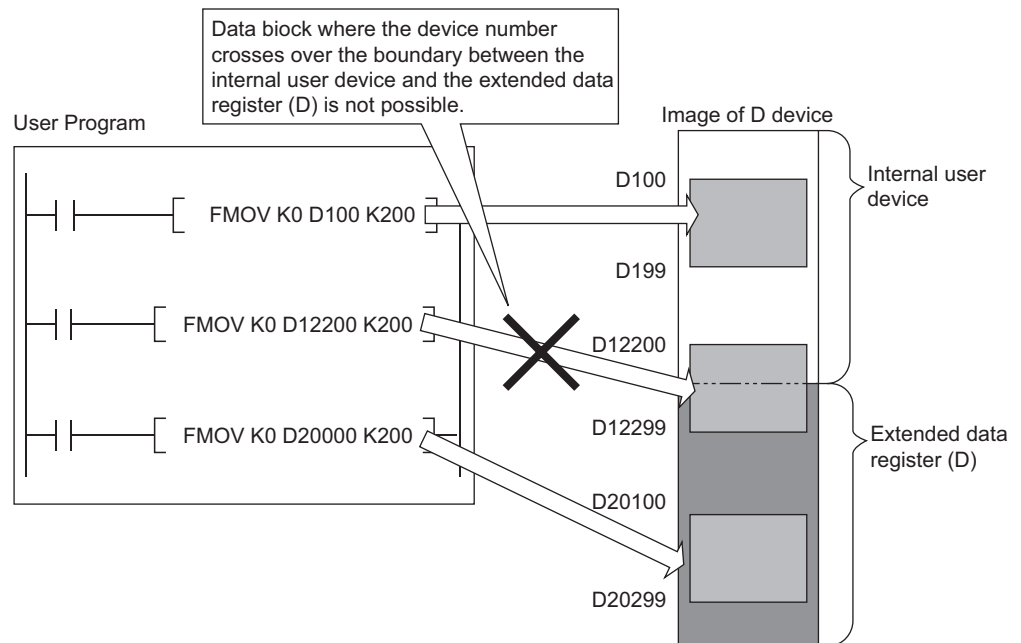
### **POINT**

When indexing is executed only with Universal model QCPU, devices between internal user devices (SW) and file registers (R) cannot be skipped. (Error code: 4101).

**Remark**

For the how to change the internal user device allocation, refer to User's Manual (Functions Explanation, Program Fundamentals) for the CPU module used.

- (d) Device range checks are conducted when indexing is performed by direct access output (DY).
- (e) Set the following items so that the specification does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).
  - Index modification
  - Indirect specification
  - Specification for the instructions which target data block\*1



\*1 Data block indicates the following data.

- Data used in the instructions, such as FMOV, BMOV, BK+, which multiple words are targeted for operation
- Control data, composed of two or more words, specified in the instructions, such as SP.FWRITE, SP.FREAD
- Data whose data type is 32-bit or more (BIN 32-bit, real number, indirect address of the device)

(2) Device data check

Device data checks for the devices used by basic instructions and application instructions in CPU module are as indicated below:

(a) When using BIN data

No error is returned even if the operation results in overflow or underflow. The carry flag does not go on at such times, either.

(b) When using BCD data

1) Each digit is check for BCD value (0 to 9). An operation error is returned if individual digits are outside the 0 to 9 (A to F) range.

2) No error is returned even if the operation results in overflow or underflow. The carry flag does not go on at such times, either.

(c) When using floating-point data

1) An operation error occurs when the following operation results are returned with the single-precision floating-point operation instruction.

When the absolute value of the floating decimal point data is  $1.0 \times 2^{-127}$  or lower

When absolute value of floating decimal point data is  $1.0 \times 2^{128}$  or higher

2) An operation error occurs when the following operation results are returned with the double-precision floating-point operation instruction.

When the absolute value of the floating decimal point data is  $1.0 \times 2^{-1023}$  or lower

When absolute value of floating decimal point data is  $1.0 \times 2^{1024}$  or higher

(d) Using character string data

No data check is conducted.

(3) Buffer memory access

For accessing buffer memories, using instructions with intelligent function module devices (from Un\G0) is recommended.

(4) Multiple CPU shared memory access

For accessing multiple CPU shared memories, using instructions with multiple CPU shared devices (from U3En\G10000) is recommended.



## 3.7 Conditions for Execution of Instructions

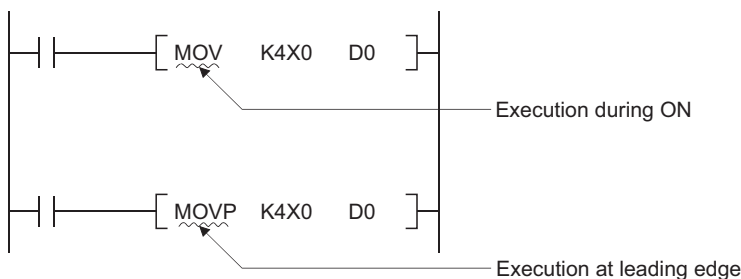
The following four types of execution conditions exist for the execution of CPU module sequence instructions, basic instructions, and application instructions:

- Non-conditional execution..... Instructions executed without regard to the ON/OFF status of the device  
**Example** LD X0, OUT Y10
- Executed at ON..... Instructions executed while input condition is ON  
**Example** MOV instruction, FROM instruction
- Executed at leading edge ..... Instructions executed only at the leading edge of the input condition (when it goes from OFF to ON) Example PLS instruction, MOVP instruction.
- Executed at trailing edge ..... Instructions executed only at the trailing edge of the input condition (when it goes from ON to OFF) Example PLF instruction.

For coil or equivalent basic instructions or application instructions, where the same instruction can be designated for either execution at ON or leading edge execution, a "P" is added after the instruction name to specify the condition for execution.

- Instruction to be executed at ON **Instruction name**
- Instruction to be executed at leading edge **Instruction name** + P

Execution at ON and execution at leading edge for the MOV instruction are designated as follows:



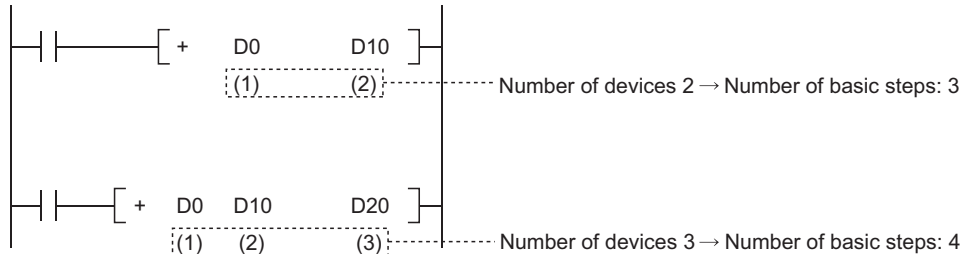
## 3.8 Counting Step Number

The number of steps in CPU module sequence instructions, basic instructions, and application instructions differs depending on whether indirect setting of the device used is possible or not.

### (1) Counting the number of basic steps

The basic number of steps for basic instructions and application instructions is calculated by adding the device number and 1.

For example, the "+" instruction" would be calculated as follows:



### (2) Conditions for increasing the number of steps

The number of steps is increased over the number of basic steps in cases where a device is used that is designated indirectly or for which the number of steps is increased.

#### (a) When device is designated indirectly

In cases where indirect designation is done by @□□□, the number of steps is increased 1 step over the number of basic steps.

For example, when a 3-step MOV instruction is designated indirectly (example: MOV K4X0 @D0), one step is added and the instruction becomes 4 steps.

#### (b) Devices with additional steps (Except Universal model QCPU)

Devices with Additional Steps	Added Steps	Example
Intelligent function module device	1	MOV <u>U4</u> \G10 D0
Multiple CPU shared device		MOV <u>U3E1</u> \G0 D0
Link direct device		MOV <u>J3</u> \B20 D0
Index register		MOV <u>Z0</u> D0
Serial number access format file register		MOV <u>ZR123</u> D0
32-bit constant		DMOV <u>K123</u> D0
Real constant		EMOV <u>E0.1</u> D0
Character string constant		For even numbers: (number of characters) / 2 For odd numbers: (number of characters + 1) / 2

(c) Devices with additional steps (Universal model QCPU(except Q00UJCPU))

1) Instructions applicable to subset processing

The following table shows steps depending on the devices.

Instruction Symbols	Devices with Additional Steps	Added Steps (Number of Instruction Steps)	Basic Number of Steps
LD,LDI,AND,ANI,OR,ORI, LDP,LDF,ANDP,ANDF,ORP,ORF	Serial number access format file register, Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device		
LDPI,LDFI	Serial number access format file register, Extended data register (D), Extended link register (W)	1(4)	3
	Multiple CPU shared device		
ANDPI,ANDFI,ORPI,ORFI	Serial number access format file register, Extended data register (D), Extended link register (W)	1(5)	4
	Multiple CPU shared device		
SET	Serial number access format file register Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device		
OUT	Timer/Counter	3(4)	1
	Serial number access format file register Extended data register (D), Extended link register (W)	1(2)	
	Multiple CPU shared device		
RST (bit device)	Serial number access format file register Extended data register (D), Extended link register (W)	1(2)	1
	Multiple CPU shared device		
RST (word device)	Timer/Counter (Bit/word device)	2(4)	2
	Serial number access format file register Extended data register (D), Extended link register (W)	1(3)	
	Multiple CPU shared device	1(3)	
LD=,LD<>,LD<,LD<=,LD>,LD>=, AND=,AND<>,AND<,AND<=,AND>,AND>=, OR=,OR<>,OR<,OR<=,OR>,OR>=	Standard device register *2	-1	3
	Serial number access format file register Extended data register (D), Extended link register (W)	1	
	Multiple CPU shared device		
LDD=,LDD<>,LDD<,LDD<=,LDD>,LDD>=, ANDD=,ANDD<>,ANDD<,ANDD<=,ANDD>,ANDD>=, AND>=,ORD=,ORD<>,ORD<,ORD<=, ORD>,ORD>=	Standard device register *2	-1	3
	Serial number access format file register Extended data register (D), Extended link register (W)	1	
	Multiple CPU shared device		
	Decimal constant, hexadecimal constant, real constant		
+,-,+P,-P,WAND,WOR,WXOR,WXNR, WANDP,WORP,WXORP,WXNRP (2 devices)	Standard device register *2	Ⓓ:-1	3
	Serial number access format file register Extended data register (D), Extended link register (W)	Ⓔ:1,Ⓓ:3	
	Multiple CPU shared device		

Instruction Symbols	Devices with Additional Steps	Added Steps (Number of Instruction Steps)	Basic Number of Steps
D+,D-,D+P,D-P,DAND,DOR,DXOR,DXNR, DANDP,DORP,DXORP,DXNRP (2 devices)	Standard device register *2	⓪:-1	3
	Serial number access format file register Extended data register (D), Extended link register (W)	Ⓢ1:1,⓪:3	
	Multiple CPU shared device		
	Decimal constant, hexadecimal constant, real constant	Ⓢ1:1	
+,-,+P,-P,WAND,WOR,WXOR,WXNR, WANDP,WORP,WXORP,WXNRP (3 devices)*1	Serial number access format file register Extended data register (D), Extended link register (W)	Ⓢ1,Ⓢ2:1,⓪:2	3
	Multiple CPU shared device		
D+,D-,D+P,D-P,DAND,DOR,DXOR,DXNR, DANDP,DORP,DXORP,DXNRP (3 devices)*1	Serial number access format file register Extended data register (D), Extended link register (W)	Ⓢ1,Ⓢ2:1,⓪:2	3
	Multiple CPU shared device		
	Decimal constant, hexadecimal constant, real constant	Ⓢ1,Ⓢ2:1	
*, *P, /, /P	Serial number access format file register Extended data register (D), Extended link register (W)	Ⓢ1,Ⓢ2:1,⓪:2	3
	Multiple CPU shared device		
D*, D*P, D/, D/P, E*, E*P	Serial number access format file register Extended data register (D), Extended link register (W)	Ⓢ1,Ⓢ2:1,⓪:2	3
	Multiple CPU shared device		
	Decimal constant, hexadecimal constant, real constant	Ⓢ1,Ⓢ2:1	

Instruction Symbols	Devices with Additional Steps	Added Steps (Number of Instruction Steps)	Basic Number of Steps
INC, INCP, DEC, DECP, DINC, DINCP, DDEC, DDECP	Index register/Standard device register *2	-1	2
	Serial number access format file register Extended data register (D), Extended link register (W)	3	
	Multiple CPU shared device		
MOV, MOVP	Serial number access format file register Extended data register (D), Extended link register (W)	1	2
	Multiple CPU shared device		
DMOV, DMOVP, EMOV, EMOVP	Serial number access format file register Extended data register (D), Extended link register (W)	1	2
	Multiple CPU shared device		
	Decimal constant, hexadecimal constant, real constant		
BCD, BCDP, BIN, BINP, FLT, FLTP, CML, CMLP	Serial number access format file register Extended data register (D), Extended link register (W)	①:1, ②:2	2
	Multiple CPU shared device		
DBCD, DBCDP, DBIN, DBINP, INT, INTP, DINT, DINTP, DFLT, DFLTP, DCML, DCMLP	Serial number access format file register Extended data register (D), Extended link register (W)	①:1, ②:2	2
	Multiple CPU shared device		
	Decimal constant, hexadecimal constant, real constant		
		①:1	

- \*1: If the same device is used for ① and ②, the number of basic steps increases by one.
- \*2: The number of steps decreases with a standard device register.

When multiple standard device registers are used in an instruction applicable to subset processing, the number of steps decreases. The following table shows the number of steps for each instruction.

Instruction Symbols	Locations Where Standard Device Register Is Used	Added Steps (Number of Instruction Steps)	Basic Number of Steps
LD=, LD<>, LD<, LD<=, LD>, LD>=, AND=, AND<>, AND<, AND<=, AND>, AND>=, OR=, OR<>, OR<, OR<=, OR>, OR>= LDD=, LDD<>, LDD<, LDD<=, LDD>, LDD>=, ANDD=, ANDD<>, ANDD<, ANDD<=, ANDD>, ANDD>=, ORD=, ORD<>, ORD<, ORD<=, ORD>, ORD>=	① and ②	-2(1)	3
+, -, +P, -P, D+, D-, D+P, D-P, WAND, WOR, WXOR, WXNR, DAND, DOR, DXOR, DXNR, WANDP, WORP, WXORP, WXNRP, DANDP, DORP, DXORP, DXNRP (2 devices)	① and ④	-2(1)	3
+, -, +P, -P, D+, D-, D+P, D-P, WAND, WOR, WXOR, WXNR, DAND, DOR, DXOR, DXNR, WANDP, WORP, WXORP, WXNRP, DANDP, DORP, DXORP, DXNRP (3 devices)*1	①, ②, and ④	-2(1)	3
	①, or ② and ④	-1(2)	
	① and ② (only when that device that the number of steps does not increase is specified for ④)	±0(3)	
	① and ② (only when a serial number access format file register is specified for ④)	+2(5)	

- \*1: If the same device is used for ① and ④, the number of basic steps increases by one.

Instruction Symbols	Locations Where Standard Device Register Is Used	Added Steps (Number of Instruction Steps)	Basic Number of Steps
*, *P, /, /P	Ⓢ1, Ⓢ2, and Ⓣ	-2(1)	3
	Ⓢ1, or Ⓢ2 and Ⓣ	-1(2)	
D*, D*P, D/, D/P, E*, E*P	Ⓢ1, Ⓢ2, and Ⓣ	-2(1)	3
	Ⓢ1, or Ⓢ2 and Ⓣ	-1(2)	
	Ⓢ1 and Ⓢ2 (only when that device that the number of steps does not increase is specified for Ⓣ)	±0(3)	
	Ⓢ1 and Ⓢ2 (only when a serial number access format file register is specified for Ⓣ)	+2(5)	
MOV,MOV P,DMOV,DMOV P,EMOV,EMOV P	Ⓢ1 and Ⓣ	-1(1)	2
BCD,BCDP,BIN,BINP,DBCD,DBCDP, DBIN,DBINP,FLT,FLTP,DFLT,DFLTP, INT,INTP,DINT,DINTP,CML,CMLP, DCML,DCMLP	Ⓢ1 and Ⓣ	-1(1)	2

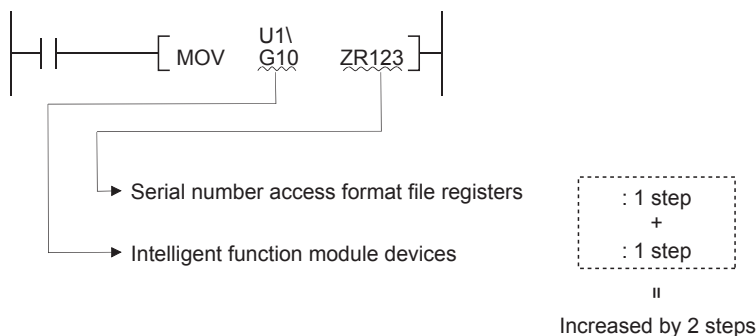
2) Except Instructions applicable to subset processing

The following table shows steps depending on the devices.

Devices with Additional Steps	Added Steps	Example
Intelligent function module device	1	MOV <u>U4</u> \G10 D0
Multiple CPU shared device		MOV <u>U3E1</u> \G10000 D0
Link direct device		MOV <u>J3</u> \B20 D0
Index register / standard device register		MOV <u>Z0</u> D0
Serial number access format file register		MOV <u>ZR123</u> D0
Extended data register(D)		MOV D123
Extended link register(W)		MOV W123
32-bit constant		DMOV <u>K123</u> D0
Real constant		EMOV <u>E0.1</u> D0
Character string constant		For even number: (number of characters) / 2 For odd numbers: (number of characters + 1) / 2

(d) In cases where the conditions described in (a) to (c) above overlap, the number of steps becomes a culmination of the two.

**Example** MOV If U1\G10 ZR123 has been designated, a total of 2 steps are added.



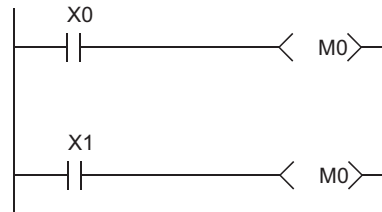
## 3.9 Operation when the OUT, SET/RST, or PLS/PLF Instructions Use the Same Device

The following describes the operation for executing multiple instructions of the OUT, SET/RST, or PLS/PLF that use the same device in one scan.

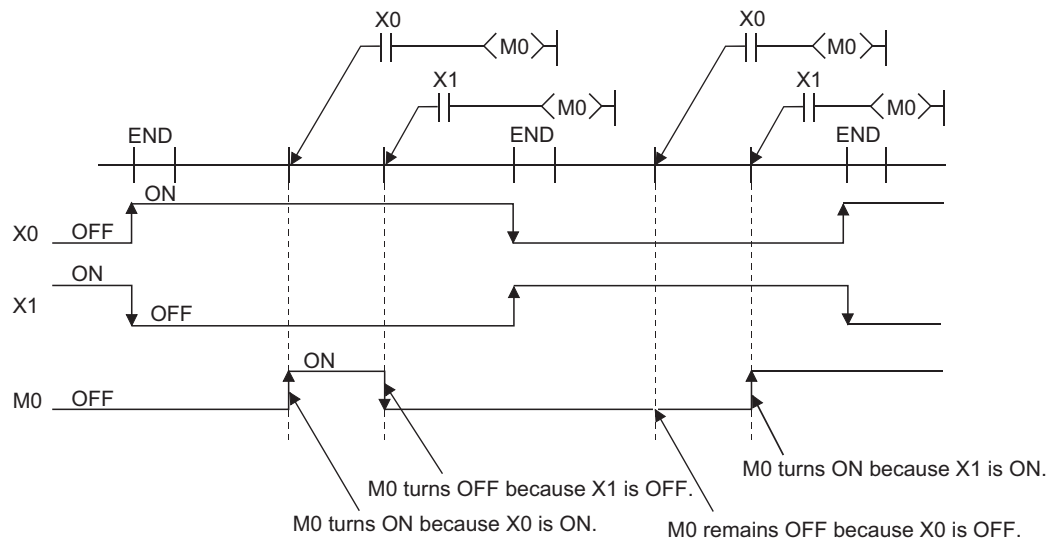
### (1) OUT instructions using the same device

Do not program more than one OUT instruction using the same device in one scan. If the OUT instructions using the same device are programmed in one scan, the specified device will turn ON or OFF every time the OUT instruction is executed, depending on the operation result of the program up to the relevant OUT instruction. Since turning ON or OFF of the device is determined when each OUT instruction is executed, the device may turn ON and OFF repeatedly during one scan. The following diagram shows an example of a ladder that turns the same internal relay (M0) with inputs X0 and X1 ON and OFF.

[Ladder]



[Timing Chart]



With the refresh type CPU module, when the output (Y) is specified by the OUT instruction, the ON/OFF status of the last OUT instruction of the scan will be output.

(2) SET/RST instructions using the same device

- (a) The SET instruction turns ON the specified device when the execution command is ON and performs nothing when the execution command is OFF.

For this reason, when the SET instructions using the same device are executed two or more times in one scan, the specified device will be ON if any one of the execution commands is ON.

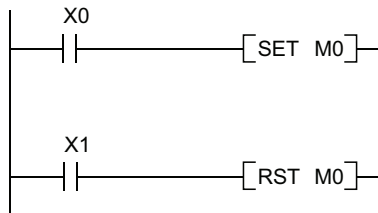
- (b) The RST instruction turns OFF the specified device when the execution command is ON and performs nothing when the execution command is OFF.

For this reason, when the RST instructions using the same device are executed two or more times in one scan, the specified device will be OFF if any one of the execution commands is ON.

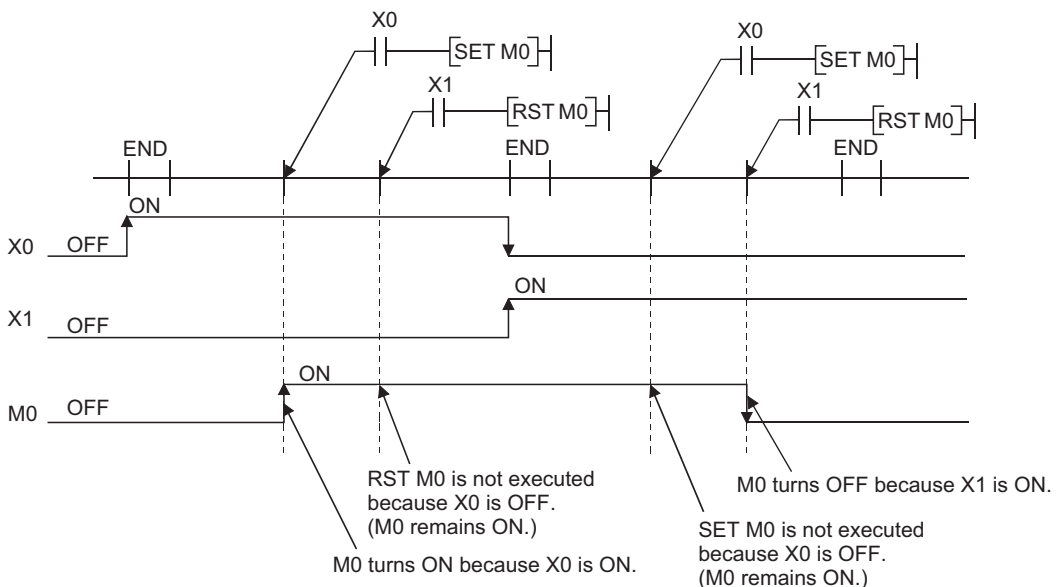
- (c) When the SET instruction and RST instruction using the same device are programmed in one scan, the SET instruction turns ON the specified device when the SET execution command is ON and the RST instruction turns OFF the specified device when the RST execution command is ON.

When both the SET and RST execution commands are OFF, the ON/OFF status of the specified device will not be changed.

[Ladder]



[Timing Chart]



When using a refresh type CPU module and specifying output (Y) in the SET/RST instruction, the ON/OFF status of the device at the execution of the last instruction in the scan is returned as the output (Y).



(3) PLS instructions using the same device

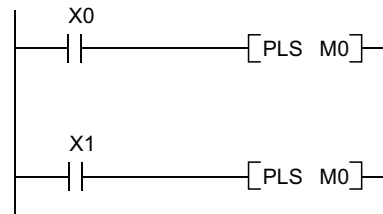
The PLS instruction turns ON the specified device when the execution command is turned ON from OFF.

It turns OFF the device at any other time (OFF to OFF, ON to ON, or ON to OFF).

If two or more PLS instructions using the same device are executed in one scan, each instruction turns ON the device when the corresponding execution command is turned ON from OFF and turns OFF the device in other cases.

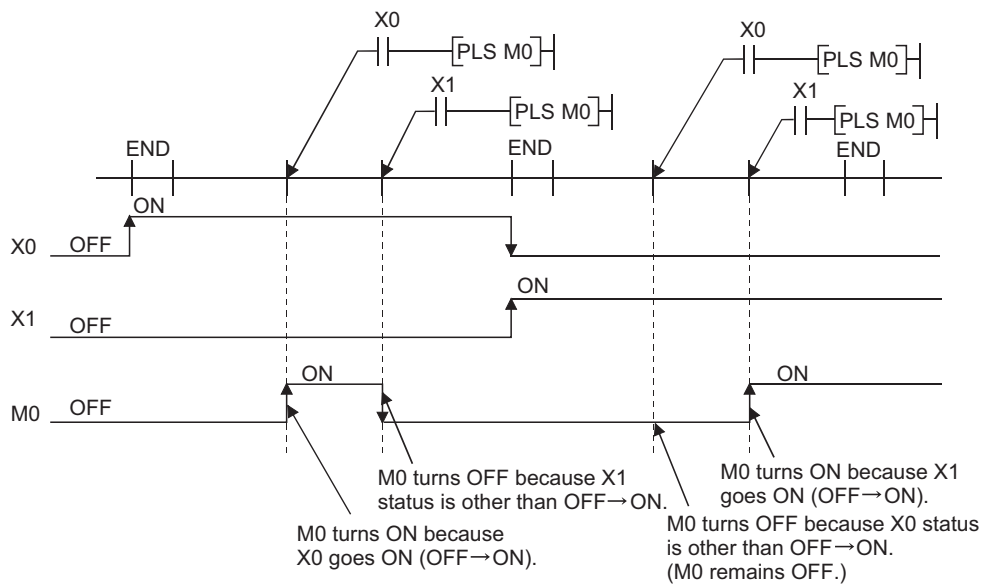
For this reason, if multiple PLS instructions using the same device are executed in a single scan, a device that has been turned ON by the PLS instruction may not be turned ON during one scan.

[Ladder]

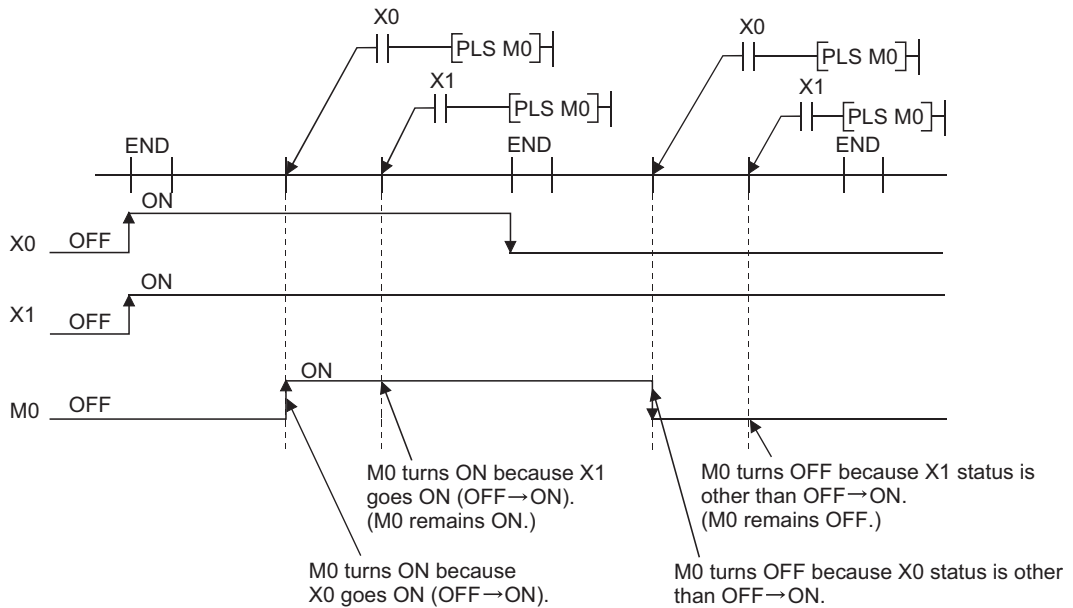


[Timing Chart]

- The ON/OFF timing of the X0 and X1 is different. (The specified device does not turn ON throughout the scan.)



- The X0 and X1 turn ON from OFF at the same time.



When using a refresh type CPU module and specifying output (Y) in the PLS instructions, the ON/OFF status of the device at the execution of the last PLS instruction in the scan is returned as the output (Y).

(4) PLF instructions using the same device

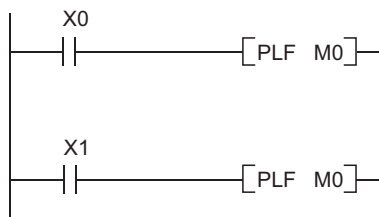
The PLF instruction turns ON the specified device when the execution command is turned OFF from ON.

It turns OFF the device at any other time (OFF to OFF, OFF to ON, or ON to ON).

If two or more PLF instructions using the same device are executed in one scan, each instruction turns ON the device when the corresponding execution command is turned OFF from ON and turns OFF the device in other cases.

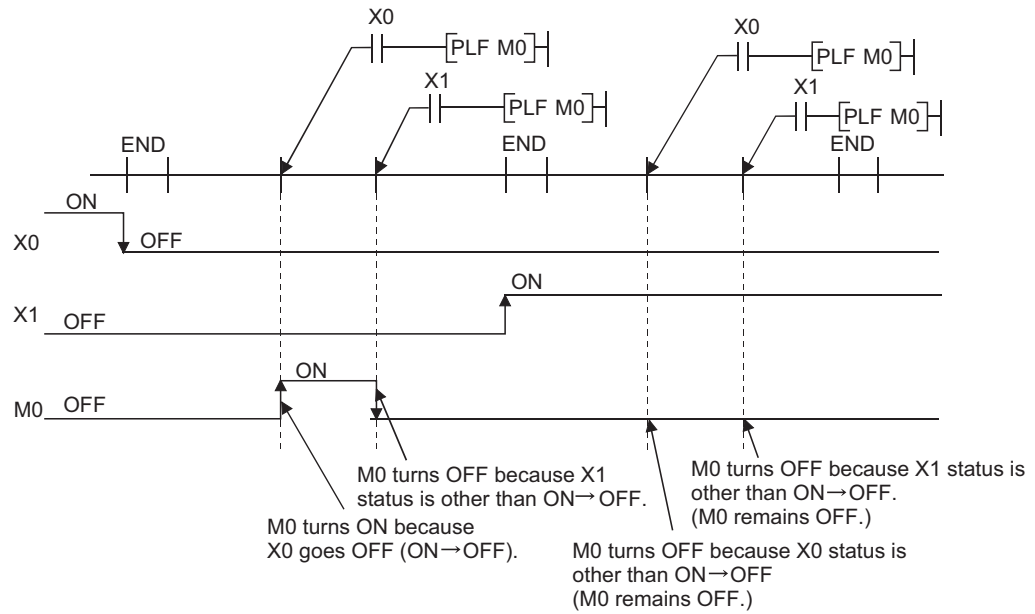
For this reason, if multiple PLF instructions using the same device are executed in a single scan, a device that has been turned ON by the PLF instruction may not be turn ON during one scan.

[Ladder]

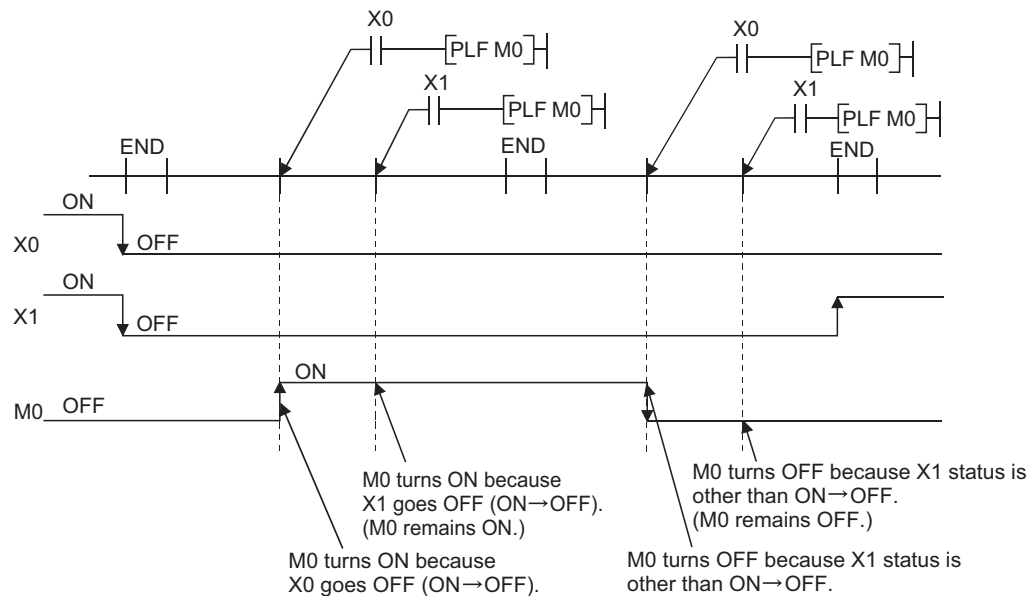


[Timing Chart]

- The ON/OFF timing of the X0 and X1 is different. (The specified device does not turn ON throughout the scan.)



- The X0 and X1 turn OFF from ON at the same time.



When using a refresh type CPU module and specifying output (Y) in the PLF instructions, the ON/OFF status of the device at the execution of the last PLF instruction in the scan is returned as the output (Y).

## 3.10 Precautions for Use of File Registers

This section explains the precautions for use of the file registers in the QCPU.

(1) CPU modules that cannot use file registers

The Q00JCPU and Q00UJCPU cannot use the file registers. When using the file registers, use the CPU module of other than the Q00JCPU and Q00UJCPU.

(2) Setting of file registers to be used

When using the file registers, the file registers to be used must be set with the PLC parameter or QDRSET instruction. (The PLC parameters of the Q00CPU and Q01CPU need not be set since they are preset to "Use file register".) If the file registers to be used have not been set, normal operation cannot be performed with the instructions that use the file registers.

### POINT

Even when file registers to be used are not set in the PLC parameter, a program that uses file registers can be created. For the CPU module other than the Universal model QCPU, an error does not occur when that program is written to the CPU module.

However, note that the correct data cannot be written/read to/from the file register. For the Universal model QCPU, an error occurs if the program where file registers are used is executed.

(3) Securing of file register area

(a) High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU

When using file registers, register the file registers to the standard RAM/memory card to secure the file register area.

(b) Basic Model QCPU (except Q00JCPU)

The file register area has been secured in the standard RAM beforehand. The user need not secure the file register area.

The following table indicates the memories that can use the file registers in each CPU module.

Memory	High Performance model QCPU	Basic Model QCPU (except Q00JCPU)
	Process CPU Redundant CPU Universal model QCPU	
Standard RAM	○	○
Memory card *1 *2	○*3	×

○ : Can be registered, × : Cannot be registered.

\*1: When the flash memory is used, only read from the file registers can be performed. (Write to the flash ROM cannot be performed.)

\*2: When the E<sup>2</sup>PROM is used, write to the E<sup>2</sup>PROM can be performed with the PROMWR instruction.

\*3: Unusable for the Q00UCPU and Q01UCPU.

**Remark**

For the file register setting method and file register area securing method, refer to User's Manual (Functions Explanation, Program Fundamentals) for the CPU module used.

(4) Designation of file register number in excess of the registered number of points

(a) CPUs other than Universal model QCPU

An error will not occur if data are written or read to or from the file registers that have numbers greater than the registered number of points. However, note that the read/write of correct data to/from the file registers cannot be performed.

(b) Universal model QCPU

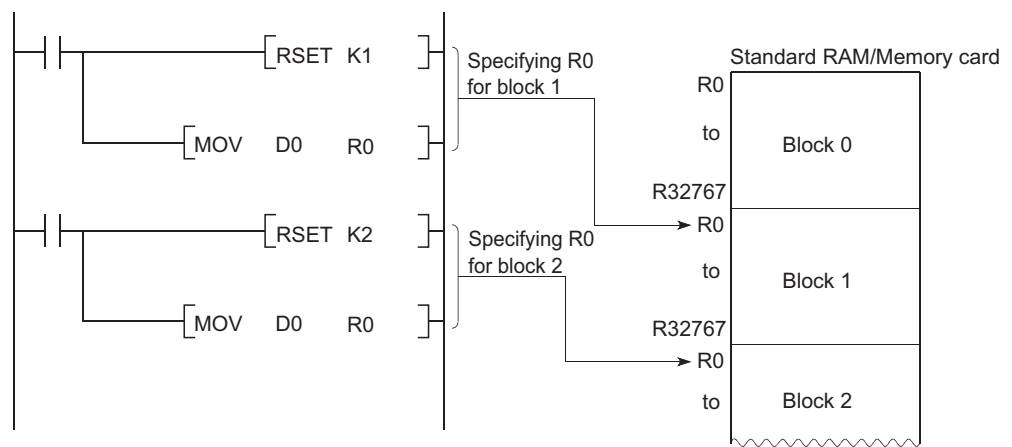
When data are written to or read from the file registers that are not registered, an error occurs. (Error code: 4101)

(5) File register specifying method

There are the block switching method and serial number access method to specify the file registers.

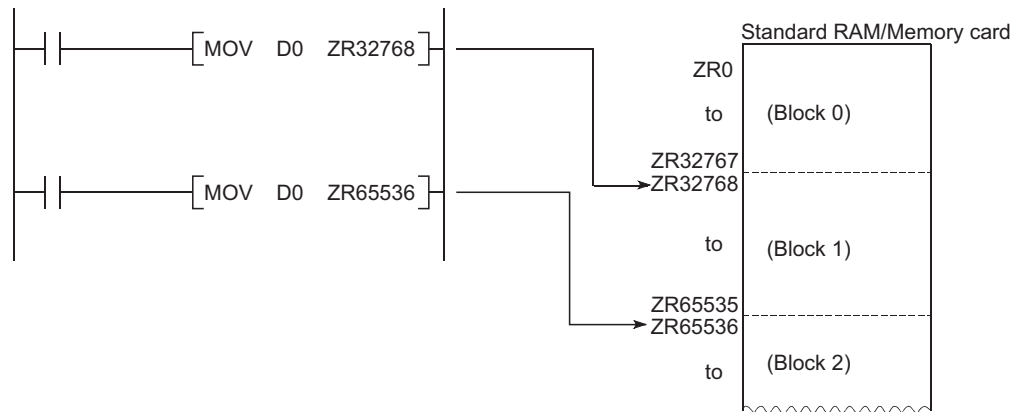
(a) Block switching method

In the block switching method, specify the number of used file register points in units of 32k points (one block). For file registers of 32k points or more, specify the file registers by switching the block No. to be used with the RSET instruction. Specify each block as R0 to R32767.



(b) Serial number access method

In the serial number access method, specify the file registers beyond 32k points with consecutive device numbers. The file registers of multiple blocks can be used as consecutive file registers. Use "ZR" as the device name.



(6) Settings and restrictions when refreshing file registers

(a) Settings

The settings of refresh devices are as follows.

- Refresh settings for CC-Link IE controller network
- Refresh settings for MELSECNET/H
- Refresh settings for CC-Link
- Auto refresh settings for the intelligent function module
- Auto refresh settings for the multiple CPU system

(b) Restrictions

The restrictions when specifying file registers to refresh devices are as follows.

- 1) Refresh cannot be performed correctly if the use of file register which has the same name as the program is specified by the PLC parameter.

When the file register which has the same name as the program is used, refresh is performed to the data of the file register having the same name as the program that is set at the last number in the [Program] tab page of PLC parameter. To read/write the refresh data, specify the file register to the refresh device after switching the file register to the corresponding one with the QDRSET instruction.

- 2) Refresh cannot be performed correctly if the file name of file register or the drive number is changed by the QDRSET instruction.

If the file name of file register or the drive number is changed by the QDRSET instruction, link refresh is performed to the data of the setting file at the time of the END instruction execution. To read/write the refresh data, specify the file register of the setting file at the time of the END instruction execution.

If the drive number is changed by the QDRSET instruction when "ZR" is specified for the device in the CPU modules other than the Universal model QCPU, an error (LINK PARA ERROR (3101)) occurs. (Note that an error does not occur when "R" is specified for the device.)

- 3) When a block number is switched by the RSET instruction, refresh is performed to the data of the file register (R) in the switched block number. When a block number is switched by the RSET instruction, refresh is performed to the data of the file register (R) in the block number at the time of the END instruction execution. To read/write the refresh data, specify the file register of the block number at the time of the END instruction execution.

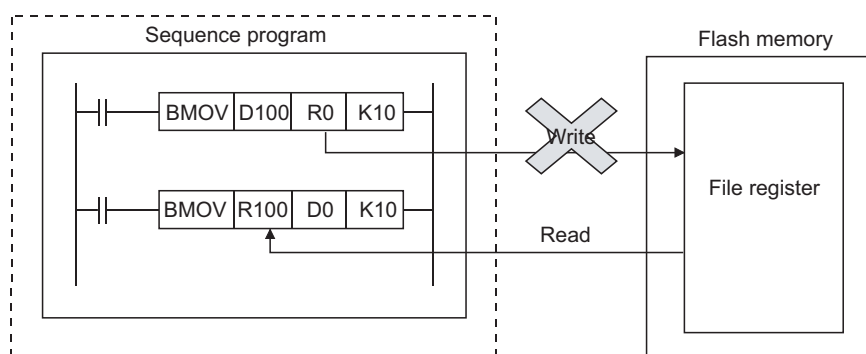
(7) Precautions when file registers in the flash memory are used

This section explains the precautions for use of the flash memory.

- (a) The following flash memory can be used.

- Flash card

- (b) File registers in the flash memory can be only read in a sequence program. (Write to the flash memory cannot be performed in a sequence program.)



When using the flash memory for the file registers, write data in advance.

Using GX Developer, write data to the flash card.





# 4

## HOW TO READ INSTRUCTIONS

---

The description of instructions that are contained in the following chapters are presented in the following format.

1) OUT

2) 5.3 Output Instructions

3) Basic High performance Process Redundant Universal

4) OUT

5) Bit device number (ⓐ)

6) Word device bit designation (ⓑ)

7) Function

8) Operation Error

Setting Data

Setting Data	Internal Devices		R, ZR	U, V, W, X, Y, Z		Zn	Constants	Other DY
	Bit	Word		Bit	Word			
ⓐ	<input type="radio"/> Other than T, C, or F)			<input type="radio"/>			-	<input type="radio"/>

Operation Results

Operation Results	Coil
OFF	OFF
ON	ON

Bit Designation

Operation Results	Bit Designated
OFF	0
ON	1

5-17

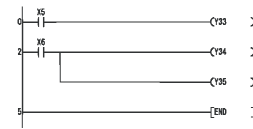
- 1) Code used to write instruction (instruction symbol).
- 2) Section number and general category of instructions described.
- 3) Shows if instructions are enabled or disabled for each CPU module type.

Icon					Meaning
Universal model QCPU	Basic model QCPU	High Performance model QCPU	Process CPU	Redundant CPU	
					A normal icon means the corresponding instruction can be used.
					The icon with Ver. means the instruction can be used with some restrictions (e.g., function version, software version).
					The icon with × (cross) means the corresponding instruction cannot be used.

**Program Example**

9)

(1) When using bit devices  
[Ladder Mode]

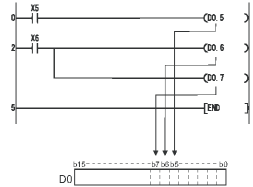


[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	Y33
2	LD	X6
3	OUT	Y34
4	OUT	Y35
5	END	

(2) When bit designation has been made for word device

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	DO.5
2	LD	X6
3	OUT	DO.6
4	OUT	DO.7
5	END	

**Remark**

The number of basic steps for the OUT instructions is as follows:

- When using internal device or file register (R) : 1
- When using direct access output (DY) : 2
- When using serial number access format file register (Only for Universal model QCPU) : 2
- (Other than Universal model QCPU) : 3
- Devices other than above : 3

4) Indicates ladder mode expressions and execution conditions for instructions.

Execution Condition	Non-conditional Execution	Executed while ON	Executed One Time at ON	Executed while OFF	Executed One Time at OFF
Code recorded on description page	No symbol recorded				

5) Indicates the data set for each instruction and the data type.

Data Type	Meaning
Bit	Bit data or head number in bit data
BIN 16 bits	BIN 16-bit data or head number in word device
BIN 32 bits	BIN 32-bit data or head number in double word device
BCD 4-digit	4-digit BCD data
BCD 8-digit	8-digit BCD data
Real number	Floating decimal point data
character string	Character string data
Device name	Device name data

- 6) Devices which can be used by the instruction in question are indicated with circle. The types of devices that can be used are as indicated below:

Setting Data	Internal Devices (System, User)		File Register R, ZR	Link direct device *4 J□□□		Intelligent function module U□□□□	Index register Zn	Constant *5	Others *5
	Bit	Word		Bit	Word				
Applicable devices *1	X, Y, M, L, SM, F, B, SB, FX, FY *2	T, ST, C, *3 D, W, SD, SW, FD, @□	R, ZR	J□□X J□□Y J□□B J□□SB	J□□W J□□SW	U□□□□	Z	K, H, E, \$	P, I, J, U, DX, DY, N, BL, TR, BL \ S, V

\*1: For the description for the individual devices, refer to the QnUCPU User's Manual

(Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals)

\*2: FX and FY can be used only for bit data, and FD only for word data.

\*3: When T, ST and C are used for other than the instructions below, only word data can be used. (Bit data cannot be used.)

[Instructions that can be used with bit data]

LD, LDI, AND, ANI, OR, ORI, LDP, LDF, ANDP, ANDF, ORP, ORF, LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI, OUT, RST

\*4: Usable with the CC-Link IE controller network, MELSECNET/H, and MELSECNET/10.

\*5: Devices which can be set are recorded in the "Constant" and the "Other" columns.

- 7) Indicates the function of the instruction.
- 8) Indicates conditions under which error is returned, and error number. See Section 3.6 for errors not included here.
- 9) Indicates both ladder and list for simple program example. Also indicates the types of individual devices used when the program is executed.

# 5

# SEQUENCE INSTRUCTIONS

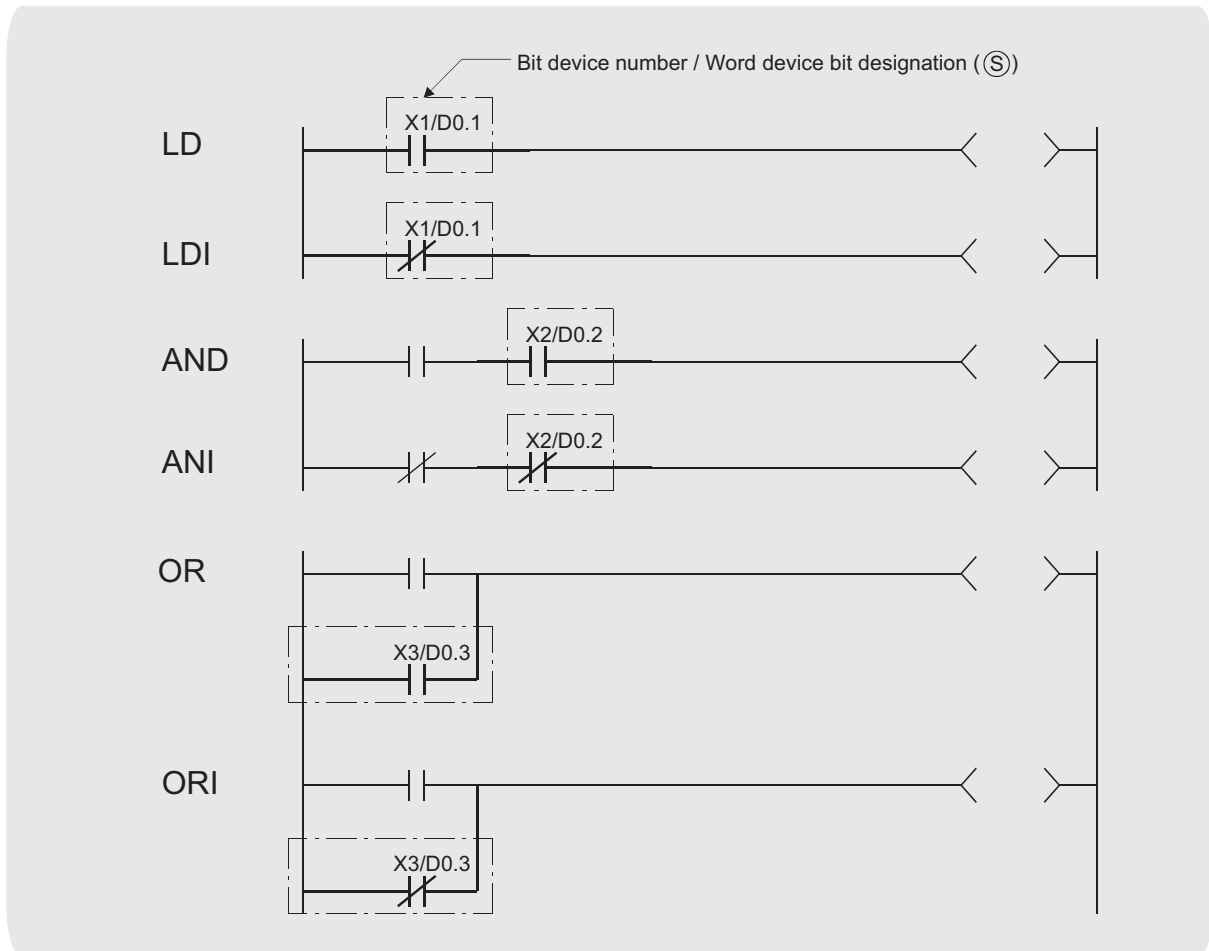
---

Category	Processing Details	Reference Section
Contact instruction	Operation start, series connection, parallel connection	Section 5.1
Association instruction	Ladder block connection, creation of pulses from operation results, store/read operation results	Section 5.2
Output instruction	Bit device output, pulse output, output reversal	Section 5.3
Shift instruction	Bit device shift	Section 5.4
Master control instruction	Master control	Section 5.5
Termination instruction	Program termination	Section 5.6
Other instruction	Program stop, instructions such as no operation which do not fit in the above categories	Section 5.7

## 5.1 Contact Instructions

### 5.1.1 Operation start, series connection, parallel connection (LD,LDI,AND,ANI,OR,ORI)

Basic High performance Process Redundant Universal



Ⓢ : Devices used as contacts (bits)

Setting Data	Internal Devices		R, ZR	JOG		U <sub>0</sub> G <sub>0</sub>	Zn	Constants	Other DX, BL
	Bit	Word		Bit	Word				
Ⓢ				○			—		○

## ★ Function

### LD, LDI

- (1) LD is the A contact operation start instruction, and LDI is the B contact operation start instruction. They read ON/OFF information from the designated device\*<sup>1</sup>, and use that as an operation result.

\*1: When a bit designation is made for a word device, the device turns ON or OFF depending on the 1/0 status of the designated bit.

## AND, ANI

- (1) AND is the A contact series connection instruction, and ANI is the B contact series connection instruction. They read the ON/OFF data of the designated bit device\*<sup>2</sup>, perform an AND operation on that data and the operation result to that point, and take this value as the operation result.
- \*2: When a bit designation is made for a word device, the device turns ON or OFF depending on the 1/0 status of the designated bit.
- (2) There are no restrictions on the use of AND or ANI, but the following applies with a peripheral device used in the ladder mode:
- (a) Write ..... When AND and ANI are connected in series, a ladder with up to 24 stages can be displayed.
- (b) Read ..... When AND and ANI are connected in series, a ladder with up to 24 stages can be displayed. If the number exceeds 24 stages, up to 24 will be displayed.

## OR, ORI

- (1) OR is the A contact single parallel connection instruction, and ORI is the B contact single parallel connection instruction. They read ON/OFF information from the designated device\*<sup>3</sup>, and perform an OR operation with the operation results to that point, and use the resulting value as the operation result.
- \*3: When a bit designation is made for a word device, the device turns ON or OFF depending on the 1/0 status of the designated bit.
- (2) There are no limits on the use of OR or ORI, but the following applies with a peripheral device used in the ladder mode.
- (a) Write ..... OR and ORI can be used to create connections of up to 23 ladders.
- (b) Read ..... OR and ORI can be used to create connections of up to 23 ladders. The 24th or subsequent ladders cannot be displayed properly.

### Remark

Word device bit designations are made in hexadecimal.

Bit b11 of D0 would be D0.0B.

See 3.2.1 for more information on word device bit designation.

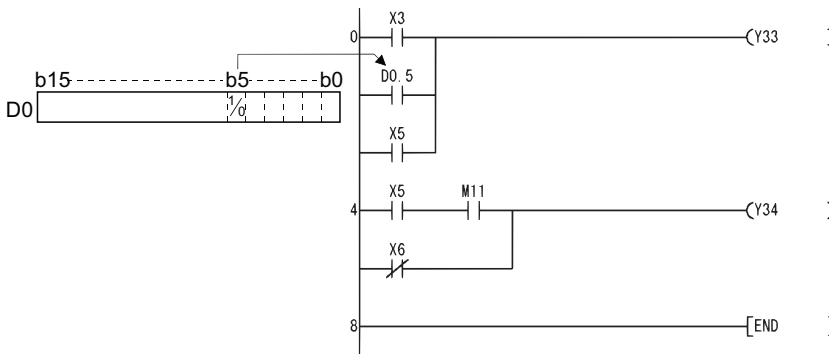
## 🔑 Operation Error

(1) There are no operation errors with LD, LDI, AND, ANI, OR, or ORI instruction.

## 📄 Program Example

(1) A program using the LD, AND, OR, and ORI instructions.

[Ladder Mode]

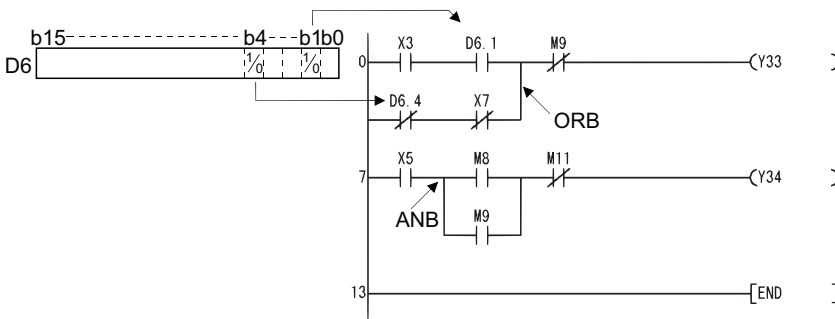


[List Mode]

Step	Instruction	Device
0	LD	X3
1	OR	D0.5
2	OR	X5
3	OUT	Y33
4	LD	X5
5	AND	M11
6	OR I	X6
7	OUT	Y34
8	END	

(2) A program linking contacts using the ANB and ORB instructions.

[Ladder Mode]

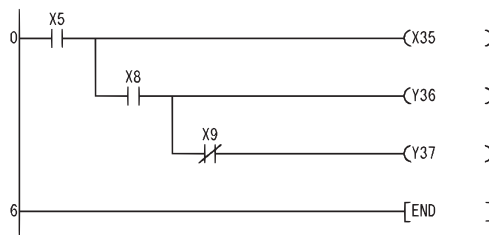


[List Mode]

Step	Instruction	Device
0	LD	X3
1	AND	D6.1
2	LD I	D6.4
3	ANI	X7
4	ORB	
5	ANI	M9
6	OUT	Y33
7	LD	X5
8	LD	M8
9	OR	M9
10	ANB	
11	ANI	M11
12	OUT	Y34
13	END	

(3) A parallel program with the OUT instruction.

[Ladder Mode]



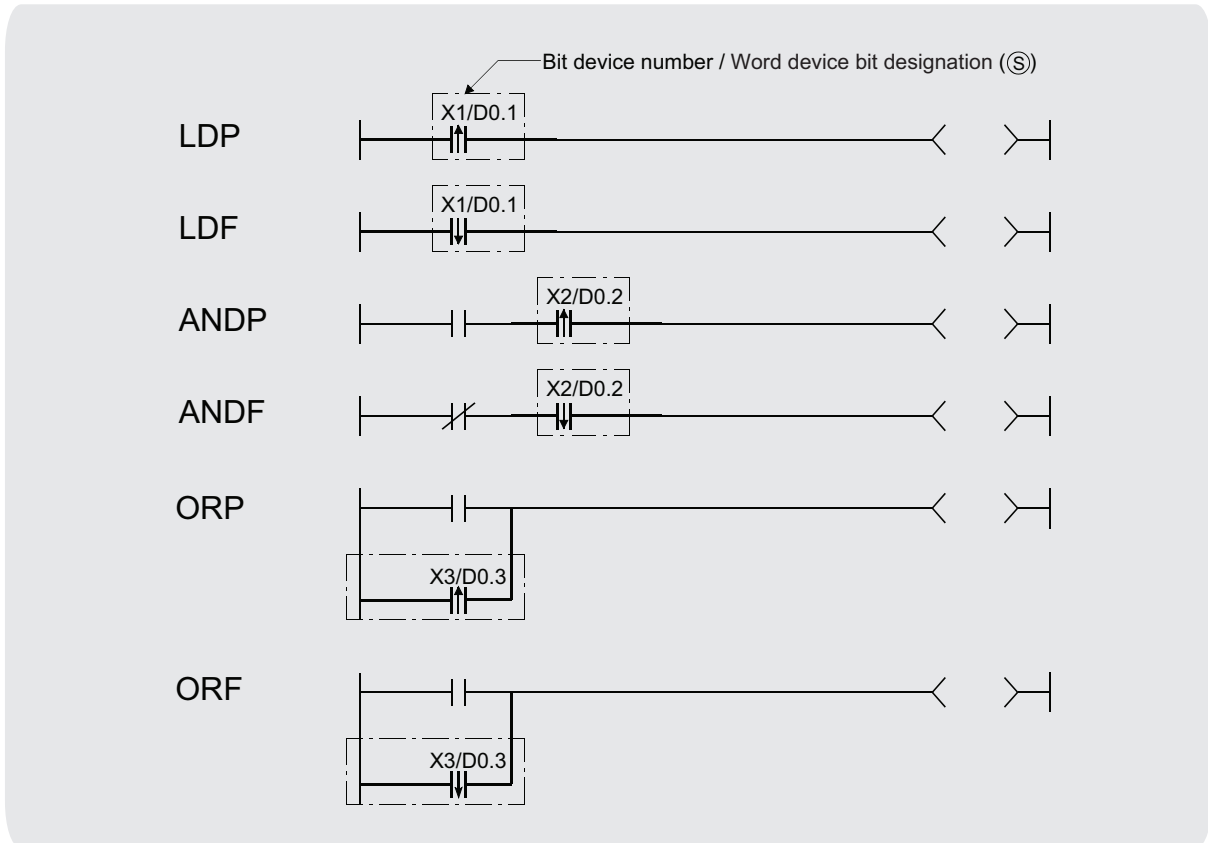
[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	X35
2	AND	X8
3	OUT	Y36
4	ANI	X9
5	OUT	Y37
6	END	



## 5.1.2 Pulse operation start, pulse series connection, pulse parallel connection (LDP,LDF,ANDP,ANDF,ORP,ORF)

Basic High performance Process Redundant Universal



Ⓢ : Devices used as contacts (bits)

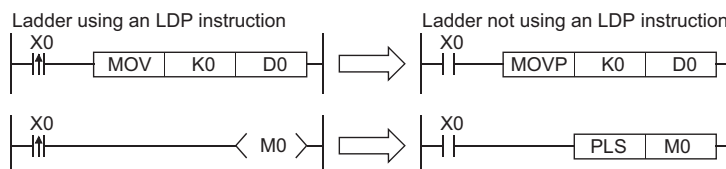
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other DX
	Bit	Word		Bit	Word				
Ⓢ				○				—	○

### ★ Function

#### LDP, LDF

- (1) LDP is the leading edge pulse operation start instruction, and is ON only at the leading edge of the designated bit device (when it goes from OFF to ON). If a word device has been designated, it is ON only when the designated bit changes from 0 to 1.

In cases where there is only an LDP instruction, it acts identically to instructions for the creation of a pulse that are executed during ON(ON(P)).



5

5.1 Contact Instructions  
5.1.2 Pulse operation start, pulse series connection, pulse parallel connection (LDP,LDF,ANDP,ANDF,ORP,ORF)

- (2) LDF is the trailing edge pulse operation start instruction, and is ON only at the trailing edge of the designated bit device (when it goes from ON to OFF).  
If a word device has been designated, it is ON only when the designated bit changes from 1 to 0.

### ANDP, ANDF

- (1) ANDP is a leading edge pulse series connection instruction, and ANDF is a trailing edge pulse series connection instruction. They perform an AND operation with the operation result to that point, and take the resulting value as the operation result.

The ON/OFF data used by ANDP and ANDF are indicated in the table below:

Device Specified in ANDP or ANDF		ANDP State	ANDF State
Bit Device	Bit Designated for Word Device		
OFF to ON	0 to 1	ON	OFF
OFF	0	OFF	
ON	1		ON
ON to OFF	1 to 0		

### ORP, ORF

- (2) ORP is a leading edge pulse parallel connection instruction, and ORF is a trailing edge pulse serial connection instruction. They perform an OR operation with the operation result to that point, and take the resulting value as the operation result.

The ON/OFF data used by ORP and ORF are indicated in the table below:

Device Specified in ORP or ORF		ORP State	ORF State
Bit Device	Bit Designated for Word Device		
OFF to ON	0 to 1	ON	OFF
OFF	0	OFF	
ON	1		ON
ON to OFF	1 to 0		



### Operation Error

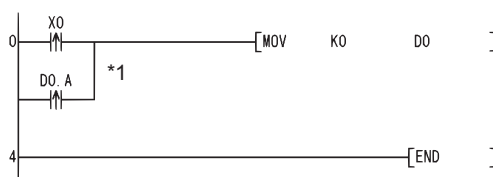
- (1) There are no operation errors with LDP, LDF, ANDP, ANDF, ORP, or ORF instruction.



### Program Example

- (1) The following program executes the MOV instruction at input X0, or at the leading edge of b10 (bit 11) of data register D0.

[Ladder Mode]



[List Mode]

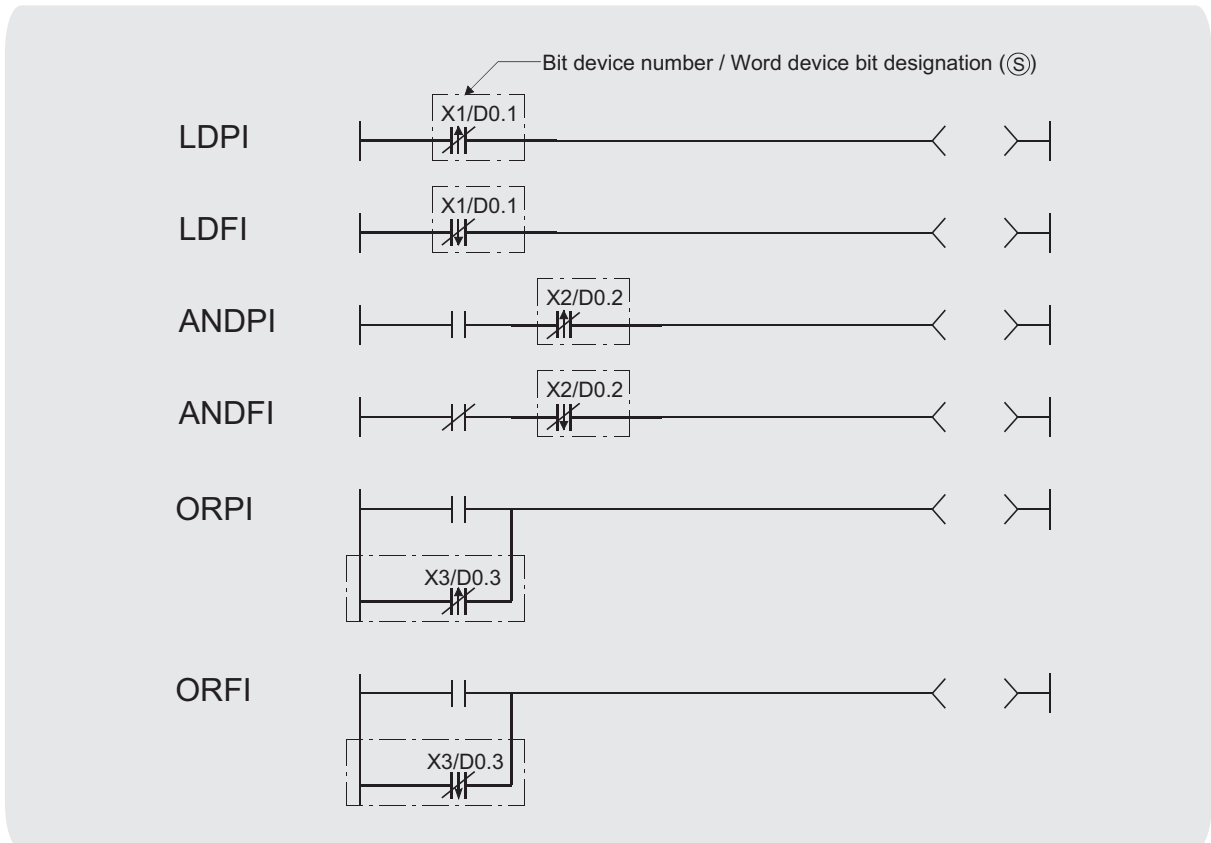
Step	Instruction	Device
0	LDP	X0
1	ORP	D0.A
2	MOV	KO D0
4	END	

\*1: Word device bit designation is performed in hexadecimal. Bit b10 of D0 will be D0.A.

### 5.1.3 Pulse NOT operation start, pulse NOT series connection, pulse NOT parallel connection (LDPI,LDFI,ANDPI,ANDFI,ORPI,ORFI)



- QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.
- QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ : Devices used as contacts (bits)

Setting Data	Internal Devices		R, ZR	J:AND		U:AND	Zn	Constants	Other DX
	Bit	Word		Bit	Word				
Ⓢ				○				-	○

## ★ Function

### LDPI, LDFI

- (1) LDPI is the leading edge pulse NOT operation start instruction that is on only at the leading edge of the specified bit device (when the bit device goes from on to off) or when the bit device is on or off. If a word device has been specified, LDPI is on only when the specified bit is 0, 1, or changes from 1 to 0.

- (2) LDFI is the trailing edge pulse NOT operation start instruction that is on only at the trailing edge of the specified bit device (when the bit device goes from off to on) or when the bit device is on or off. If a word device has been specified, LDFI is on only when the specified bit is 0, 1, or changes from 0 to 1.

Device Specified in LDPI or LDFI		LDPI State	LDFI State
Bit Device	Bit Designated for Word Device		
OFF to ON	0 to 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON to OFF	1 to 0	ON	OFF

## ANDPI, ANDFI

- (1) ANDPI is a leading edge pulse NOT series connection, and ANDFI is a trailing pulse NOT series connection. ANDPI and ANDFI execute an AND operation with the previous operation result, and take the resulting value as the operation result.

The on or off data used by ANDPI and ANDFI are indicated in the table below.

Device Specified in ANDPI or ANDFI		LDPI State	LDFI State
Bit Device	Bit Designated for Word Device		
OFF to ON	0 to 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON to OFF	1 to 0	ON	OFF

## ORPI, ORFI

- (1) ORPI is a leading edge pulse NOT parallel connection, and ORFI is a trailing pulse NOT parallel connection. ORPI and ORFI execute an OR operation with the previous operation result, and take the resulting value as the operation result.

The on or off data used by ORPI and ORFI are indicated in the table below.

Device Specified in ORPI or ORFI		ORPI State	ORFI State
Bit Device	Bit Designated for Word Device		
OFF to ON	0 to 1	OFF	ON
OFF	0	ON	ON
ON	1	ON	ON
ON to OFF	1 to 0	ON	OFF



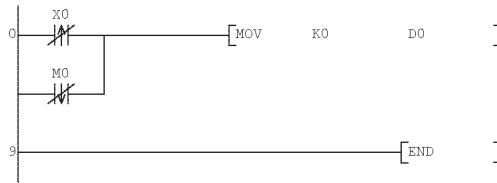
## Operation Error

- (1) There are no operation errors with LDPI, LDFI, ANDPI, ANDFI, ORPI, or ORFI instruction.

## Program Example

- (1) The following program stores 0 into D0 when X0 is on, off, or turns from on to off, or M0 is on, off, or turns from off to on.

[Ladder Mode]

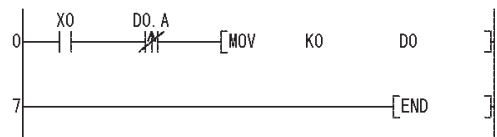


[List Mode]

Step	Instruction	Device
0	LDPI	X0
3	ORFI	M0
7	MOV	K0 D0
9	END	

- (2) The following program stores 0 into D0 when X0 is on and b10 (bit 11) of D0 is on, off, or turns from on to off.

Ladder Mode]



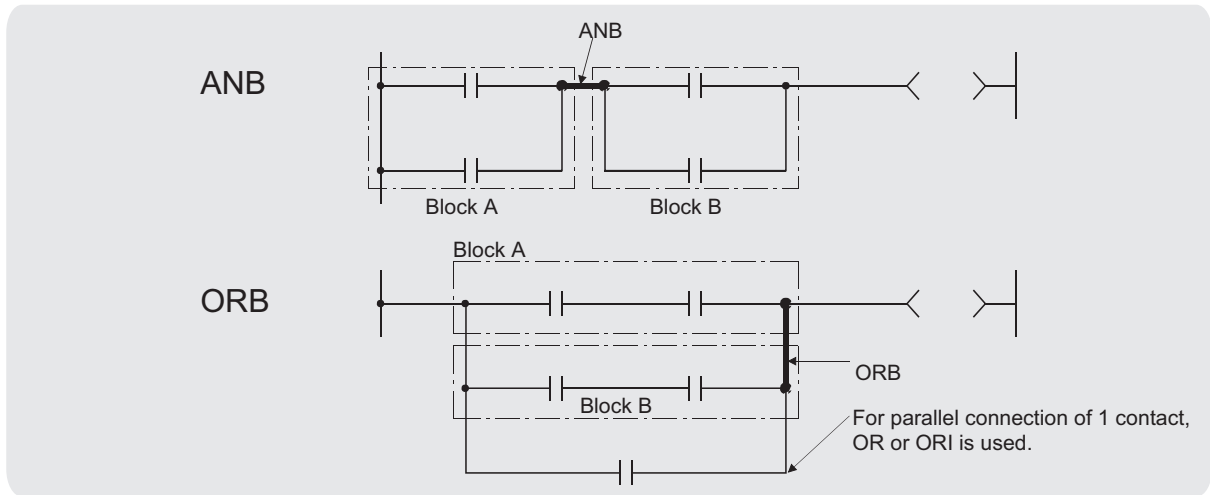
[List Mode]

Step	Instruction	Device
0	LD	X0
1	ANDPI	D0.A
5	MOV	K0 D0
7	END	

## 5.2 Association Instructions

### 5.2.1 Ladder block series connection and parallel connection (ANB,ORB)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

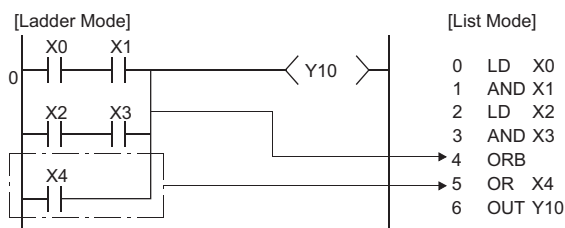
## ★ Function

### ANB

- (1) Performs an AND operation on block A and block B, and takes the resulting value as the operation result.
- (2) The symbol for ANB is not the contact symbol, but rather is the connection symbol.
- (3) When programming in the list mode, up to 15 ANB instructions (16 blocks) can be written consecutively.

### ORB

- (1) Conducts an OR operations on Block A and Block B, and takes the resulting value as the operation result.
- (2) ORB is used to perform parallel connections for ladder blocks with two or more contacts. For ladder blocks with only one contact, use OR or ORI; there is no need for ORB in such cases.



- (3) The ORB symbol is not the contact symbol, but rather is the connection symbol.
- (4) When programming in the list mode, it is possible to use up to 15 ORB instructions successively (16 blocks).



# Operation Error

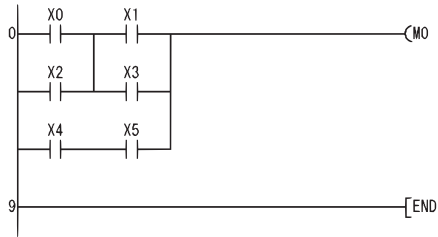
(1) There are no operation errors associated with ANB or ORB instruction.



# Program Example

(1) A program using the ANB and ORB instructions.

[Ladder Mode]

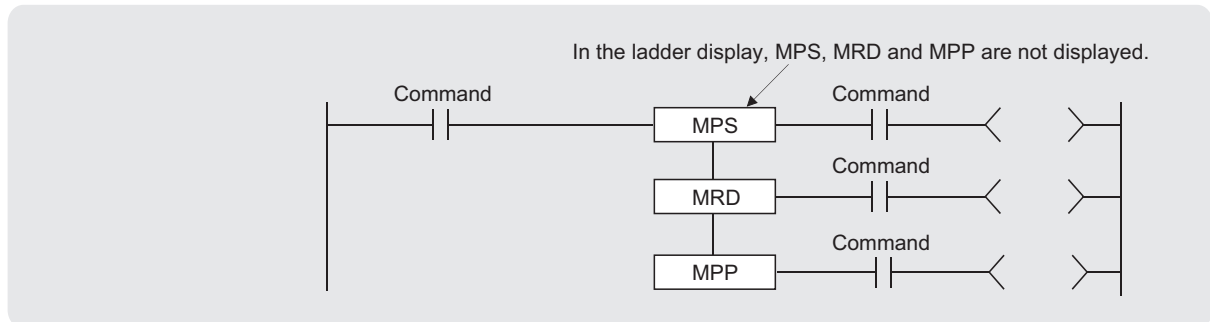


[List Mode]

Step	Instruction	Device
0	LD	X0
1	OR	X2
2	LD	X1
3	OR	X3
4	ANB	
5	LD	X4
6	AND	X5
7	ORB	
8	OUT	M0
9	END	

## 5.2.2 Operation results push,read,pop (MPS,MRD,MPP)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	JOG		U <sub>0</sub> G <sub>0</sub>	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

#### MPS

- (1) Stores the memory of the operation result (ON or OFF) immediately prior to the MPS instruction.
- (2) Up to 16 MPS instructions can be used successively.  
If the MPP instruction is used during this process, the number of uses calculated for the MPS instruction will be decremented by one.

#### MRD

- (1) Reads the operation result stored for the MPS instruction, and uses that result to perform the operation in the next step.

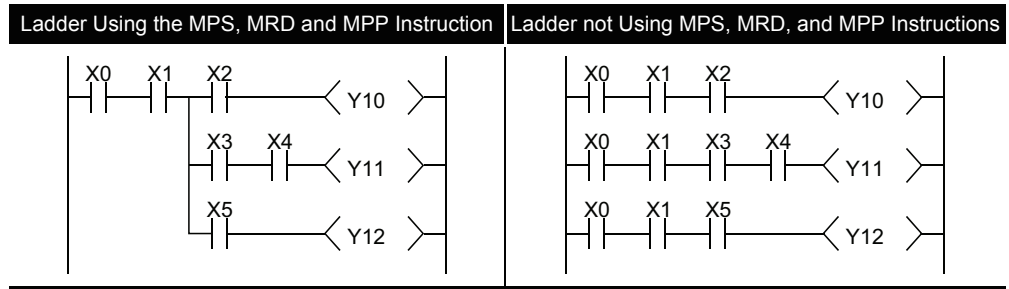
#### MPP

- (1) Reads the operation result stored for the MPS instruction, and uses that result to perform the operation in the next step.
- (2) Clears the operation results stored by the MPS instruction.
- (3) Subtracts 1 from the number of MPS instruction times of use.



**POINT**

1. The following shows ladders both using and not using the MPS, MRD, and MPP instructions.



2. The MPS and MPP instructions must be used the same number of times. Failure to observe this will not correctly display the ladder in the ladder mode of the peripheral device.

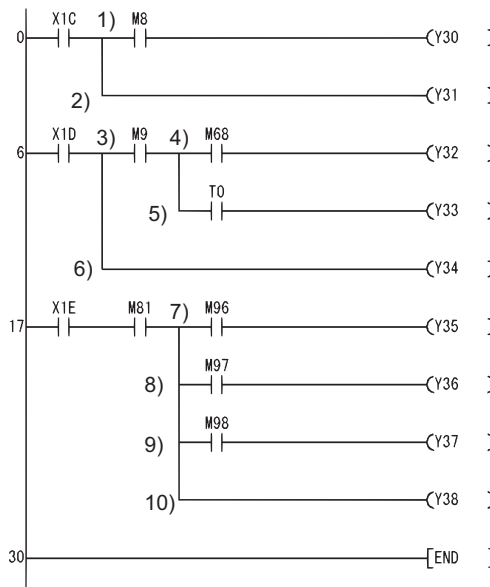
**! Operation Error**

(1) There are no errors associated with the MPS, MRD, or MPP instruction.

**Program Example**

(1) A program using the MPS, MRD, and MPP instructions.

[Ladder Mode]

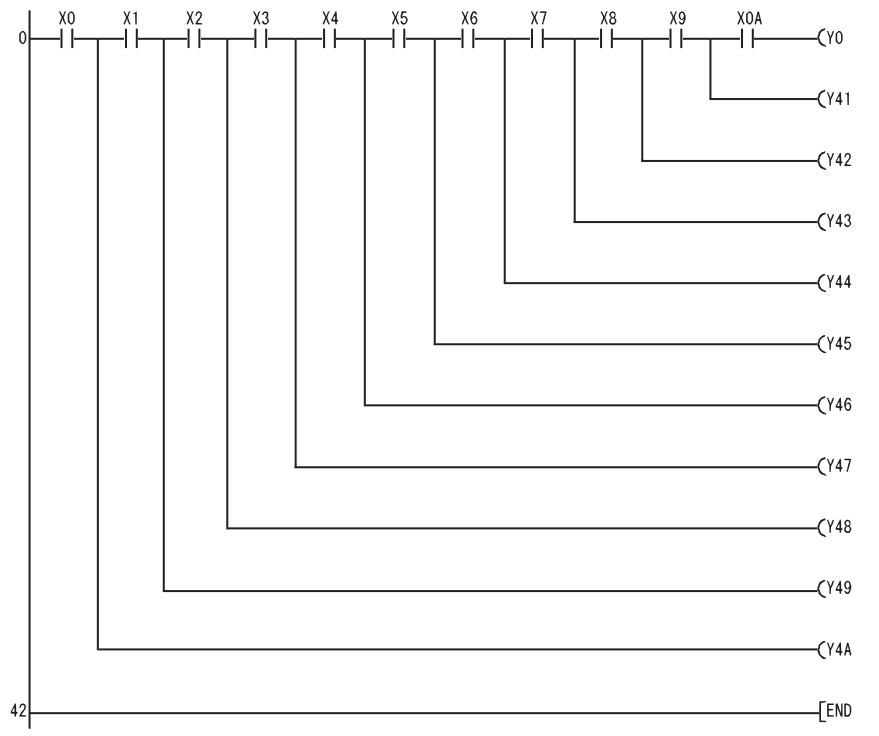


[List Mode]

Step	Instruction	Device
0	LD	X1C
1	MPS	
2	AND	M8
3	OUT	Y30
4	MPP	
5	OUT	Y31
6	LD	X1D
7	MPS	
8	AND	M9
9	MPS	
10	AND	M68
11	OUT	Y32
12	MPP	
13	AND	TO
14	OUT	Y33
15	MPP	
16	OUT	Y34
17	LD	X1E
18	AND	M81
19	MPS	
20	AND	M96
21	OUT	Y35
22	MRD	
23	AND	M97
24	OUT	Y36
25	MRD	
26	AND	M98
27	OUT	Y37
28	MPP	
29	OUT	Y38
30	END	

(2) A program using the MPS and MPP instructions successively.

[Ladder Mode]

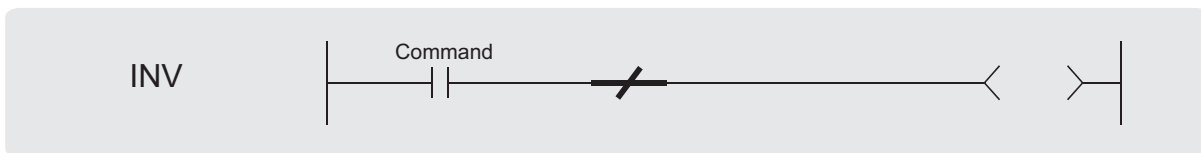


[List Mode]

Step	Instruction	Device
0	LD	X0
1	MPS	
2	AND	X1
3	MPS	
4	AND	X2
5	MPS	
6	AND	X3
7	MPS	
8	AND	X4
9	MPS	
10	AND	X5
11	MPS	
12	AND	X6
13	MPS	
14	AND	X7
15	MPS	
16	AND	X8
17	MPS	
18	AND	X9
19	MPS	
20	AND	X0A
21	OUT	Y0
22	MPP	
23	OUT	Y41
24	MPP	
25	OUT	Y42
26	MPP	
27	OUT	Y43
28	MPP	
29	OUT	Y44
30	MPP	
31	OUT	Y45
32	MPP	
33	OUT	Y46
34	MPP	
35	OUT	Y47
36	MPP	
37	OUT	Y48
38	MPP	
39	OUT	Y49
40	MPP	
41	OUT	Y4A
42	END	

## 5.2.3 Operation results inversion (INV)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	J:Q:G		U:Q:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

5

### ★ Function

Inverts the operation result immediately prior to the INV instruction.

Operation Result Immediately Prior to the INV Instruction	Operation Result Following the Execution of the INV Instruction
OFF	ON
ON	OFF

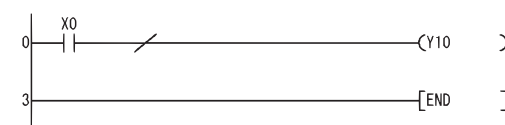
### ! Operation Error

- (1) There are no operation errors associated with the INV instruction.

### 📄 Program Example

- (1) A program which inverts the X0 ON/OFF data, and outputs from Y10.

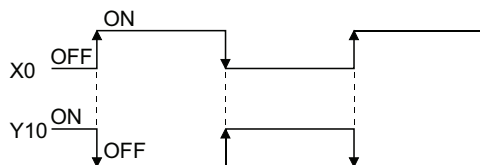
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	INV	
2	OUT	Y10
3	END	

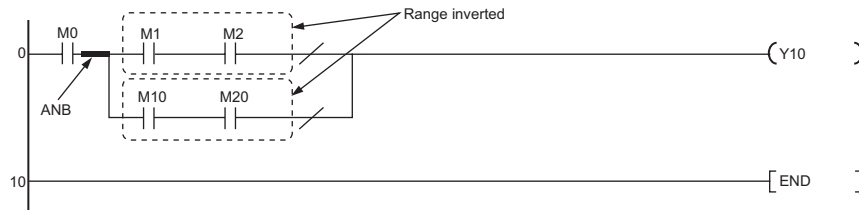
[Timing Chart]



5.2 Association Instructions  
5.2.3 Operation results inversion (INV)

**POINT**

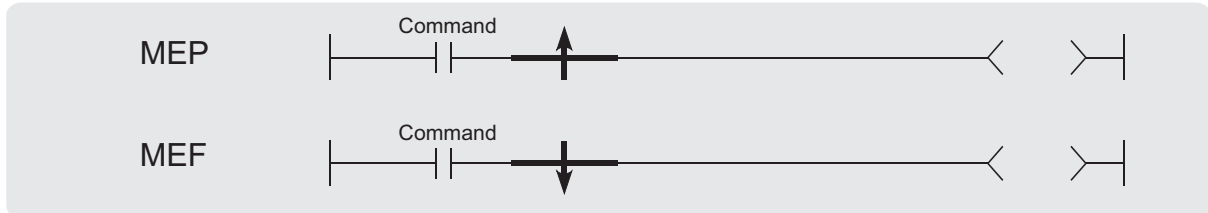
1. The INV instruction operates based on the results of calculation made until the INV instruction is given. Accordingly, use it in the same position as that of the AND instruction.  
The INV instruction cannot be used at the LD and OR positions.
2. When a ladder block is used, the operation result is inverted within the range of the ladder block. To operate a ladder using the INV instruction in combination with the ANB instruction, pay attention to the range that will be inverted.



For details of the ANB instruction, refer to Section 5.2.1

## 5.2.4 Operation result conversions (MEP,MEF)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	JOG		U/G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

5

### ★ Function

#### MEP

- (1) If operation results up to the MEP instruction are leading edge (from OFF to ON), goes ON (continuity status).  
If operation results up to the MEP instruction are anything other than leading edge, goes OFF (non-continuity status).
- (2) Use of the MEP instruction simplifies pulse conversion processing when multiple contacts are connected in series.

#### MEF

- (1) If operation results up to the MEF instruction are trailing edge (from ON to OFF), goes ON (continuity status).  
If operation results up to the MEF instruction are anything other than trailing edge, goes OFF (non-continuity status).
- (2) Use of the MEF instruction simplifies pulse conversion processing when multiple contacts are connected in series.

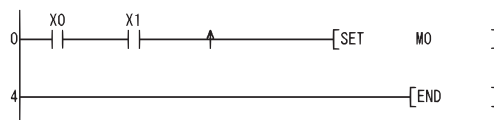
### ! Operation Error

- (1) There are no operation errors associated with the MEP or MEF instruction.

### 📄 Program Example

- (1) A program which performs pulse conversion to the operation results of X0 and X1

[Ladder Mode]



[List Mode]

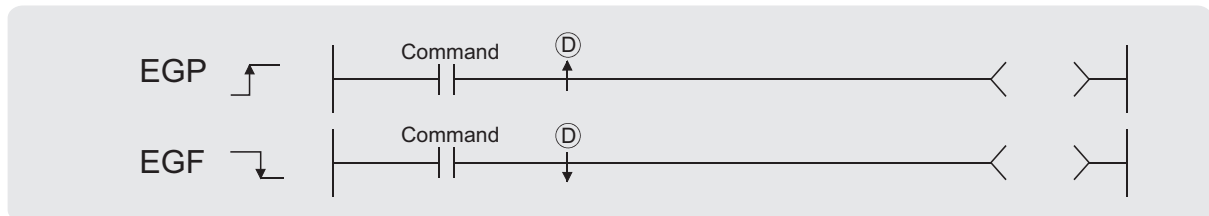
Step	Instruction	Device
0	LD	X0
1	AND	X1
2	MEP	
3	SET	M0
4	END	

### ☒ POINT

1. The MEP and MEF instructions will occasionally not function properly when pulse conversion is conducted for a contact that has been indexed by a subroutine program or by the FOR to NEXT instructions. If pulse conversion is to be conducted for a contact that has been indexed by a subroutine program or by the FOR to NEXT instructions, use the EGP/EGF instructions.
2. Because the MEP and MEF instructions operate with the operation results immediately prior to the MEP and MEF instructions, the AND instruction should be used at the same position. The MEP and MEF instructions cannot be used at the LD or OR position.

## 5.2.5 Pulse conversions of edge relay operation results (EGP,EGF)

Basic High performance Process Redundant Universal



Ⓧ: Edge relay number where operation results are stored (bits)

Setting Data	Internal Devices		R, ZR	J:GO		U:GO	Zn	Constants	Other V
	Bit	Word		Bit	Word				
Ⓧ									○

### ★ Function

#### EGP

- Operation results up to the EGP instruction are stored in memory by the edge relay (V).
- Goes ON (continuity status) at the leading edge (OFF to ON) of the operation result up to the EGP instruction.  
If the operation result up to the EGP instruction is other than a leading edge (i.e., from ON to ON, ON to OFF, or OFF to OFF), it goes OFF (non-continuity status).
- The EGP instruction is used for subroutine programs, and for conducting pulse operations for programs designated by indexing between the FOR and NEXT instructions.
- The EGP instruction can be used like an AND instruction.

#### EGF

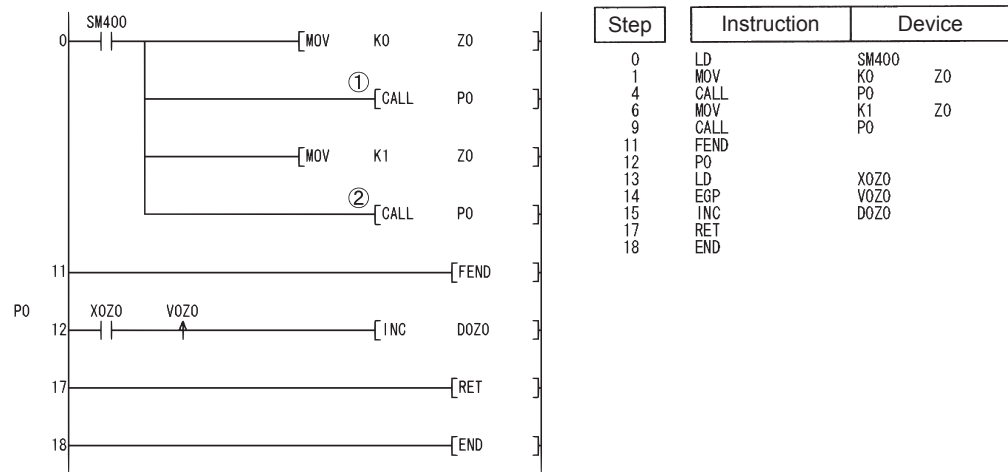
- Operation results up to the EGF instruction are stored in memory by the edge relay (V).
- Goes ON at the trailing edge (from ON to OFF) of the operation result up to the EGF instruction.  
If the operation result up to the EGF instruction is other than a trailing edge (i.e., from OFF to ON, ON to ON, or OFF to OFF), it goes OFF (non-continuity status).
- The EGF instruction is used for subroutine programs, and for conducting pulse operations for programs designated by indexing between the FOR and NEXT instructions.
- The EGF instruction can be used like an AND instruction.

### ! Operation Error

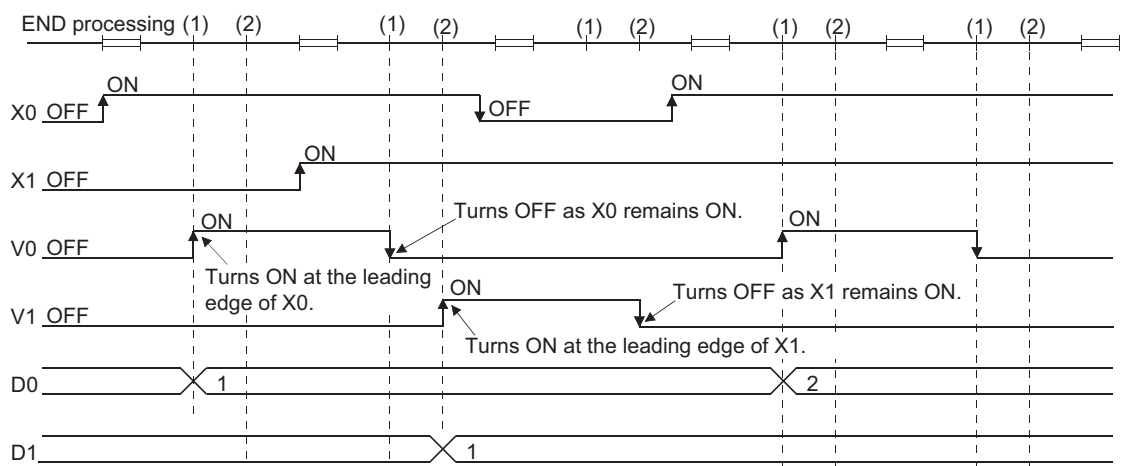
- There are no operation errors associated with the EGP or EGF instruction.

## Program Example

- (1) A program using the EGP instruction in the subroutine program using the EGD instruction  
 [Ladder Mode] [List Mode]

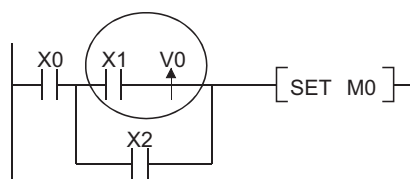


### [Operation]



### POINT

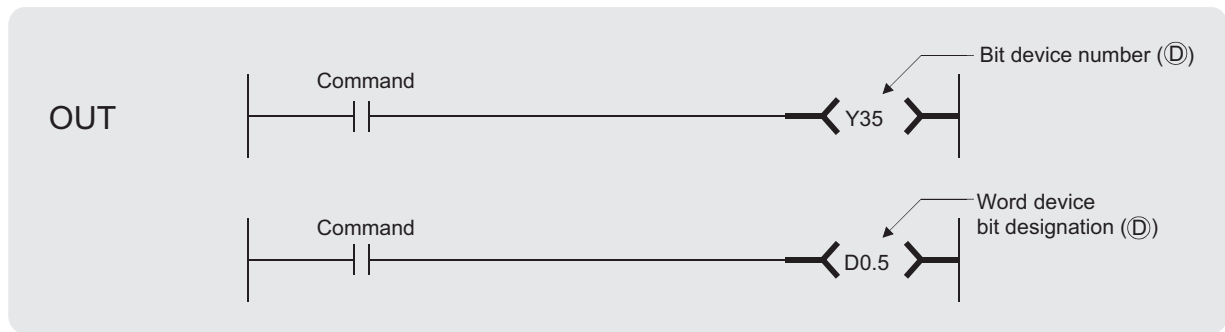
- Since the EGP and EGF instructions are executed according to the operation results performed immediately before the EGP/EGF instructions, these instructions must be used at the same position as the AND instruction. (Refer to Section 5.1.1.)  
 The EGP and EGF instruction cannot be used at the position of the LD or OR instruction.
- EGP and EGF instructions cannot be used at the ladder block positions shown below.



## 5.3 Output Instructions

### 5.3.1 Out instruction (excluding timers, counters, and annunciators) (OUT)

Basic High performance Process Redundant Universal



Ⓧ : Number of the device to be turned ON and OFF (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other DY
	Bit	Word		Bit	Word				
Ⓧ	○ (Other than T, C, or F)			○			—		○

### ★ Function

(1) Operation results up to the OUT instruction are output to the designated device.

(a) When Using Bit Devices

Operation Results	Coil
OFF	OFF
ON	ON

(b) When Bit Designation has been Made for Word Device

Operation Results	Bit Designated
OFF	0
ON	1

### ! Operation Error

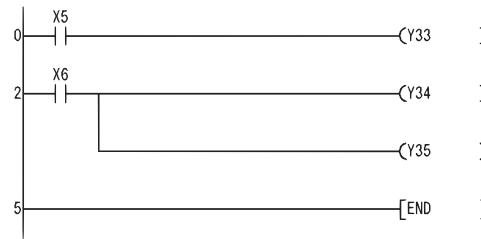
(1) There are no operation errors associated with the OUT instruction.



# Program Example

(1) When using bit devices

[Ladder Mode]

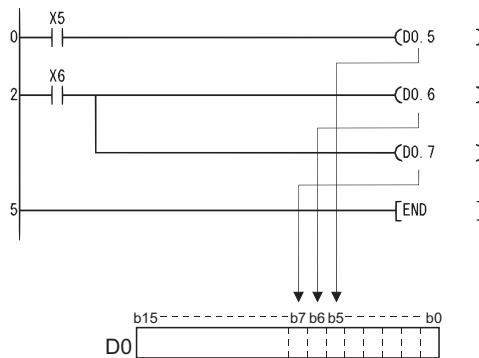


[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	Y33
2	LD	X6
3	OUT	Y34
4	OUT	Y35
5	END	

(2) When bit designation has been made for word device

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X5
1	OUT	D0.5
2	LD	X6
3	OUT	D0.6
4	OUT	D0.7
5	END	

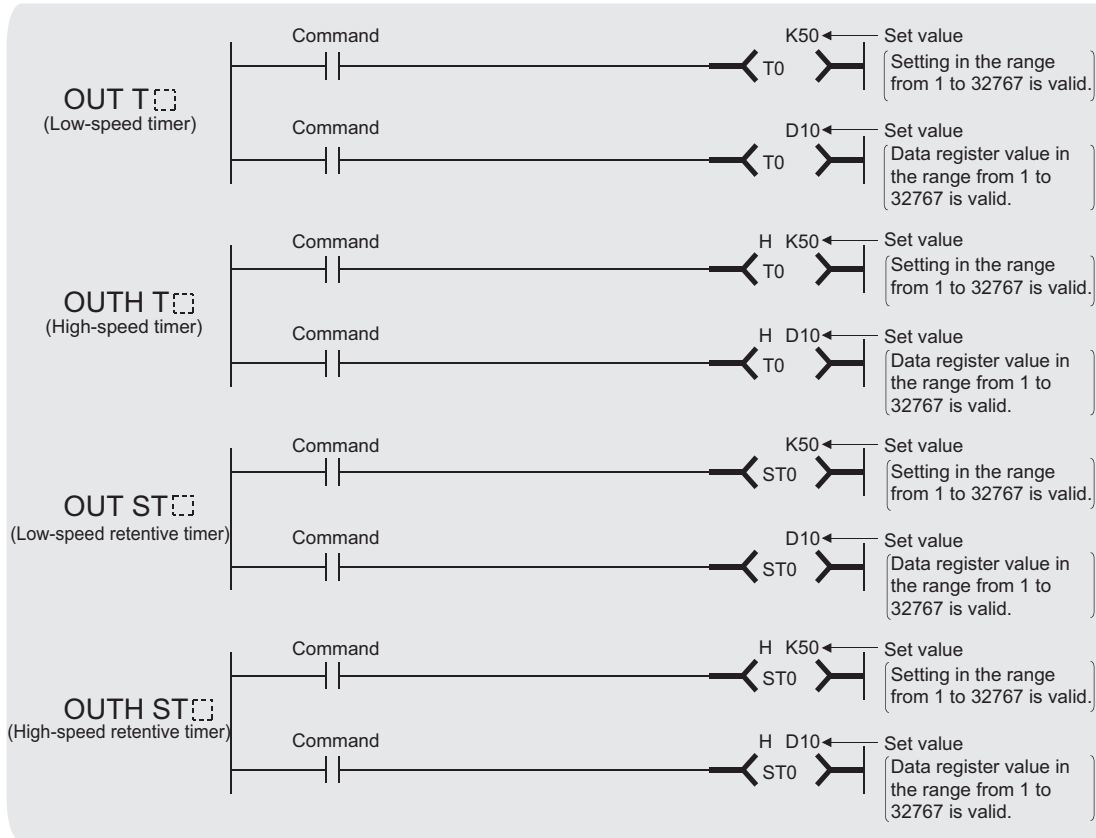
**Remark**

The number of basic steps for the OUT instructions is as follows:

- When using internal device or file register (R) : 1
- When using direct access output (DY) : 2
- When using serial number access format file register (Only for Universal model QCPU) : 2
- (Other than Universal model QCPU) : 3
- Devices other than above : 3

## 5.3.2 Timers (OUT T,OUTH T)

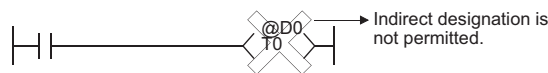
Basic High performance Process Redundant Universal



Ⓧ : Timer number (bit)  
 Set value : Value set for timer (BIN 16 bits \*1)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K	Other
	Bit	Word		Bit	Word				
Ⓧ	○ (Only T)	—	—	—	—	—	—	—	—
Set value	—	○ (Other than T, C)	○	—	○	—	—	○ *2	—

\*1: The value setting for the timer cannot be designated indirectly.



See Section 3.4 for further information on indirect designation.

\*2: Timer values can be set only as a decimal constant (K). Hexadecimal constants (H) and real numbers cannot be used for timer settings.

### ★ Function

- (1) When the operation results up to the OUT instruction are ON, the timer coil goes ON and the timer counts up to the value that has been set; when the time up status (total numeric value is equal to or greater than the setting value), the contact responds as follows:

A Contact	Continuity
B Contact	Non-continuity

- (2) The contact responds as follows when the operation result up to the OUT instruction is a change from ON to OFF:

Type of Timer	Timer Coil	Present Value of Timer	Prior to Time Up		After Time Up	
			A Contact	B Contact	A Contact	B Contact
Low speed timer	OFF	0	Non-continuity	Continuity	Non-continuity	Continuity
High speed timer						
Low speed retentive timer	OFF	Maintains the present value	Non-continuity	Continuity	Continuity	Non-continuity
High speed retentive timer						

- (3) To clear the present value of a retentive timer and turn the contact OFF after time up, use the RST instruction.
- (4) A negative number (−32768 to −1) cannot be set as the setting value for the timer.\*<sup>3</sup>  
If the setting value is 0, the timer will time out when the time the OUT instruction is executed.
- \*3: When specifying a setting value for the timer using a word device (D, W, R, ZR, J or U), whether the value is in the setting range is not checked. Check the value in the user program so that a negative number is not set.
- (5) The following processing is conducted when the OUT instruction is executed:
- OUT T coil turned ON or OFF
  - OUT T contact turned ON or OFF
  - OUT T present value updated
- In cases where a JMP instruction or the like is used to jump to an OUT T instruction while the OUT T instruction is ON, no present value update or contact ON/OFF operation is conducted.
- Also, if the same OUT T instruction is conducted two or more times during the same scan, the present value of the number of repetitions executed will be updated.
- (6) Indexing for timer coils or contacts can be conducted only by Z0 or Z1.  
Timer setting value has no limitation for indexing.

### Remark

#### 1. Timer's time limit

Time limit of the timer is set in the PLC system of the PLC parameter dialog box.

Type of Timer	QCPU	
	Setting Range	Setting Unit
Low speed timer	1 ms to 1000 ms (Default: 100 ms)	1 ms
Low speed retentive timer		
High speed timer	0.1 ms to 100.0 ms (Default: 10.0 ms)	0.1 ms
High speed retentive timer		

2. For information on timer counting methods, User's Manual (Functions Explanation, Program Fundamentals) for the CPU module used.
3. The number of basic steps of the OUT C instruction is 4.

## Operation Error

- (1) There are no operation errors associated with the OUT T instruction.

 **Caution**

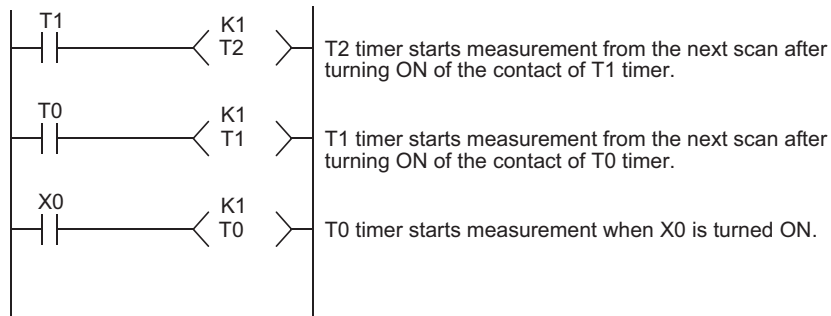
- (1) When creating a program in which the operation the timer contact triggers the operation of other timer, create the program for the timer that operates later first.

In the following cases, all timers go ON at the same scan if the program is created in the order the timers operate.

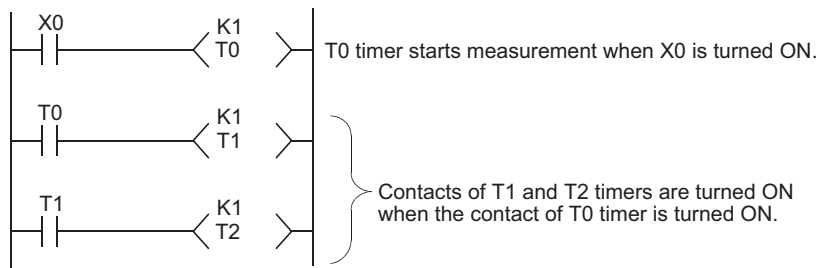
- If the set value is smaller than a scan time.
- If "1" is set

**Example**

- For timers T0 to T2, the program is created in the order the timer operates later.



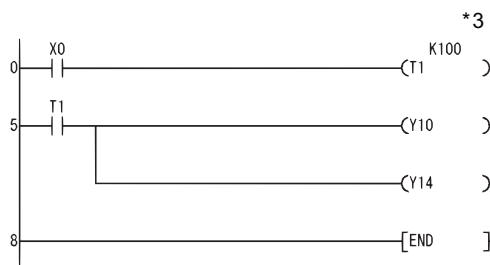
- For timers T0 to T2, the program is created in the order of timer operation.



# Program Example

(1) The following program turns Y10 and Y14 ON 10 seconds after X0 has gone ON.

[Ladder Mode]



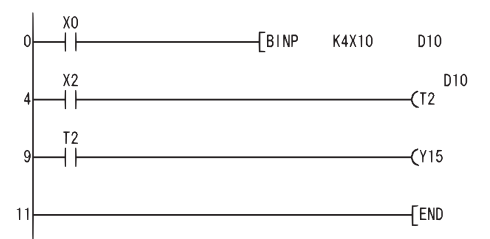
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	T1 K100
5	LD	T1
6	OUT	Y10
7	OUT	Y14
8	END	

\*3: The setting value of the low-speed timer indicates its default time limit (100 ms).

(2) The following program uses the BCD data at X10 to X1F as the timer's set value.

[Ladder Mode]



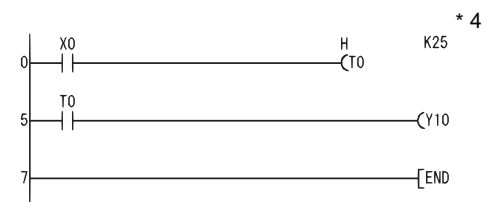
Converts the BCD data at X10 to X1F to BIN and stores the converted value at D10.  
When X2 is turned ON, T2 starts measurement using the data stored in D10 as the set value.  
Y15 goes ON at the count-up of T2.

[List Mode]

Step	Instruction	Device
0	LD	X0
1	B1NP	K4X10 D10
4	LD	X2
5	OUT	T2 D10
9	LD	T2
10	OUT	Y15
11	END	

(3) The following program turns Y10 ON 250 ms after X0 goes ON.

[Ladder Mode]



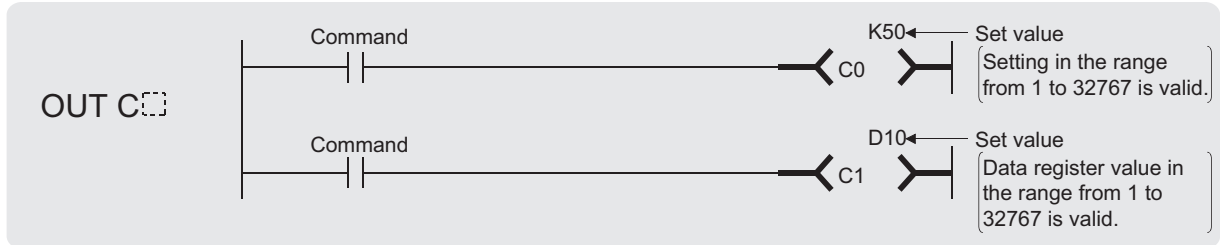
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUTH	T0 K25
5	LD	T0
6	OUT	Y10
7	END	

\*4: The setting value of the high-speed timer indicates its default time limit (10 ms).

### 5.3.3 Counter (OUT C)

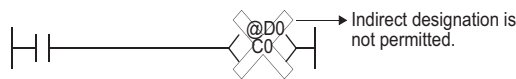
Basic High performance Process Redundant Universal



ⓐ : Counter number (bits)  
Set value : Counter setting value (BIN 16 bits \*1)

Setting Data	Internal Devices		R, ZR	Jm		U, G, O	Zn	Constants K	Other
	Bit	Word		Bit	Word				
ⓐ	○ (Only C)	—	—	—	—	—	—	—	—
Set value	—	○ (Other than T, C)	○	—	○	—	—	○ *2	—

\*1: Counter value cannot be set by indirect designation.



See Section 3.4 for further information on indirect designation.

\*2: Counter value can be set only with a decimal constant (K). A hexadecimal constant (H) or a real number cannot be used for the counter value setting.

### ★ Function

- (1) When the operation results up to the OUT instruction change from OFF to ON, 1 is added to the present value (count value) and the count up status (present value ≥ set value), and the contacts respond as follows:

A Contact	Continuity
B Contact	Non-continuity

- (2) No count is conducted with the operation results at ON. (There is no need to perform pulse conversion on count input.)
- (3) After the count up status is reached, there is no change in the count value or the contacts until the RST instruction is executed.
- (4) A negative number (−32768 to −1) cannot be set as the setting value for the timer. If the set value is 0, the processing is identical to that which takes place for 1.
- (5) Indexing for the counter coil and contact can use only Z0 and Z1. Counter setting value has no limitation for indexing.

#### Remark

1. For counter counting methods, refer to the User's Manual (Functions Explanation, Program Fundamentals) for the CPU module used.
2. The number of basic steps of the OUT C instruction is 4.

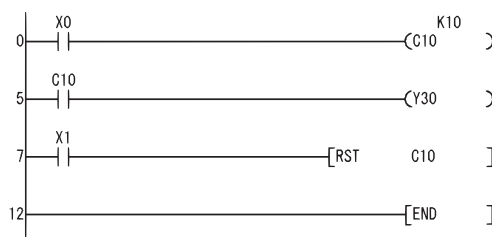
## ! Operation Error

- (1) There are no operation errors associated with the OUT C instruction.

## Program Example

- (1) The following program turns Y30 ON after X0 has gone ON 10 times, and resets the counter when X1 goes ON.

[Ladder Mode]

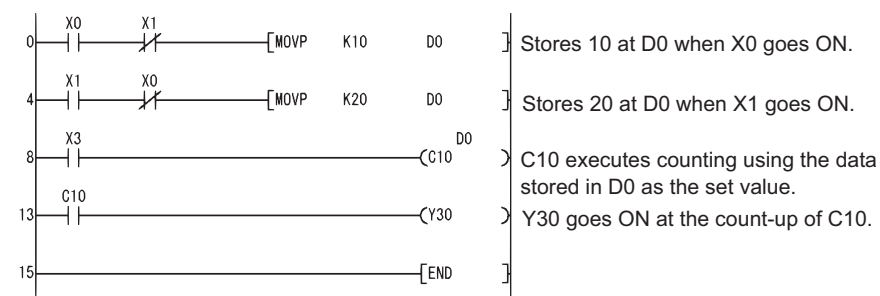


[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	C10 K10
5	LD	C10
6	OUT	Y30
7	LD	X1
8	RST	C10
12	END	

- (2) The following program sets the value for C10 at 10 when X0 goes ON, and at 20 when X1 goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ANI	X1
2	MOV P	K10 D0
4	LD	X1
5	ANI	X0
6	MOV P	K20 D0
8	LD	X3
9	OUT	C10 D0
13	LD	C10
14	OUT	Y30
15	END	

## 5.3.4 Annunciator output (OUT F)

Basic High performance Process Redundant Universal



ⓐ : Number of the annunciator to be turned ON (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
ⓐ	<input type="radio"/> (Only F)					—			

### ★ Function

- (1) Operation results up to the OUT instruction are output to the designated annunciator.
- (2) The following responses occur when an annunciator (F) is turned ON.
  - The "USER"/"ERR." LED goes ON.
  - The annunciator numbers which are ON (F numbers) are stored in special registers (SD64 to SD79).
  - The value of SD63 is incremented by 1.
- (3) If the value of SD63 is 16 (which happens when 16 annunciators are already ON), even if a new annunciator is turned ON, its number will not be stored at SD64 to SD79.
- (4) The following responses occur when the annunciator is turned OFF by the OUT instruction.
 

The coil goes OFF, but there are no changes in the status of the "USER" / "ERR." LED and the contents of the values stored in SD63 to SD79.

Use the RST F instruction to make the "USER"/"ERR." LED go OFF as well as to delete the annunciator which was turned OFF by the OUT F instruction from SD63 to SD79.



## ! Operation Error

- (1) There are no operation errors associated with the OUT F instruction.

### Remark

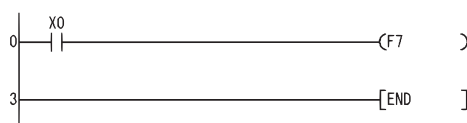
1. For details of annunciators, refer to the User's Manual (Functions Explanation, Program Fundamentals) for the CPU module used.
2. The number of basic steps for the OUT module F instruction is 2.
3. The table below shows which CPU module features either the LED display device on front of the CPU module or "USER" LED.

Type of LED	CPU Module Type Name
"USER" LED	High Performance model QCPU, Process CPU, Redundant CPU, Universal model QCPU
"ERR." LED	Basic model QCPU

## Program Example

- (1) The following program turns F7 ON when X0 goes ON, and stores the value 7 from SD64 to SD79.

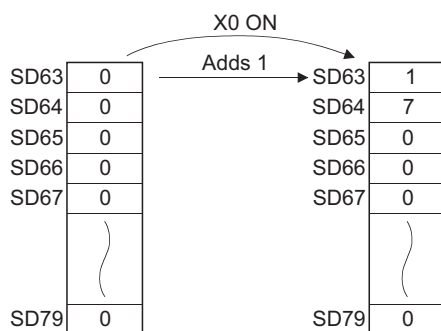
[Ladder Mode]



[List Mode]

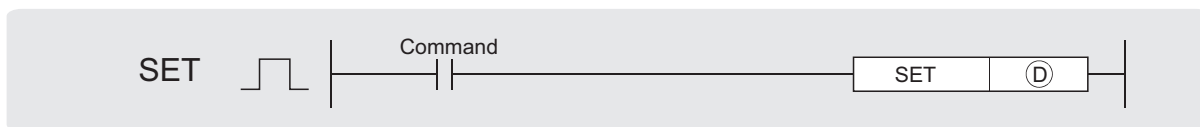
Step	Instruction	Device
0	LD	X0
1	OUT	F7
3	END	

[Operation]



### 5.3.5 Setting devices (except for annunciators) (SET)

Basic High performance Process Redundant Universal



ⓓ : Bit device number to be set (ON)/Word device bit designation (bits)

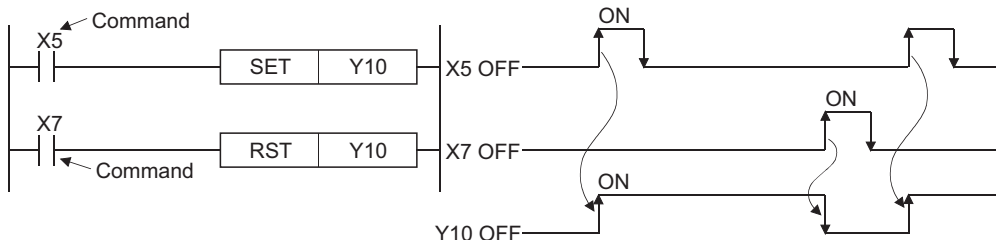
Setting Data	Internal Devices		R, ZR	J, D		U, G, O	Zn	Constants	Other BL, DY
	Bit	Word		Bit	Word				
ⓓ	○	○ (Other than T, C)			○		—		○

#### ★ Function

- (1) When the execution command is turned ON, the status of the designated devices becomes as shown below:

Device	Device Status
Bit device	Coils and contacts turned ON
When Bit Designation has been Made for Word Device	Designation bit set at 1

- (2) Devices turned ON by the instruction remain ON when the same command is turned OFF. Devices turned ON by the SET instruction can be turned OFF by the RST instruction.



- (3) When the execution command is OFF, the status of devices does not change.

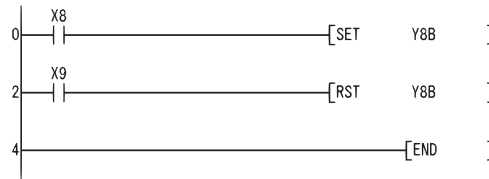
## ! Operation Error

- (1) There are no operation errors associated with the SET instruction.

## Program Example

- (1) The following program sets Y8B (ON) when X8 goes ON, and resets Y8B (OFF) when X9 goes ON.

[Ladder Mode]

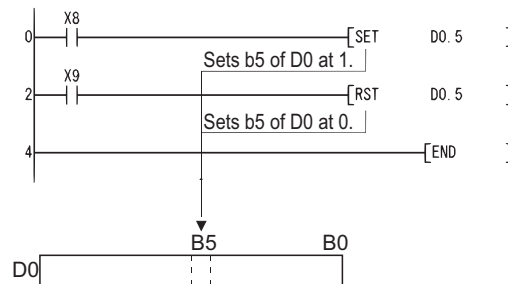


[List Mode]

Step	Instruction	Device
0	LD	X8
1	SET	Y8B
2	LD	X9
3	RST	Y8B
4	END	

- (2) The following program sets the value of D0 bit 5 (b5) to 1 when X8 goes ON, and set the bit value to 0 when X9 goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X8
1	SET	D0.5
2	LD	X9
3	RST	D0.5
4	END	

### Remark

1. The number of basic steps for the SET instruction is as follows:

- When internal device or file register (R0 to R32767) are in use : 1
- When direct access output (DY) or SFC program device (BL) are in use : 2
- When using serial number access format file register (Only for Universal model QCPU) : 2  
(Other than Universal model QCPU) : 3
- When some other device is in use : 3

2. When using X as a device, use the device numbers that are not used for the actual input. If the same number is used for the actual input device and input X, the data of the actual input will be written over the input X specified in the SET instruction.

### 5.3.6 Resetting devices (except for annunciators) (RST)

Basic High performance Process Redundant Universal



Ⓧ : Bit device number to be reset/ Word device bit designation (bits)  
Word device number to be reset (BIN 16 bits)

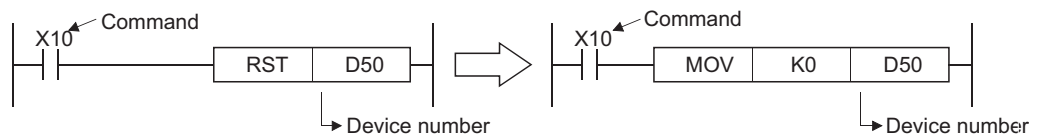
Setting Data	Internal Devices		R, ZR	JYD		UDYG	Zn	Constants	Other DY
	Bit	Word		Bit	Word				
Ⓧ					○			—	○

#### ★ Function

- (1) When the execution command is turned ON, the status of the designated devices becomes as shown below:

Device	Device Status
Bit device	Turns coils and contacts OFF
Timers and counters	Sets the present value to 0, and turns coils and contacts OFF
When Bit Designation has been Made for Word Device	Sets value of designated bit to 0
Word devices other than timers and counters	Sets contact to 0

- (2) When the execution command is OFF, the status of devices does not change.
- (3) The functions of the word devices designated by the RST instruction are identical to the following ladder:



## ! Operation Error

(1) There are no operation errors associated with the RST instruction.

**Remark**

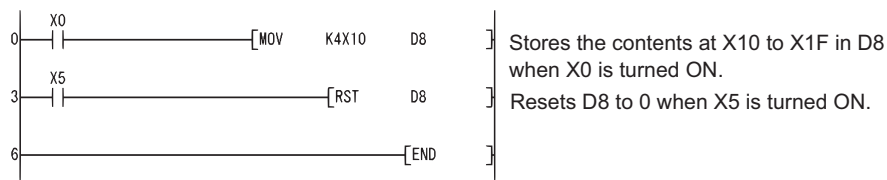
The basic number of steps of the RST instruction is as follows.

- a) For bit processing
  - Internal device (bit to be specified by bit device or word device) : 1
  - Direct access output : 2
  - Timer, counter : 4
  - When using serial number access format file register
    - (Only for Universal model QCPU) : 2
    - (Other than Universal model QCPU) : 3
  - Other than above : 3
- b) For word processing
  - Internal device : 2
  - Index register : 2
  - When using serial number access format file register
    - (Only for Universal model QCPU) : 2
    - (Other than Universal model QCPU) : 3
  - Other than above : 3

## Program Example

(1) The following program sets the value of the data register to 0.

[Ladder Mode]

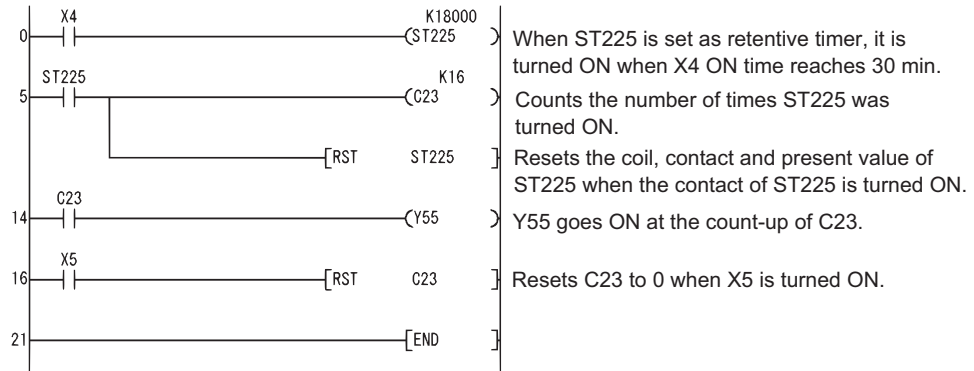


[List Mode]

Steps	Instruction	Device
0	LD	X0
1	MOV	K4X10 D8
3	LD	X5
4	RST	D8
6	END	

(2) The following program resets the 100 ms retentive timer and counter.

[Ladder Mode]

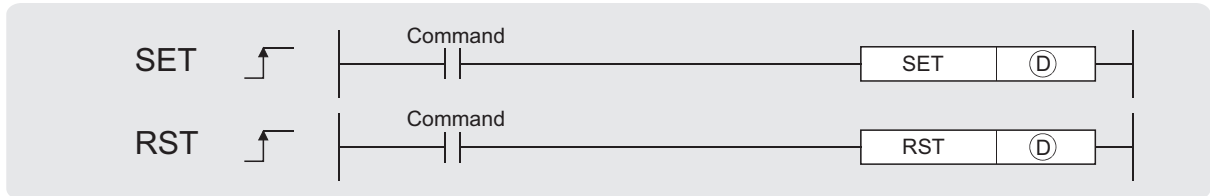


[List Mode]

Step	Instruction	Device
0	LD	X4
1	OUT	ST225 K18000
5	LD	ST225
6	OUT	C23 K16
10	RST	ST225
14	LD	C23
15	OUT	Y55
16	LD	X5
17	RST	C23
21	END	

# 5.3.7 Setting and resetting the annunciators (SET F,RST F)

Basic High performance Process Redundant Universal



SET  $\text{D}$  : Number of the annunciator to be set (F number) (bits)  
 RST  $\text{D}$  : Number of the annunciator to be reset (F number) (bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
$\text{D}$	<input type="radio"/> (Only F)								

5

## ★ Function

### SET

- (1) The annunciator designated by  $\text{D}$  is turned ON when the execution command is turned ON.
- (2) The following responses occur when an annunciator (F) is turned ON.
  - The "USER" LED goes ON.\*1
  - The annunciator numbers which are ON (F numbers) are stored in special registers (SD64 to SD79).
  - The value of SD63 is incremented by 1.
- \*1: When using the Basic model QCPU, the "ERR."LED goes ON.
- (3) If the value of SD63 is 16 (which happens when 16 annunciators are already ON), even if a new annunciator is turned ON, its number will not be stored at SD64 to SD79.

### RST

- (1) The annunciator designated by  $\text{D}$  is turned OFF when the execution command is turned ON.
- (2) The annunciator numbers (F numbers) of annunciators that have gone OFF are deleted from the special registers (SD64 to SD79), and the value of SD63 is decremented by 1.

**Remark** .....

1. For details of annunciators, refer to the User's Manual (Functions Explanation Program Fundamentals) for the CPU module used.
  2. The number of basic steps for the SET F and RST F instructions is 2.
- .....

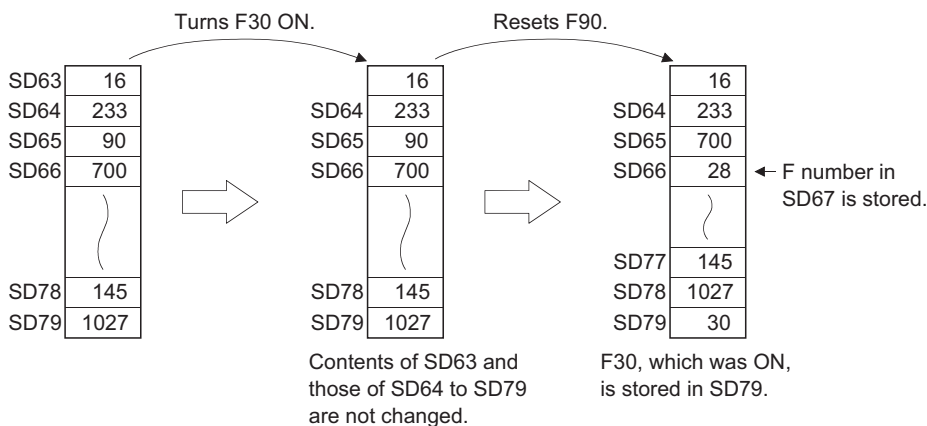
5.3 Output Instructions  
5.3.7 Setting and resetting the annunciators (SET F,RST F)

- (3) When the value of SD63 is "16", the annunciator numbers are deleted from SD64 to SD79 by the use of the RST instruction. If the annunciators whose numbers are not registered in SD64 to SD79 are ON, these numbers will be registered.

If all annunciator numbers from SD64 to SD79 are turned OFF, the LED display device on the front of the CPU module, or the "USER" LED, will be turned OFF.\*2

\*2: When using the Basic model QCPU, the "ERR." LED goes OFF.

**[Operations which take place when SD63 is 16]**



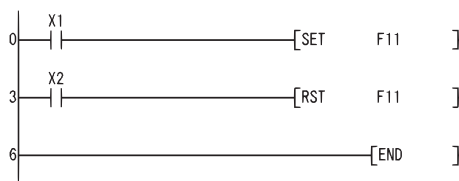
**! Operation Error**

- (1) There are no operation errors associated with the SET F□ or RST F□ instruction.

**Program Example**

- (1) The following program turns annunciator F11 ON when X1 goes ON, and stores the value 11 at the special register (SD64 to SD79). Further, the program resets annunciator F11 if X2 goes ON, and deletes the value 11 from the special registers (SD64 to SD79).

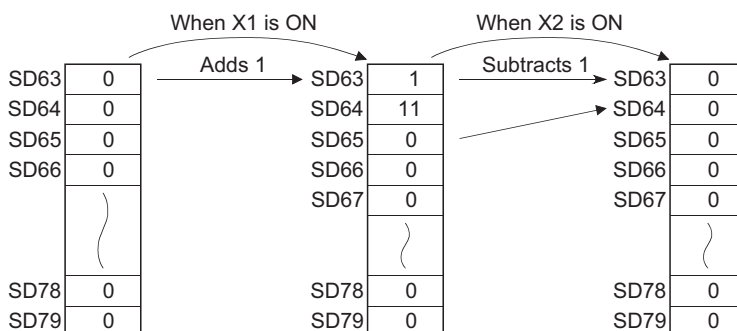
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1
1	SET	F11
3	LD	X2
4	RST	F11
6	END	

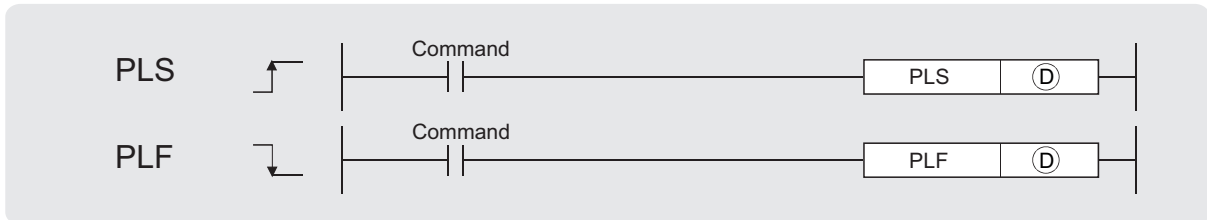
[Operation]





### 5.3.8 Leading edge and trailing edge outputs (PLS,PLF)

Basic High performance Process Redundant Universal



Ⓧ : Pulse conversion device (bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other DY
	Bit	Word		Bit	Word				
Ⓧ				○			—		○

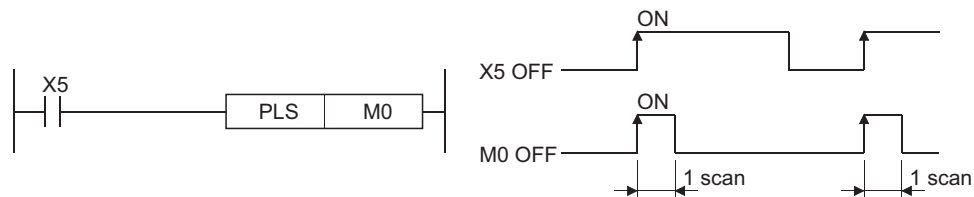
## ★ Function

### PLS

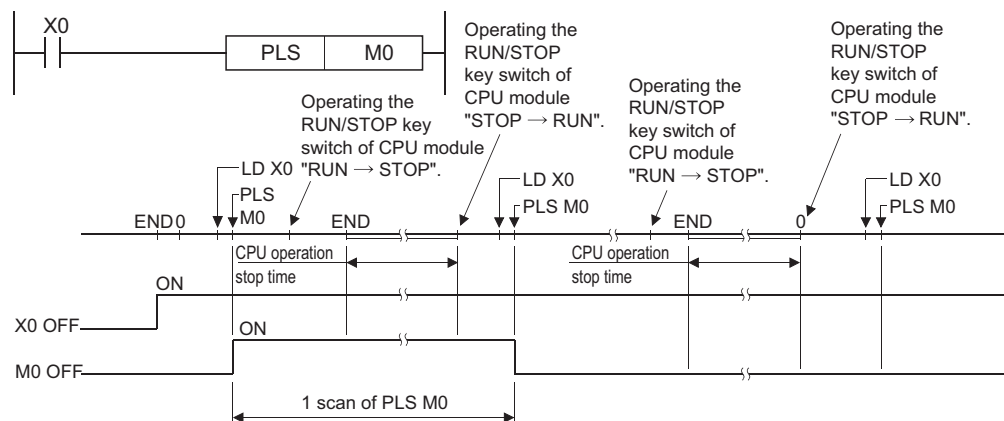
- (1) Turns ON the designated device when the execution command is turned OFF→ON, and turns OFF the device in any other case the execution command is turned OFF → ON (i.e., at ON → ON, ON → OFF or OFF → OFF of the execution command).

When there is one PLS instruction for the device designated by Ⓧ during one scan, the specified device turns ON one scan.

See Section 3.9 for the operation to be performed when the PLS instruction for the same device is executed more than once during one scan.



- (2) If the RUN/STOP key switch is changed from RUN to STOP after the execution of the PLS instruction, the PLS instruction will not be executed again even if the switch is set back to RUN.



5.3 Output Instructions  
5.3.8 Leading edge and trailing edge outputs (PLS,PLF)

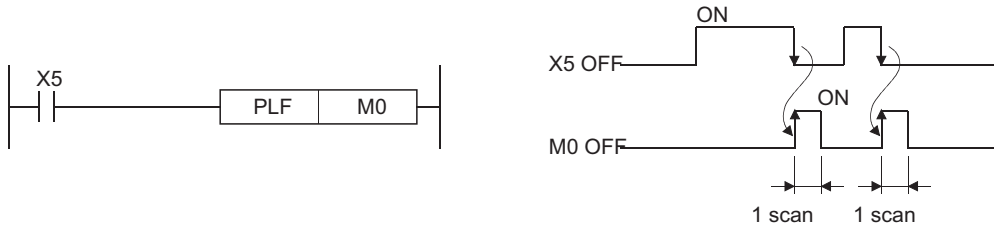
- (3) When designating a latch relay (L) for the execution command and turning the power supply OFF to ON with the latch relay ON, the execution command turns OFF to ON at the first scan, executing the PLS instruction and turning ON the designated device.  
The device turned ON at the first scan after power-ON turns OFF at the next PLS instruction.

**PLF**

- (1) Turns ON the designated device when the execution command is turned ON→OFF, and turns OFF the device in any other case the execution command is turned ON → OFF (i.e., at OFF → OFF, OFF → ON or ON → ON of the execution command).

When there is one PLF instruction for the device designated by Ⓣ during one scan, the specified device turns ON one scan.

See Section 3.9 for the operation to be performed when the PLF instruction for the same device is executed more than once during one scan.



- (2) If the RUN/STOP key switch is changed from RUN to STOP after the execution of the PLF instruction, the PLF instruction will not be executed again even if the switch is set back to RUN.

**POINT**

Note that the device designated by Ⓣ may remain ON for more than one scan if the PLS or PLF instruction is jumped by the CJ instruction or if the executed subroutine program was not called by the CALL instruction.

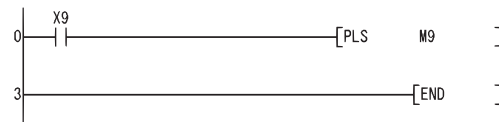
**! Operation Error**

- (1) There are no operation errors associated with the PLS or PLF instruction.

**Program Example**

- (1) The following program executes the PLS instruction when X9 goes ON.

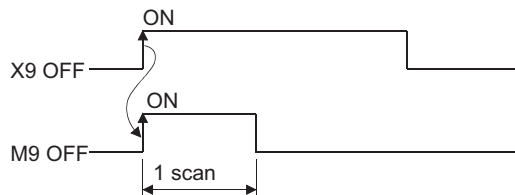
[Ladder Mode]



[List Mode]

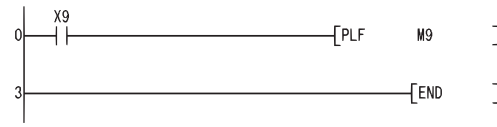
Step	Instruction	Device
0	LD	X9
1	PLS	M9
3	END	

[Timing Chart]



(2) The following program executes the PLF instruction when X9 goes OFF.

[Ladder Mode]



[List Mode]

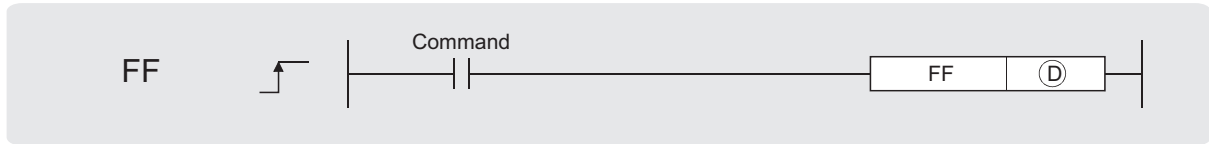
Step	Instruction	Device
0	LD	X9
1	PLF	M9
3	END	

[Timing Chart]



## 5.3.9 Bit device output reverse (FF)

Basic High performance Process Redundant Universal



Ⓧ : Device number of the device to be reversed (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other DY
	Bit	Word		Bit	Word				
Ⓧ				○			—		○

### ★ Function

- (1) Reverses the output status of the device designated by Ⓧ when the execution command is turned OFF→ON.

Device	Device Status	
	Prior to FF Execution	After FF Execution
Bit device	OFF	ON
	ON	OFF
Bit designated for word device	0	1
	1	0

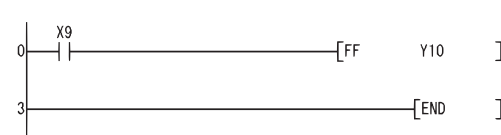
### ! Operation Error

- (1) There are no operation errors associated with the FF instruction.

### 📄 Program Example

- (1) The following program reverses the output of Y10 when X9 goes ON.

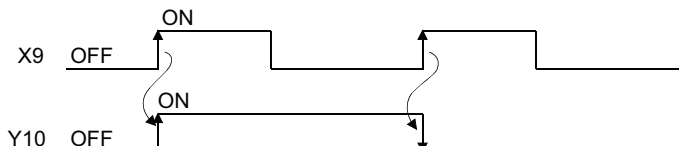
[Ladder Mode]



[List Mode]

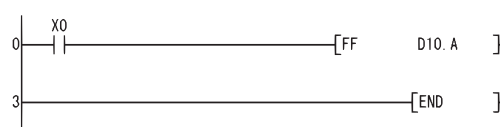
Step	Instruction	Device
0	LD	X9
1	FF	Y10
3	END	

[Timing Chart]



(2) The following program reverses b10 (bit 10) of D10 when X0 goes ON.

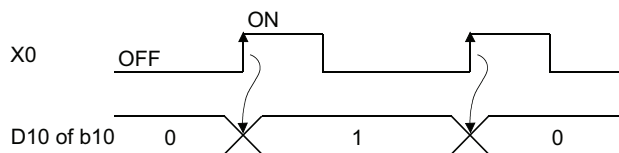
[Ladder Mode]



[List Mode]

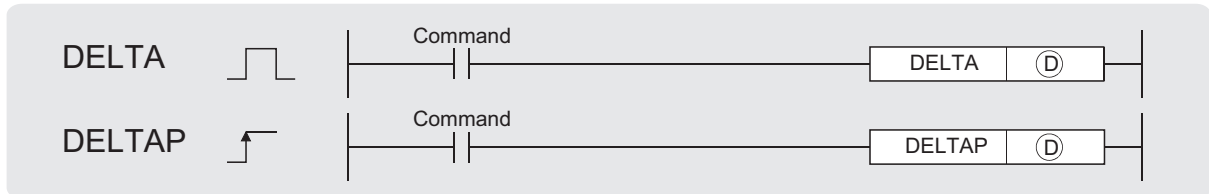
Step	Instruction	Device
0	LD	X0
1	FF	D10. A
3	END	

[Timing Chart]



## 5.3.10 Pulse conversions of direct outputs (DELTA(P))

Basic High performance Process Redundant Universal



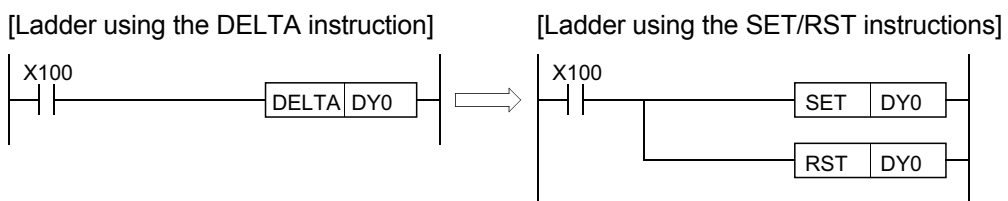
Ⓧ : Bit for which pulse conversion is to be conducted (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other DY
	Bit	Word		Bit	Word				
Ⓧ									○

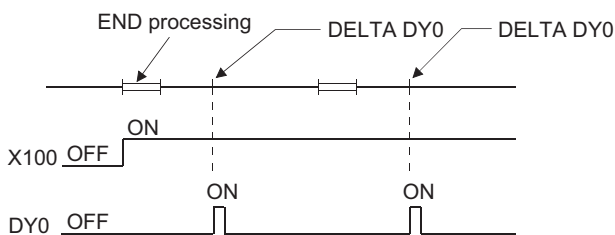
### ★ Function

- (1) Conducts pulse output of direct access output (DY) designated by Ⓧ.

If DELTA DY0 has been designated, the resulting operation will be identical to the ladder shown below, which uses the SET/RST instructions.



[Operation]



- (2) The DELTA (P) instruction is used by commands for leading edge execution for an intelligent function module.

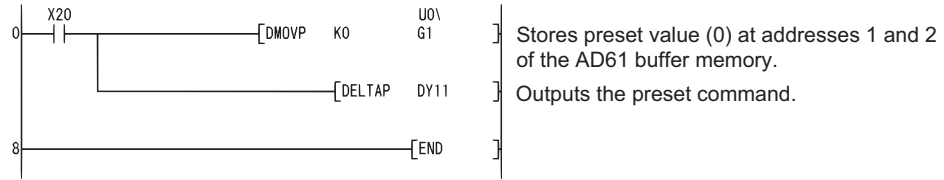
### ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- A direct access output number designated by Ⓧ has exceeded the CPU module output range. (Error code: 4101)

## Program Example

- (1) The following program presets CH1 of the AD61 mounted at slot 0 of the main base unit, when X20 goes ON.

[Ladder Mode]



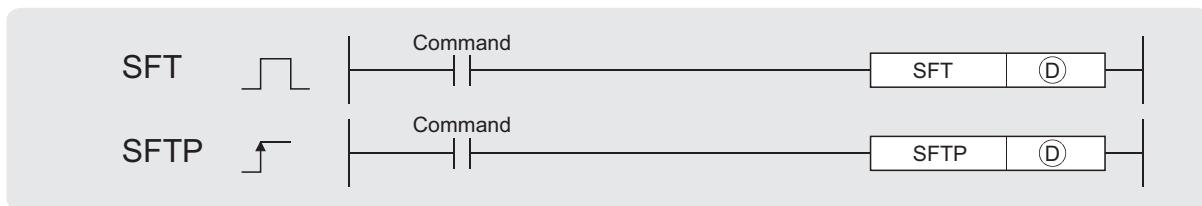
[List Mode]

Step	Instruction	Device
0	LD	X20
1	DMOVP	K0 U0#G1
6	DELTAP	DY11
8	END	

## 5.4 Shift Instructions

### 5.4.1 Bit device shifts (SFT(P))

Basic High performance Process Redundant Universal

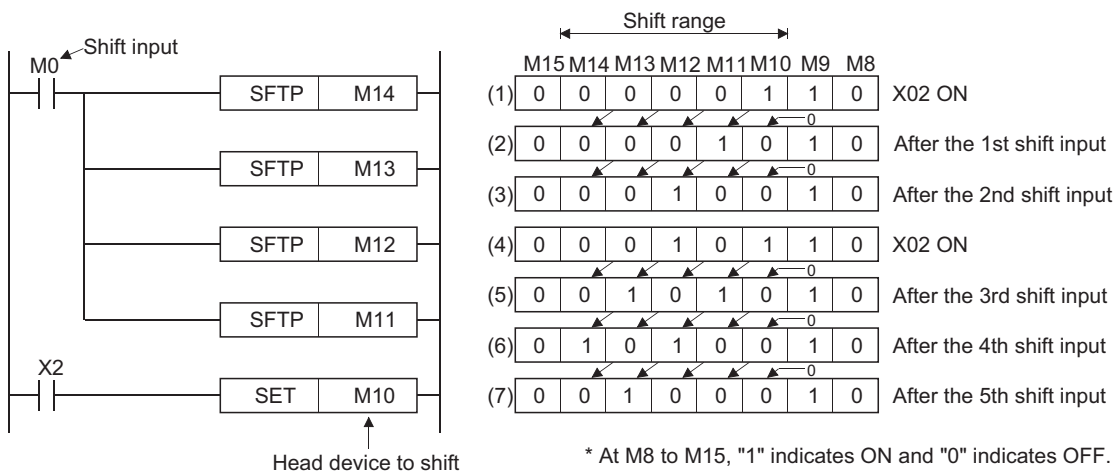


Ⓧ : Device number to shift (bits)

Setting Data	Internal Devices		R, ZR	JOG		U/GO	Zn	Constants	Other DY
	Bit	Word		Bit	Word				
Ⓧ	○ (Other than T, C)						—	○	

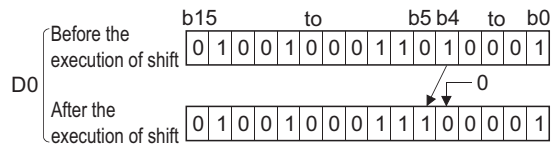
### ★ Function

- (1) When bit device is used
  - (a) Shifts to a device designated by Ⓧ the ON/OFF status of the device immediately prior to the one designated by Ⓧ, and turns the prior device OFF.  
For example, if M11 has been designated by the SFT instruction, when the SFT instruction is executed, it will shift the ON/OFF status of M10 to M11, and turn M10 OFF.
  - (b) Turn the first device to be shifted ON with the SET instruction.
  - (c) When the SFT and SFTP are to be used consecutively, the program starts from the device with the larger number.





- (2) When word device bit designation is used
- (a) Shifts to a bit in the device designated by  $\textcircled{D}$  the 1/0 status of the bit immediately prior to the one designated by  $\textcircled{D}$ , and turns the prior bit to 0.
- For example, if D0.5 (bit 5 [b5] of D0) has been designated by the SFT instruction, when the SFT instruction is executed, it will shift the 1/0 status of b4 of D0 to b5, and turn b4 to 0.



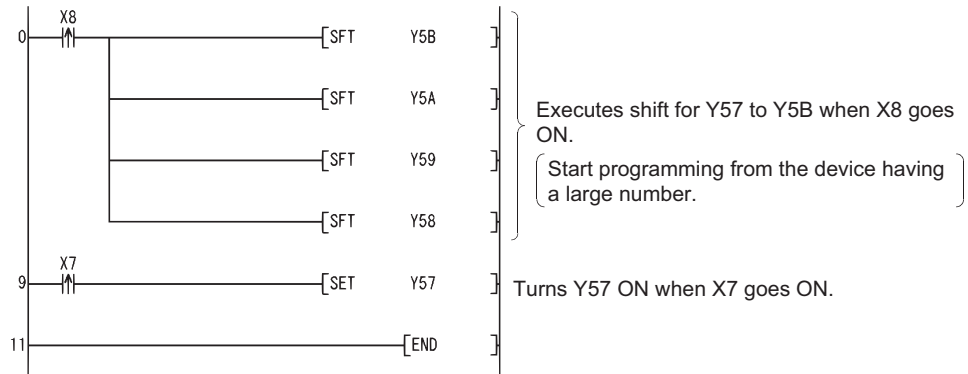
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified by  $\textcircled{D}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

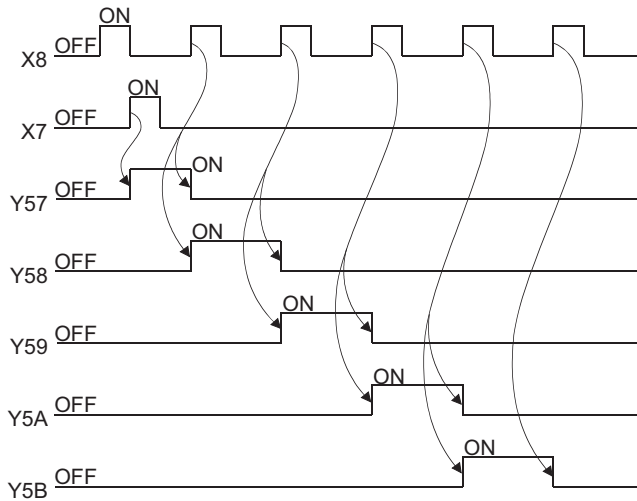
# Program Example

(1) The following program shifts Y57 to Y5B when X8 goes ON.

[Ladder Mode]



[Timing Chart]



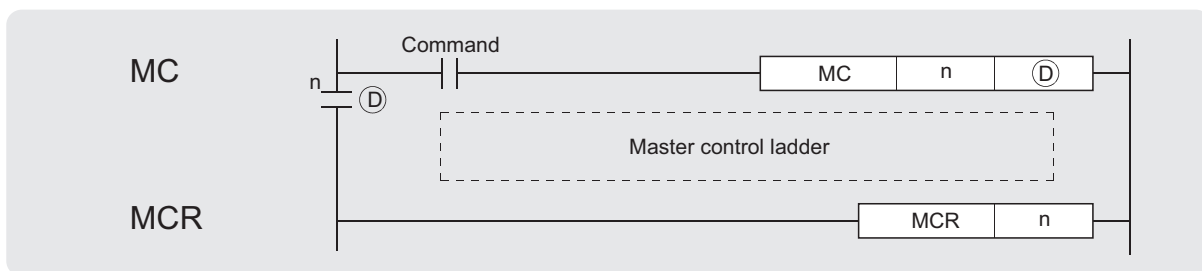
[List Mode]

Step	Instruction	Device
0	LDP	X8
1	SFT	Y5B
3	SFT	Y5A
5	SFT	Y59
7	SFT	Y58
9	LDP	X7
10	SET	Y57
11	END	

# 5.5 Master Control Instructions

## 5.5.1 Setting and resetting the master control (MC,MCR)

Basic High performance Process Redundant Universal



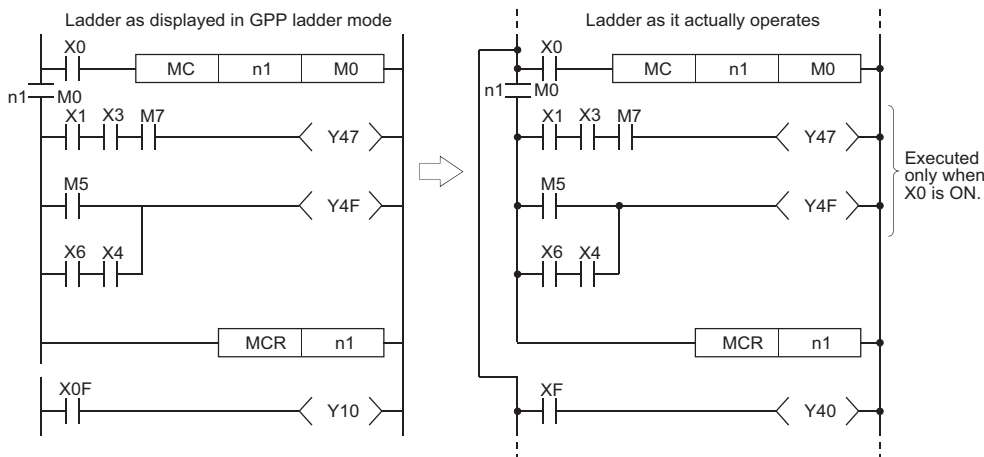
n : Nesting (N0 to N14) (Nesting)  
 (D) : Device number to be turned ON (bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other	
	Bit	Word		Bit	Word				N	DY
n							—		○	—
(D)				○			—		—	○

### ★ Function

The master control instruction is used to enable the creation of highly efficient ladder switching sequence programs, through the opening and closing of a common bus for ladders.

A ladder using the master control is as follows:



#### Remark

Inputting of contacts on the vertical bus is not necessary when programming in the write mode of a peripheral device. These will be automatically displayed when the "conversion" operation is conducted after the creation of the ladder and then "read" mode is set.

5.5 Master Control Instructions  
5.5.1 Setting and resetting the master control (MC,MCR)

5

**MC**

- (1) If the execution command of the MC instruction is ON when master control is started, the result of the operation from the MC instruction to the MCR instruction will be exactly as the instruction (ladder) shows.

If the execution command of the MC instruction is OFF, the result of the operation from the MC instruction to the MCR instruction will be as shown below:

Device	Device Status
High speed timer Low speed timer	Count value goes to 0, coils and contacts all go OFF.
High speed retentive timer Low speed retentive timer Counter	Coils go OFF, but counter values and contacts all maintain current status.
Devices in OUT instruction	All turned OFF
SET, RST SFT Basic, Application	Devices in the following instructions: Maintain current status

- (2) Even when the MC instruction is OFF, instructions from the MC instruction to the MCR instruction will be executed, so scan time will not be shortened.

**POINT**

When a ladder with master control contains instructions that do not require any contact instruction (such as FOR to NEXT, EI, DI instructions), the CPU module executes these instructions regardless of the ON/OFF status of the MC instruction execution command.

- (3) By changing the device designated by  $\text{Ⓢ}$ , the MC instruction can use the same nesting (N) number as often as desired.
- (4) Coils from devices designated by  $\text{Ⓢ}$  are turned ON when the MC instruction is ON. Further, using these same devices with the OUT instruction or other instructions will cause them to become double coils, so devices designated by  $\text{Ⓢ}$  should not be used within other instructions.

**MCR**

- (1) This is the instruction for recovery from the master control, and indicates the end of the master control range of operation.
- (2) Do not place contact instructions before the MCR instruction.
- (3) Use the MC instruction and MCR instruction of the same nesting number as a set. However, when the MCR instructions are nested in one place, all master controls can be terminated with the lowest nesting (N) number.  
(Refer to the "Precautions for nesting" in the program example.)

**Operation Error**

- (1) There are no operation errors associated with the MC or MCR instruction.

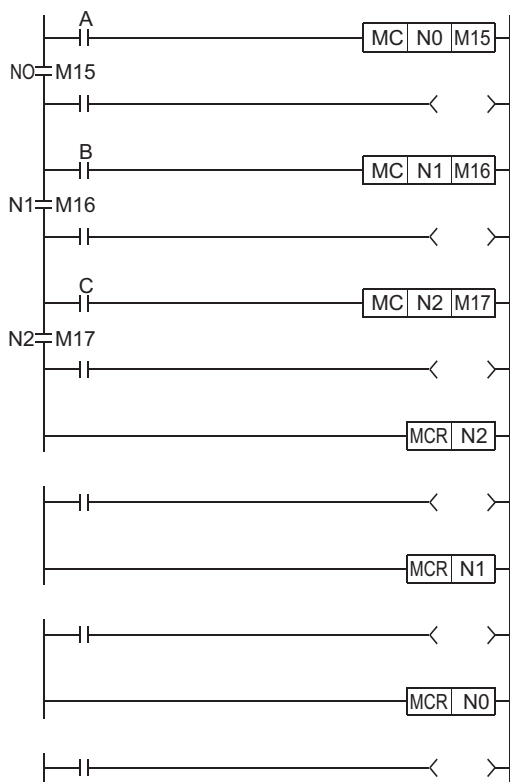
## Program Example

The master control instruction can be used in nesting. The different master control regions are distinguished by nesting (N). Nesting can be performed from N0 to N14.

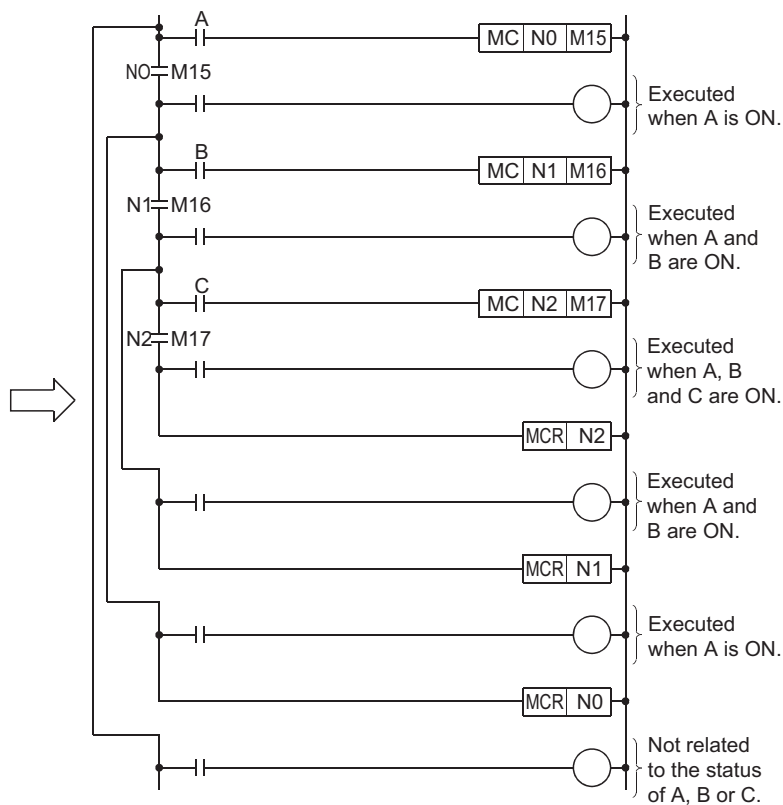
The use of nesting enables the creation of ladders which successively limit the execution condition of the program.

A ladder using nesting would appear as shown below:

[Ladder as displayed in the GPP ladder mode]



[Ladder as it actually operates]

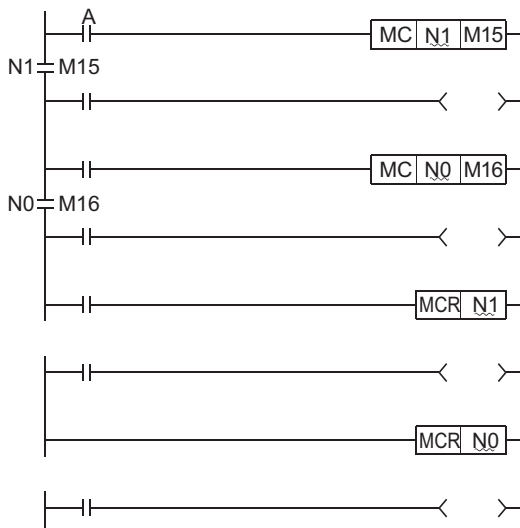


### Cautions when Using Nesting Architecture

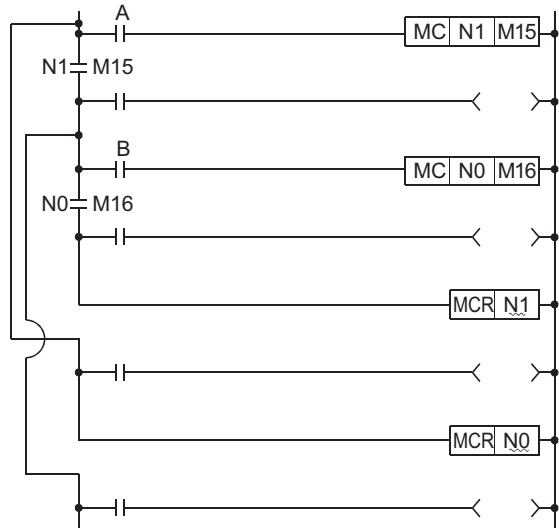
(1) Nesting can be used up to 15 times (N0 to N14)

When using nesting, nests should be inserted from the lower to higher nesting number (N) with the MC instruction, and from the higher to the lower order with the MCR instruction. If this order is reversed, there will be no nesting architecture, and the CPU module will not be capable of performing correct operations. For example, if nesting is designated in the order N1 to N0 by the MC instruction, and also designated in the N1 to N0 order by the MCR instruction, the vertical bus will intersect and a correct master control ladder will not be produced.

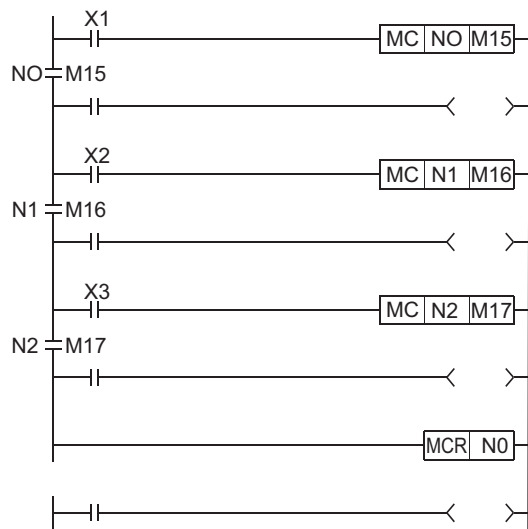
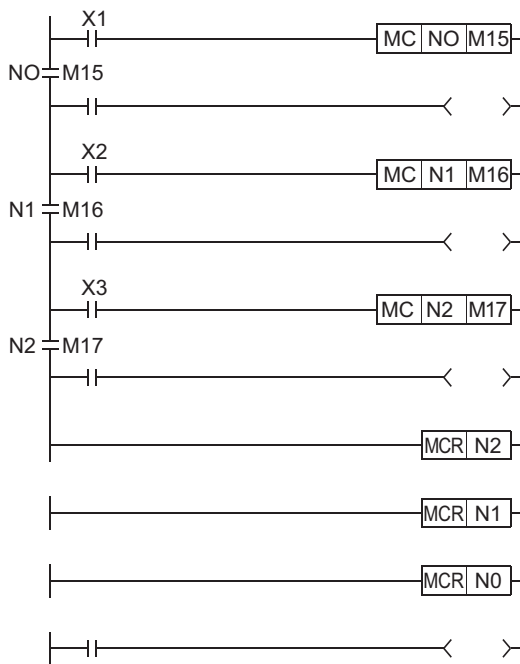
[Ladder as displayed in the GPP ladder mode]



[Ladder as it actually operates]



(2) If the nesting architecture results in MCR instructions concentrated in one location, all master controls can be terminated by use of just the lowest nesting number (N).



# 5.6 Termination Instructions

## 5.6.1 End main routine program (FEND)

Basic High performance Process Redundant Universal

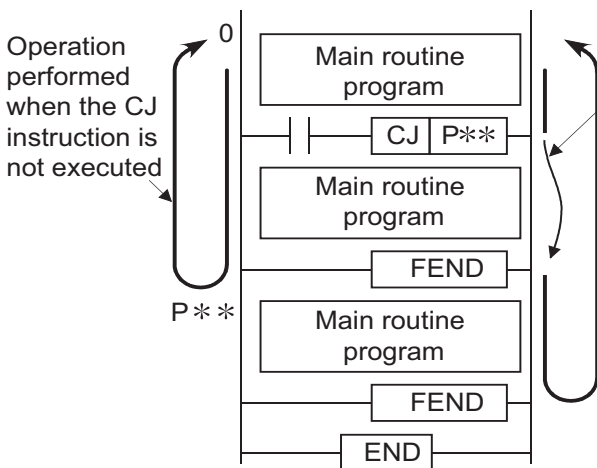


Setting Data	Internal Devices		R, ZR	J:G:Q		U:G:Q	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

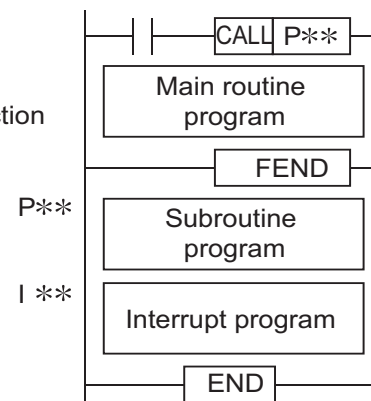
5

### ★ Function

- (1) The FEND instruction is used in cases where the CJ instruction or other instructions are used to cause a branch in the sequence program operations, and in cases where the main routine program is to be split from a subroutine program or an interrupt program.
- (2) Execution of the FEND instruction will cause the CPU module to terminate the program it was executing.
- (3) Even sequence programs following the FEND instruction can be displayed in ladder display at a peripheral device.  
(Peripheral devices continue to display ladders until encountering the END instruction.)



(a) When the CJ instruction is used



(b) When there are subroutine and interrupt programs

5.6 Termination Instructions  
5.6.1 End main routine program (FEND)



## Operation Error

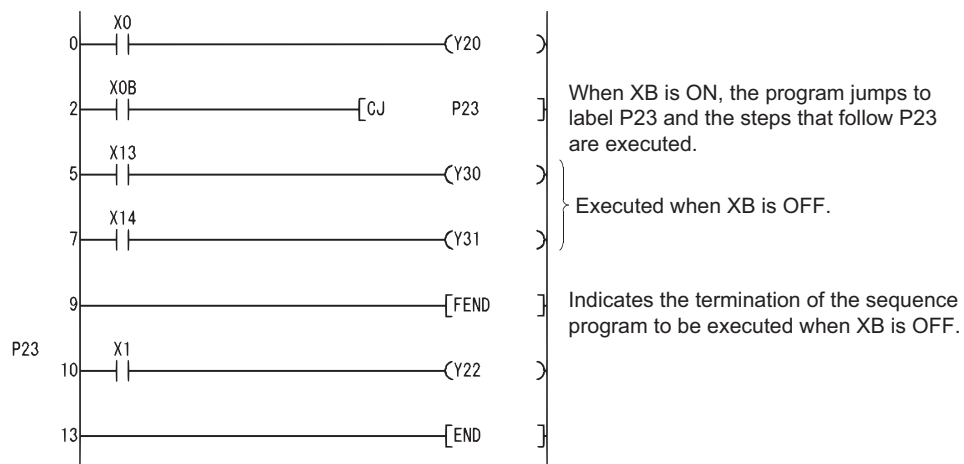
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The FEND instruction is executed after the execution of the CALL, FCALL, ECALL, or EFCALL instruction, and before the execution of the RET instruction. (Error code: 4211)
  - The FEND instruction is executed after the execution of the FOR instruction, and before the execution of the NEXT instruction. (Error code: 4200)
  - The FEND instruction is executed during an interrupt program, and before the execution of the IRET instruction. (Error code: 4221)
  - The FEND instruction is executed between the CHKCIR and CHKEND instructions. (Error code: 4230)
  - The FEND instruction is executed between the IX and IXEND instructions. (Error code: 4231)



## Program Example

- (1) The following program uses the CJ instruction.

[Ladder Mode]



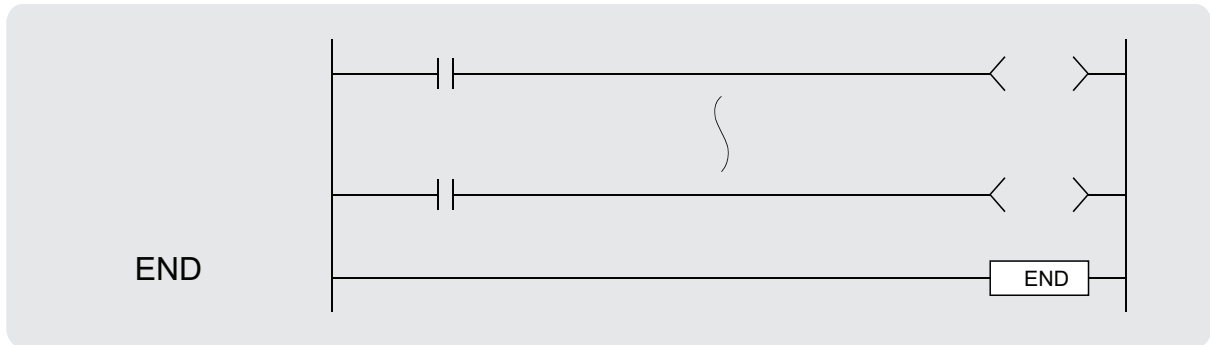
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	Y20
2	LD	X0B
3	CJ	P23
5	LD	X13
6	OUT	Y30
7	LD	X14
8	OUT	Y31
9	FEND	
10	P23	
11	LD	X1
12	OUT	Y22
13	END	



## 5.6.2 End sequence program (END)

Basic High performance Process Redundant Universal

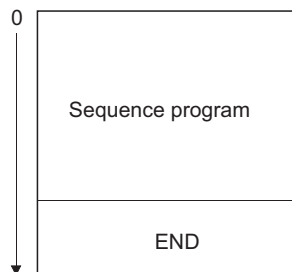


Setting Data	Internal Devices		R, ZR	J:V:Q		U:G:Q	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

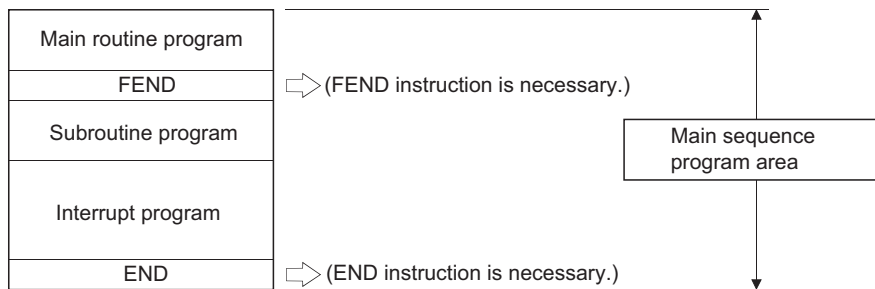
- (1) Indicates termination of programs, including main routine program, subroutine program, and interrupt programs.

Execution of the END instruction will cause the CPU module to terminate the program that was being executed.



- (2) The END instruction cannot be used during the execution of the main sequence program. If it is necessary to perform END processing during the execution of a program, use the FEND instruction.
- (3) When programming in the ladder mode of a peripheral device, it is not necessary to input the END instruction.

- (4) The use of the END and FEND instructions is broken down as follows for main routine programs, subroutine programs, and interrupt programs:



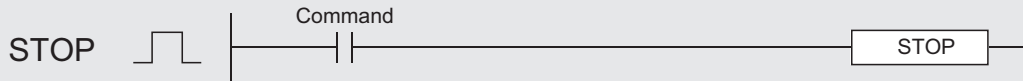
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The END instruction was executed before the execution of the RET instruction and after the execution of the CALL, FCALL, ECALL, or EFCALL instruction.  
(Error code: 4211)
  - The END instruction was executed before the execution of the NEXT instruction and after the execution of the FOR instruction.  
(Error code: 4200)
  - The END instruction was executed during an interrupt program prior to the execution of the IRET instruction.  
(Error code: 4221)
  - The END instruction was executed within the CHKCIR to CHKEND instruction loop.  
(Error code: 4230)
  - The END instruction was executed within the IX to IXEND instruction loop.  
(Error code: 4231)

## 5.7 Other instructions

### 5.7.1 Sequence program stop (STOP)

Basic High performance Process Redundant Universal

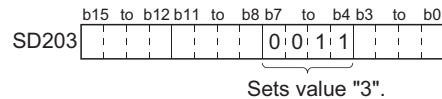


Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

#### ★ Function

- (1) Resets the output (Y) and stops the CPU module operation when the execution command is turned ON.  
(The same result will take place if the RUN/STOP (key) switch is turned to the STOP setting.)

- (2) Execution of the STOP instruction will cause the value of b4 to b7 of the special register SD203 to become "3".



- (3) In order to restart CPU module operations after the execution of the STOP instruction, return the RUN/STOP key switch, which has been changed from RUN to STOP, back to the RUN position.



## Operation Error

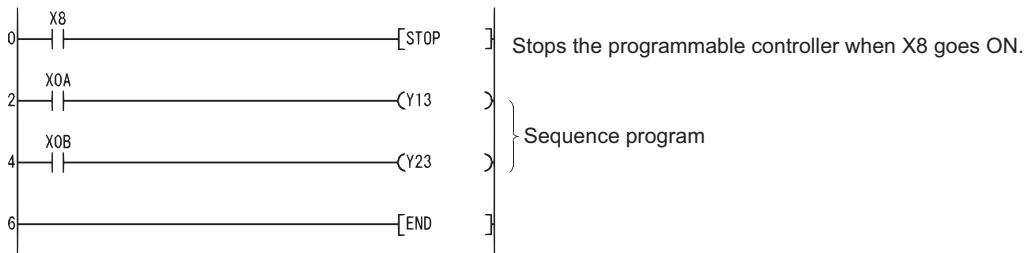
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The STOP instruction was executed before the execution of the RET instruction and after the execution of the CALL/FCALL/ECALL/EFCALL/XCALL instruction. (Error code: 4211)
  - The STOP instruction was executed before the execution of the NEXT instruction and after the execution of the FOR instruction. (Error code: 4200)
  - The STOP instruction was executed during an interrupt program prior to the execution of the IRET instruction. (Error code: 4221)
  - The STOP instruction was executed within the CHKCIR to CHKEND instruction loop. (Error code: 4230)
  - The STOP instruction was executed within the IX to IXEND instruction loop. (Error code: 4231)
  - The STOP instruction was executed during the fixed scan execution type program. (For the Universal model QCPU only) (Error code: 4223)



## Program Example

- (1) The following program stops the CPU module when X8 goes ON.

[Ladder Mode]

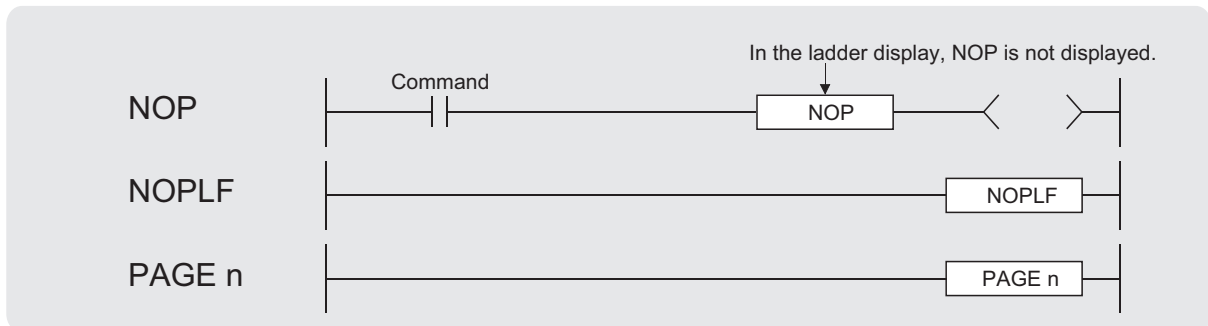


[List Mode]

Step	Instruction	Device
0	LD	X8
1	STOP	
2	LD	X0A
3	OUT	Y13
4	LD	X0B
5	OUT	Y23
6	END	

## 5.7.2 No operations (NOP,NOPLF,PAGE n)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

#### NOP

- (1) This is a no operation instruction that has no impact on any operations up to that point.
- (2) The NOP instruction is used in the following cases:
  - (a) To insert space for sequence program debugging.
  - (b) To delete an instruction without having to change the number of steps. (Replace the instruction with NOP.)
  - (c) To temporarily delete an instruction.

#### NOPLF

- (1) This is a no operation instruction that has no impact on any operations up to that point.
- (2) The NOPLF instruction is used when printing from a peripheral device to force a page change at any desired location.
  - (a) When printing ladders
    - A page break will be inserted between ladder blocks with the presence of the NOPLF instruction.
    - The ladder cannot be displayed correctly if an NOPLF instruction is inserted in the midst of a ladder block.  
Do not insert an NOPLF instruction in the midst of a ladder block.
  - (b) When printing instruction lists
    - The page will be changed after the printing of the NOPLF instruction.
- (3) Refer to the Operating Manual for the peripheral device in use for details of printouts from peripheral devices.

## PAGE n

- (1) This is a no operation instruction that has no impact on any operations up to that point.
- (2) No processing is performed at peripheral devices with this instruction.



## Operation Error

- (1) There are no errors associated with the NOP, NOPLF, or PAGE instruction.

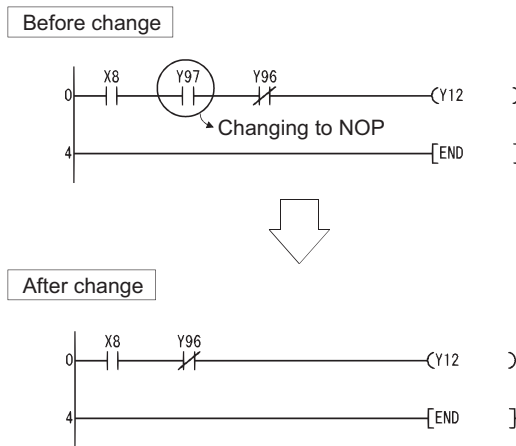


## Program Example

### NOP

- (1) Contact closed.... Deletes the AND or ANI instruction.

[Ladder Mode]



[List Mode]

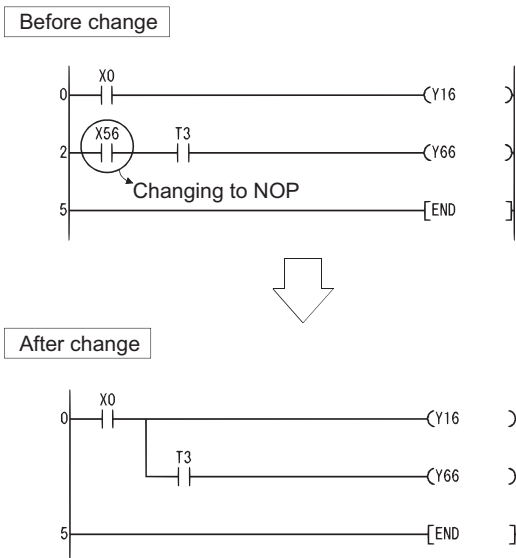
Step	Instruction	Device
0	LD	X8
1	AND	Y97
2	ANI	Y96
3	OUT	Y12
4	END	

Step	Instruction	Device
0	LD	X8
1	NOP	
2	ANI	Y96
3	OUT	Y12
4	END	

- (2) Contact closed.... LD, LDI changed to NOP. (Note carefully that changing the LD and LDI instructions to NOP completely changes the nature of the ladder.)

[Ladder Mode]



[List Mode]

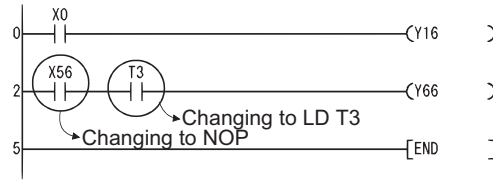
Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	LD	X56
3	AND	T3
4	OUT	Y66
5	END	

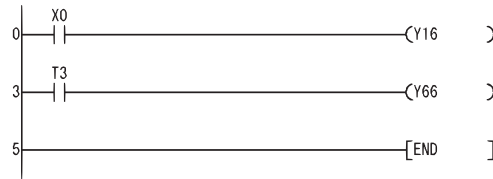
Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	NOP	
3	AND	T3
4	OUT	Y66
5	END	

[Ladder Mode]

Before change



After change



[List Mode]

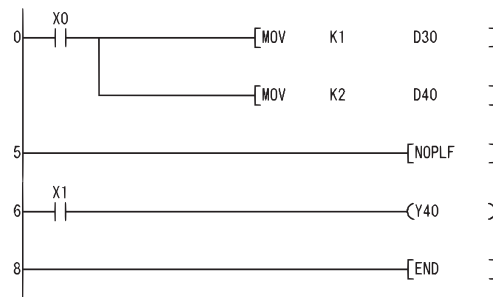
Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	LD	X56
3	AND	T3
4	OUT	Y66
5	END	

Step	Instruction	Device
0	LD	X0
1	OUT	Y16
2	NOP	
3	LD	T3
4	OUT	Y66
5	END	

5

**NOPLF**

[Ladder Mode]

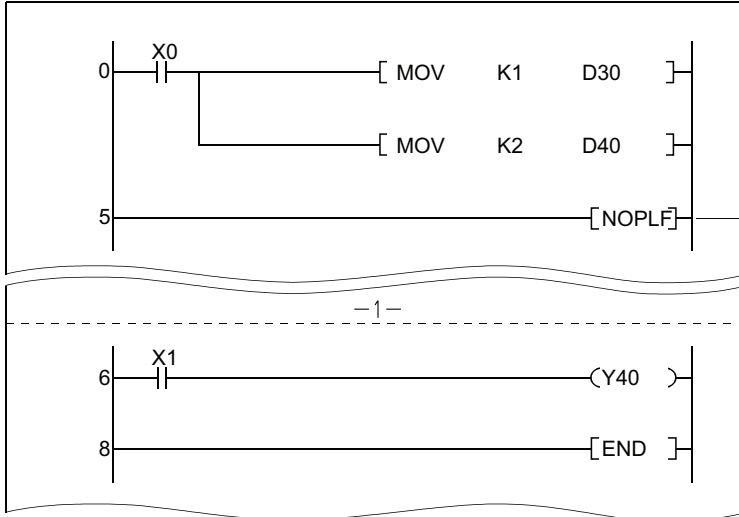


[List Mode]

Step	Instruction	Device
0	LD	X0
1	MOV	K1 D30
3	MOV	K2 D40
5	NOPLF	
6	LD	X1
7	OUT	Y40
8	END	

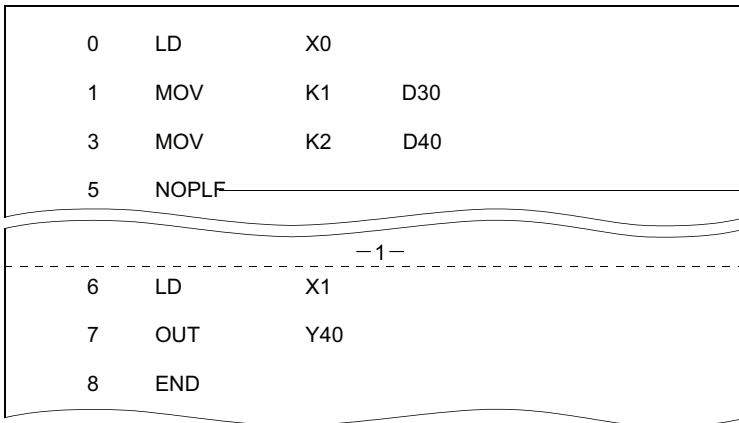
5.7 Other instructions  
5.7.2 No operations (NOP,NOPLF,PAGE n)

- Printing the ladder will result in the following:



→ NOPLF instruction, inserted as a delimiter of ladder blocks, causes print out page to be changed forcibly.

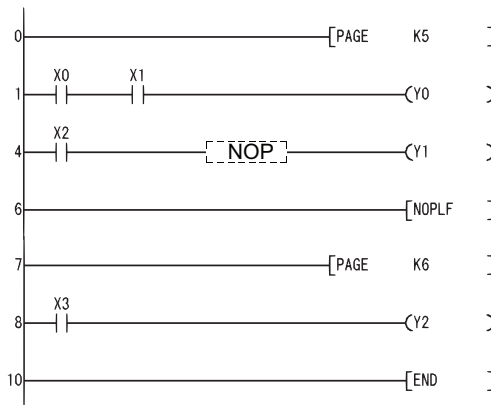
- Printing an instruction list with the NOPLF instruction will result in the following:



→ Changes print output page after printing NOPLF.

### PAGE n

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	PAGE	K5
1	LD	X0
2	AND	X1
3	OUT	Y0
4	LD	X2
5	NOP	
6	OUT	Y1
7	NOPLF	
8	PAGE	K6
9	LD	X3
10	OUT	Y2
11	END	



# 6

# BASIC INSTRUCTIONS

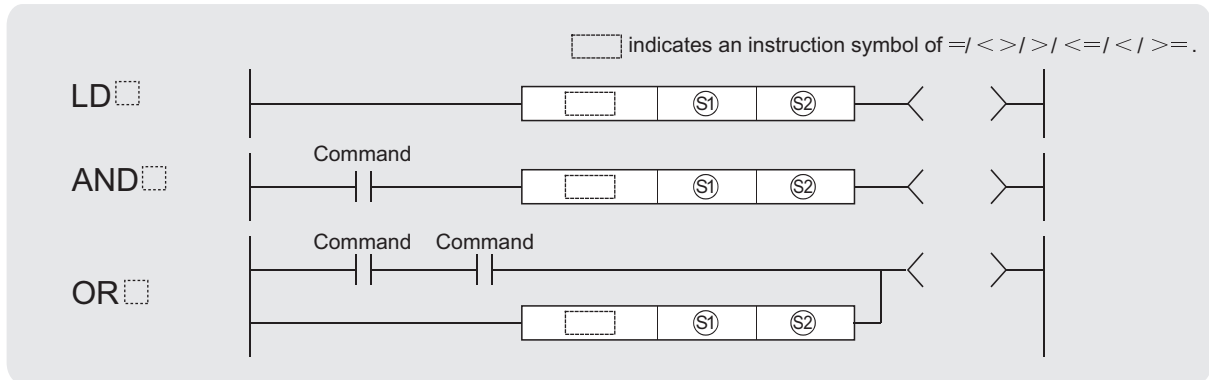
---

Category	Processing Details	Reference Section
Comparison operation instruction	Compares data to data.	Section 6.1
Arithmetic operation instruction	Adds, subtracts, multiplies, divides, increments, or decrements data with other data.	Section 6.2
Data conversion instructions	Converts data types.	Section 6.3
Data transfer instruction	Transmits designated data.	Section 6.4
Program branch instruction	Program jumps.	Section 6.5
Program run control instruction	Enables and disables program interrupts.	Section 6.6
I/O refresh instruction	Refreshes bit devices.	Section 6.7
Other convenient instructions	Up/down counters, teaching timers, special function timers, rotary table shortest direction controls, etc.	Section 6.8

# 6.1 Comparison Operation Instructions

## 6.1.1 BIN 16-bit data comparisons (=,<>,>,<=,<,>=)

Basic High performance Process Redundant Universal



S1, S2 : Data for comparison or head number of the devices where the data for comparison is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
S1					○				—
S2					○				—

### ★ Function

- (1) Treats BIN 16-bit data from device designated by S1 and BIN 16-bit data from device designated by S2 as an a normally-open contact, and performs comparison operation.
- (2) The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol in <span style="border: 1px dashed black; padding: 2px;"> </span>	Condition	Comparison Operation Result	Instruction Symbol in <span style="border: 1px dashed black; padding: 2px;"> </span>	Condition	Comparison Operation Result
=	S2 = S1	Continuity	=	S1 ≠ S2	Non-continuity
<>	S1 ≠ S2		<>	S2 = S1	
>	S1 > S2		>	S1 ≧ S2	
<=	S1 ≧ S2		<=	S1 > S2	
<	S1 < S2		<	S1 ≧ S2	
>=	S1 ≧ S2		>=	S1 < S2	

- (3) When S1 and S2 are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b15) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.

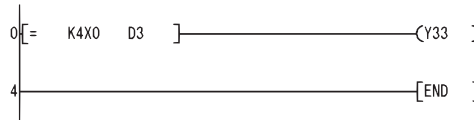
## ! Operation Error

- (1) There are no operation errors associated with the =, <>, >, <=, <, or >= instruction.

## Program Example

- (1) The following program compares the data at X0 to XF with the data at D3, and turns Y33 ON if the data is identical.

[Ladder Mode]

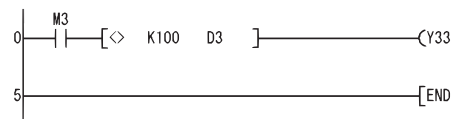


[List Mode]

Step	Instruction	Device
0	LD=	K4X0 D3
3	OUT	Y33
4	END	

- (2) The following program compares BIN value K100 to the data at D3, and establishes continuity if the data in D3 is something other than 100.

[Ladder Mode]

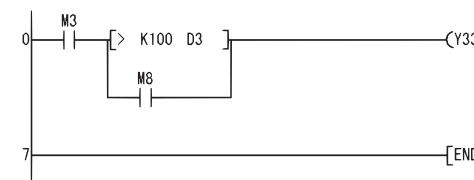


[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND<>	K100 D3
4	OUT	Y33
5	END	

- (3) The following program compares the BIN value 100 with the data at D3, and establishes continuity if the D3 data is less than 100.

[Ladder Mode]

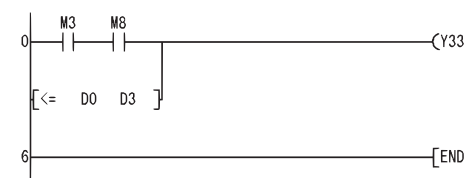


[List Mode]

Step	Instruction	Device
0	LD	M3
1	LD>	K100 D3
4	OR	M8
5	AND	
6	OUT	Y33
7	END	

- (4) The following program compares the data in D0 and D3, and if the data in D0 is equal to or less than the data in D3, establishes continuity.

[Ladder Mode]

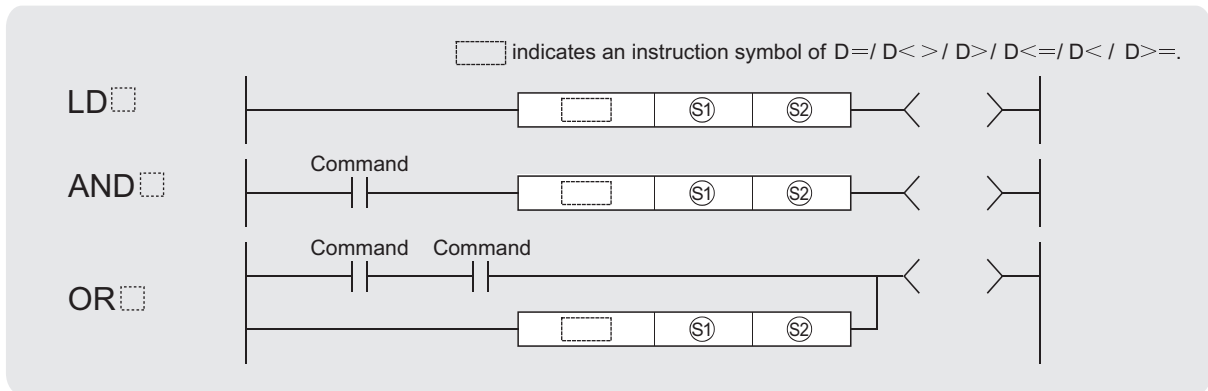


[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND	M8
2	OR<=	D0 D3
5	OUT	Y33
6	END	

## 6.1.2 BIN 32-bit data comparisons (D=,D<>,D>,D<=,D<,D>=)

Basic High performance Process Redundant Universal



S1, S2 : Data for comparison or head number of the devices where the data for comparison is stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
S1									—
S2									—

### ★ Function

- (1) Treats BIN 32-bit data from device designated by S1 and BIN 32-bit data from device designated by S2 as an a normally-open contact, and performs comparison operation.
- (2) The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol in □	Condition	Comparison Operation Result	Instruction Symbol in □	Condition	Comparison Operation Result
D=	S2 = S1	Continuity	D=	S1 ≠ S2	Non-continuity
D<>	S1 ≠ S2		D<>	S2 = S1	
D>	S1 > S2		D>	S1 ≦ S2	
D<=	S1 ≦ S2		D<=	S1 > S2	
D<	S1 < S2		D<	S1 ≧ S2	
D>=	S1 ≧ S2		D>=	S1 < S2	

- (3) When S1 and S2 are assigned by a hexadecimal constant and the numerical value (8 to F) whose most significant bit (b31) is "1" is designated as a constant, the value is considered as a negative BIN value in comparison operation.
- (4) Data used for comparison should be designated by a 32-bit instruction (DMOV instruction, etc.).  
If designation is made with a 16-bit instruction (MOV instruction, etc.), comparisons of large and small values cannot be performed correctly.

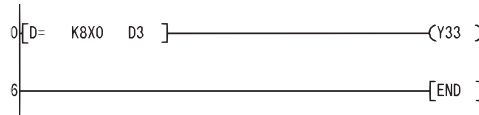
## ! Operation Error

- (1) There are no operation errors associated with the D=, D<>, D>, D<=, D< or D>= instruction.

## Program Example

- (1) The following program compares the data at X0 to X1F with the data at D3 and D4, and turns Y33 ON, if the data at X0 to X1F and the data at D3 and D4 match.

[Ladder Mode]

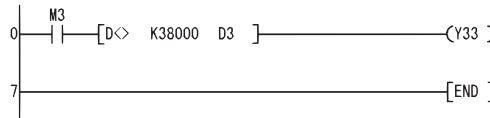


[List Mode]

Step	Instruction	Device
0	LDD=	K8X0 D3
5	OUT	Y33
6	END	

- (2) The following program compares BIN value K38000 to the data at D3, and D4, and establishes continuity if the data in D3 and D4 is something other than 38000.

[Ladder Mode]

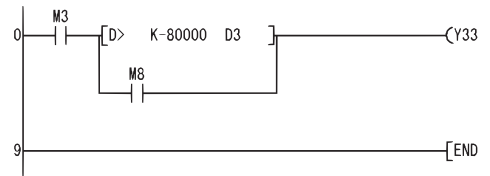


[List Mode]

Step	Instruction	Device
0	LD	M3
1	ANDD<>	K38000 D3
6	OUT	Y33
7	END	

- (3) The following program compares BIN value K-80000 to the data at D3 and D4, and establishes continuity if the data in D3 and D4 is less than -80000.

[Ladder Mode]

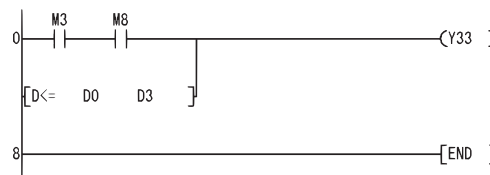


[List Mode]

Step	Instruction	Device
0	LD	M3
1	LDD>	K-80000 D3
6	OR	M8
7	ANB	
8	OUT	Y33
9	END	

- (4) The following program compares the data in D0 and D1 with the data in D3 and D4, and establishes continuity if the data in D0 and D1 is equal to or less than the data in D3 and D4.

[Ladder Mode]



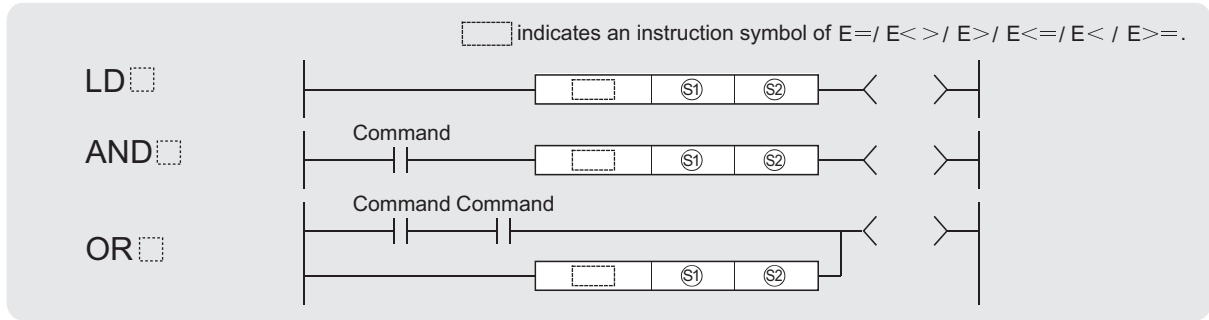
[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND	M8
2	ORD<=	D0 D3
7	OUT	Y33
8	END	

## 6.1.3 Floating decimal point data comparisons (Single precision) (E=,E<>,E>,E<=,E<,E>=)



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ1, Ⓢ2 : Data for comparison or head number of the devices where the data for comparison is stored (real number)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○	—	○	—	—	○	—	
Ⓢ2	—	○	—	○	—	—	○	—	

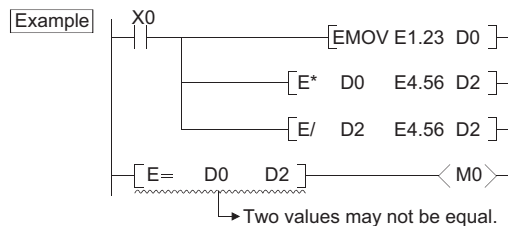
\*1: Available only in multiple Universal model QCPU

- The 32-bit floating decimal point data from device designated by Ⓢ1 and 32-bit floating decimal point data from device designated by Ⓢ2 as a normally-open contact, and performs comparison operation.
- The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol in □	Condition	Comparison Operation Result	Instruction Symbol in □	Condition	Comparison Operation Result
E=	Ⓢ2 = Ⓢ1	Continuity	E=	Ⓢ1 ≠ Ⓢ2	Non-continuity
E<>	Ⓢ1 ≠ Ⓢ2		E<>	Ⓢ2 = Ⓢ1	
E>	Ⓢ1 > Ⓢ2		E>	Ⓢ1 ≦ Ⓢ2	
E<=	Ⓢ1 ≦ Ⓢ2		E<=	Ⓢ1 > Ⓢ2	
E<	Ⓢ1 < Ⓢ2		E<	Ⓢ1 ≧ Ⓢ2	
E>=	Ⓢ1 ≧ Ⓢ2		E>=	Ⓢ1 < Ⓢ2	

### POINT

Note that use of the E= instruction can on occasion result in situations where errors cause the two values to not be equal.



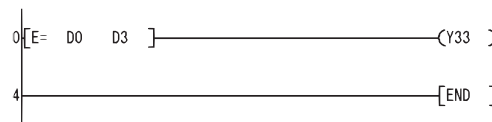
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is  $-0$ . \*1  
(For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
  - \*1: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to Section 3.2.4.
  - The value of the specified device is outside the following range. (For the Universal model QCPU)  
 $0, 2^{-126} \leq | \text{value of specified device} | < 2^{128}$  (Error code: 4140)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
(For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program compares 32-bit floating decimal point real number data at D0 and D1 to 32-bit floating decimal point real number data at D3 and D4.

[Ladder Mode]

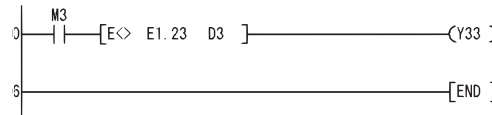


[List Mode]

Step	Instruction	Device
0	LDE=	D0 D3
3	OUT	Y33
4	END	

- (2) The following program compares the floating decimal point real number 1.23 to the 32-bit floating decimal point real number data at D3 and D4.

[Ladder Mode]

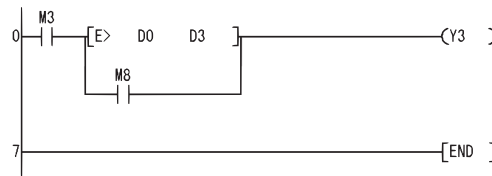


[List Mode]

Step	Instruction	Device
0	LD	M3
1	ANDE<>	E1.23 D3
5	OUT	Y33
6	END	

- (3) The following program compares 32-bit floating decimal point real number data at D0 and D1 to 32-bit floating decimal point real number data at D3 and D4.

[Ladder Mode]

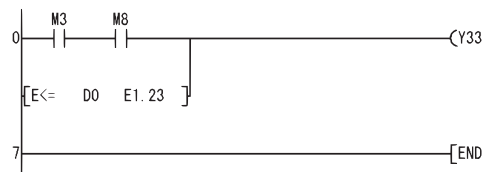


[List Mode]

Step	Instruction	Device
0	LD	M3
1	LDE>	D0 D3
4	OR	M8
5	ANB	
6	OUT	Y3
7	END	

- (4) The following program compares the 32-bit floating decimal point data at D0 and D1 to the floating decimal point real number 1.23.

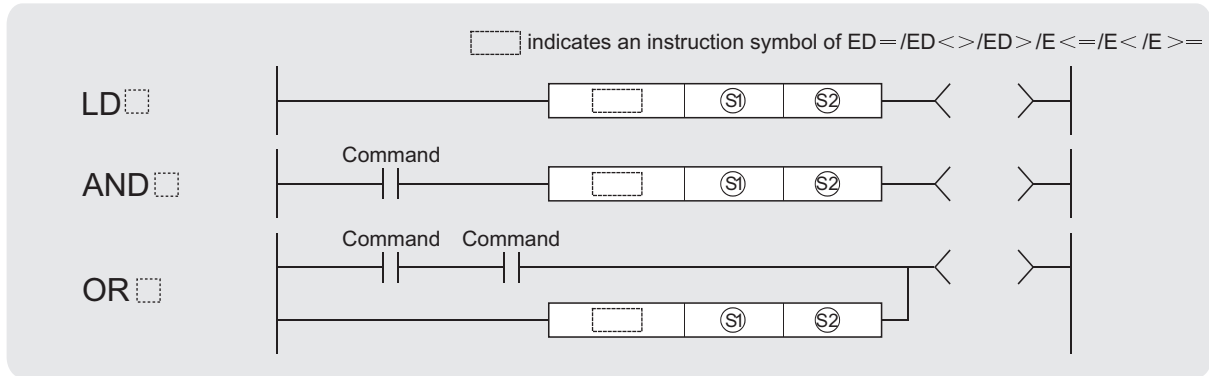
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND	M8
2	ORE<=	D0 E1.23
6	OUT	Y33
7	END	

## 6.1.4 Floating decimal point data comparisons (Double precision) (ED=,ED<>,ED>,ED<=,ED<,ED>=)



Ⓢ1, Ⓢ2: Data for comparison or head number of the devices where the data for comparison is stored (real number)

Setting Data	Internal Devices		R, ZR	JdG		UdG	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○	○	—	○	—	—	○	—
Ⓢ2	—	○	○	—	○	—	—	○	—

### ★ Function

- The 64-bit floating decimal point real number from device designated by Ⓢ1 and 64-bit floating decimal point real number from device designated by Ⓢ2 as A normally-open contact, and performs comparison operation.
- The results of the comparison operations for the individual instructions are as follows:

Instruction Symbol in □	Condition	Comparison Operation Result	Instruction Symbol in □	Condition	Comparison Operation Result
ED=	Ⓢ2 = Ⓢ1	Continuity	ED=	Ⓢ1 ≠ Ⓢ2	Non-continuity
ED<>	Ⓢ1 ≠ Ⓢ2		ED<>	Ⓢ2 = Ⓢ1	
ED>	Ⓢ1 > Ⓢ2		ED>	Ⓢ1 ≤ Ⓢ2	
ED<=	Ⓢ1 ≤ Ⓢ2		ED<=	Ⓢ1 > Ⓢ2	
ED<	Ⓢ1 < Ⓢ2		ED<	Ⓢ1 ≥ Ⓢ2	
ED>=	Ⓢ1 ≥ Ⓢ2		ED>=	Ⓢ1 < Ⓢ2	



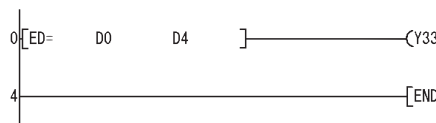
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)

## Program Example

- (1) The following program compares 64-bit floating decimal point real number data at D0 to D3 with 64-bit floating decimal point real number data at D4 to D7.

[Ladder Mode]

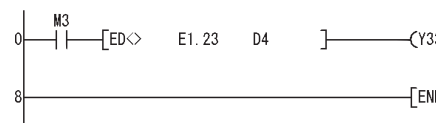


[List Mode]

Step	Instruction	Device
0	LDED=	D0 D4
3	OUT	Y33
4	END	

- (2) The following program compares the floating decimal point real number 1.23 with the 64-bit floating decimal point real number data at D4 to D7.

[Ladder Mode]

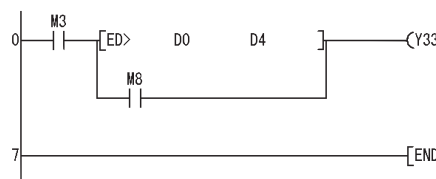


[List Mode]

Step	Instruction	Device
0	LD	M3
1	ANDED<	E1.23 D4
7	OUT	Y33
8	END	

- (3) The following program compares 64-bit floating decimal point real number data at D0 to D3 with 64-bit floating decimal point real number data at D4 to D7.

[Ladder Mode]

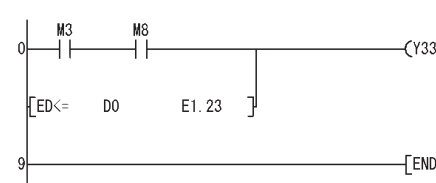


[List Mode]

Step	Instruction	Device
0	LD	M3
1	LDED>	D0 D4
4	OR	M8
5	ANB	
6	OUT	Y33
7	END	

- (4) The following program compares the 64-bit floating decimal point data at D0 to D3 with the floating decimal point real number 1.23.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND	M8
2	ORED<=	D0 E1.23
8	OUT	Y33
9	END	

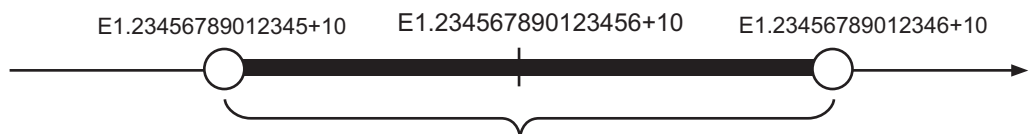
## Caution

- (1) Since the number of digits of the real number that can be input by GX Developer is up to 15 digits, the comparison with the real number whose number of significant digits is 16 or more cannot be made by the instruction shown in this section.

When judging match/mismatch with the real number whose significant digits is 16 or more by the instruction in this section, compare it with the approximate values of the real number to be compared and judge by the sizes.

**Example** When judging the match of E1.23456789012345+10 (Number of significant digits is 16) and the double-precision floating-point data.

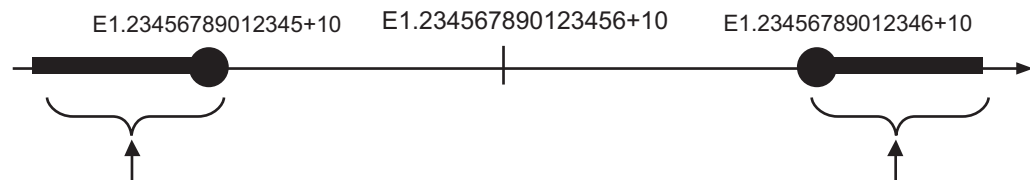
```
[ED<  E1.23456789012345+10  D0  ][ED<  D0  E1.23456789012346+10  ]-(Y10 )
```



Whether D0 to D3 is within this range is checked.(Values on boundaries are excluded.)

**Example** When judging the mismatch of E1.234567890123456+10 (Number of significant digits is 16) and the double-precision floating-point data.

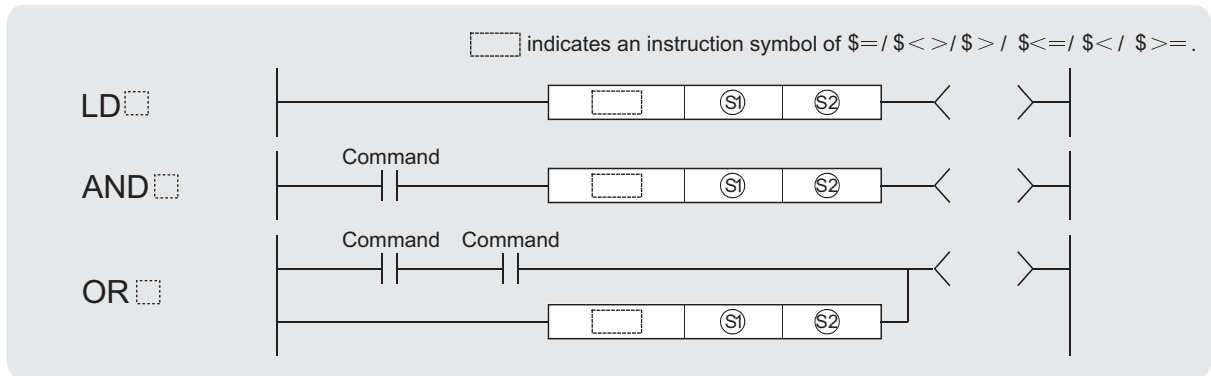
```
[ED<= D0  E1.23456789012345+10  ]-(Y20 )
[ED>= D0  E1.23456789012346+10  ]
```



Whether D0 to D3 is within this range is checked.(Values on boundaries are included.)

## 6.1.5 Character string data comparisons (\$=, \$<>, \$>, \$<=, \$<, \$>=)

Basic ~~X~~ High performance Process Redundant Universal

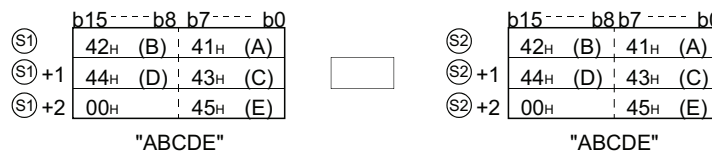


Ⓢ1, Ⓢ2: Data for comparison or head number of the devices where the data for comparison is stored (character string)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		○	—
Ⓢ2	—	○				—		○	—

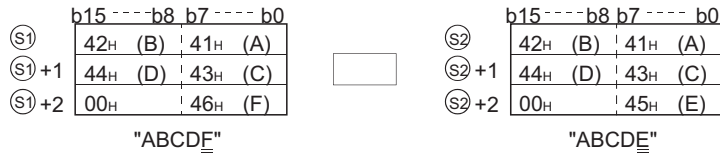
### ★ Function

- (1) Compares the character string data designated by Ⓢ1 with the character string data designated by Ⓢ2 as a normally-open contact.
- (2) A comparison operation involves the character-by-character comparison of the ASCII code of the first character in the character string.
- (3) The character string data of Ⓢ1 and Ⓢ2 for comparison refers to the data stored at the range from the designated device number to the device number where "00H" code is stored.
  - (a) If all character strings match, the comparison result will be matched.



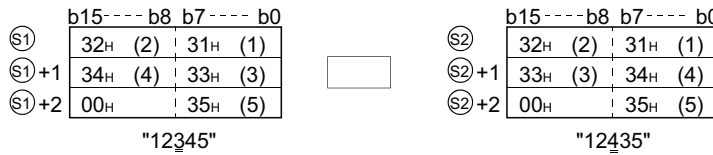
Instruction Symbol in □	Comparison Operation Result	Instruction Symbol in □	Comparison Operation Result
\$=	Continuity	\$<=	Continuity
\$<>	Non-continuity	\$<	Non-continuity
\$>	Non-continuity	\$>=	Continuity

(b) If the character strings are different, the character string with the larger character code will be the larger.



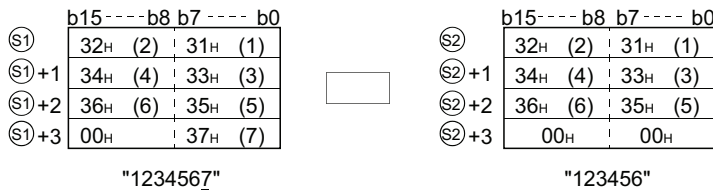
Instruction Symbol in <input type="checkbox"/>	Comparison Operation Result	Instruction Symbol in <input type="checkbox"/>	Comparison Operation Result
\$=	Non-continuity	\$<=	Non-continuity
\$<>	Continuity	\$<	Non-continuity
\$>	Continuity	\$>=	Continuity

(c) If the character strings are different, the first different sized character code will determine whether the character string is larger or smaller.



Instruction Symbol in <input type="checkbox"/>	Comparison Operation Result	Instruction Symbol in <input type="checkbox"/>	Comparison Operation Result
\$=	Non-continuity	\$<=	Continuity
\$<>	Continuity	\$<	Continuity
\$>	Non-continuity	\$>=	Non-continuity

(4) If the character strings designated by Ⓢ1 and Ⓢ2 are of different lengths, the data with the longer character string will be larger.



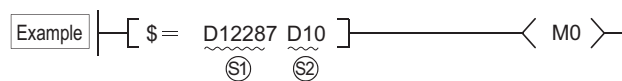
Instruction Symbol in <input type="checkbox"/>	Comparison Operation Result	Instruction Symbol in <input type="checkbox"/>	Comparison Operation Result
\$=	Non-continuity	\$<=	Non-continuity
\$<>	Continuity	\$<	Non-continuity
\$>	Continuity	\$>=	Continuity

## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The code "00H" does not exist within the range of the relevant device, starting from the device number designated by S1 and S2. (Error code: 4101)
  - The character string of S1 and S2 exceeds 16383 characters. (Error code: 4101)

### POINT

The character string data comparison instruction checks the device range while comparing the designated character string data. For this reason, if the "00H" code does not exist in the relevant device range, the instruction outputs the comparison result instead of returning an operation error when no match of characters is detected.



Data of S1

D12287	B	A
W0	00H	C

Data of S2

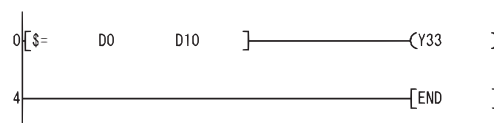
D10	Z	A
D11	00H	C

If S1 and S2 data are as shown above, the second character of S1 does not match with that of S2, and the comparison result is expressed as S1 ≠ S2 (the operation result is "non-conductive"). Though the "00H" code is not included within the S1 device range, no operation error is returned, because the no-match is detected at D12287, which is within the device range.

## Program Example

- (1) The following program compares character strings stored following D0 and characters following D10.

[Ladder Mode]

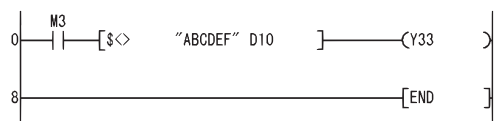


[List Mode]

Step	Instruction	Device
0	LD\$=	D0 D10
3	OUT	Y33
4	END	

- (2) The following program compares the character string "ABCDEF" with the character string stored following D10.

[Ladder Mode]

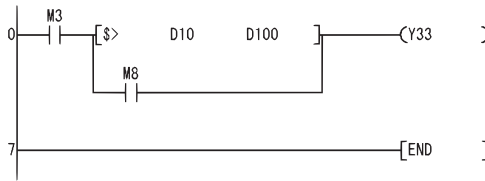


[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND\$<>	"ABCDEF" D10
7	OUT	Y33
8	END	

- (3) The following program compares the character string stored following D10 with the character string stored following D100.

[Ladder Mode]

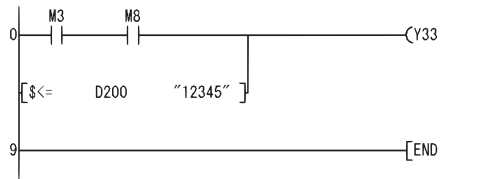


[List Mode]

Step	Instruction	Device
0	LD	M3
1	LD\$>	D10 D100
4	OR	M8
5	ANB	
6	OUT	Y33
7	END	

- (4) The following program compares the character string stored following D200 with the character string "12345".

[Ladder Mode]

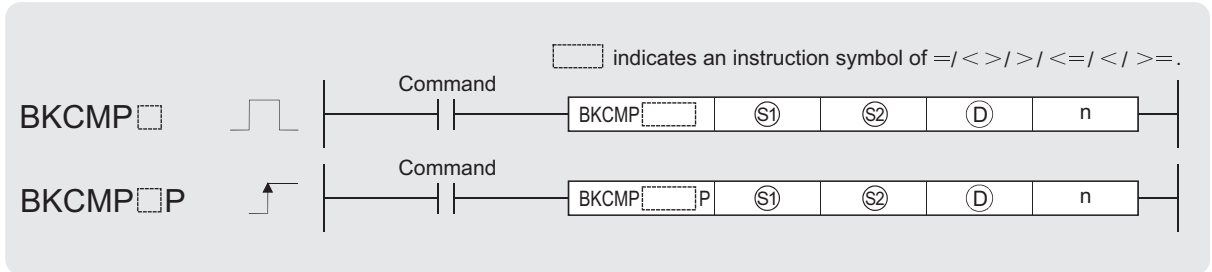


[List Mode]

Step	Instruction	Device
0	LD	M3
1	AND	M8
2	OR\$<=	D200 "12345"
8	OUT	Y33
9	END	

# 6.1.6 BIN block data comparisons (BKCMP □, BKCMP □ P)

Basic High performance Process Redundant Universal

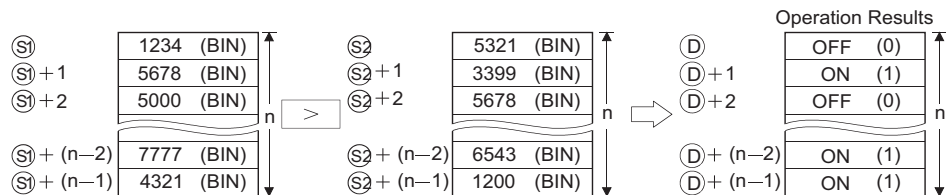


- Ⓢ<sub>1</sub> : Data to be compared or head number of the devices where the data to be compared is stored (BIN 16 bits)
- Ⓢ<sub>2</sub> : Head number of the devices where the comparison data is stored (BIN 16 bits)
- ⓓ : Head number of the devices where the comparison operation result will be stored (bits)
- n : Number of comparison data blocks (BIN 16 bits)

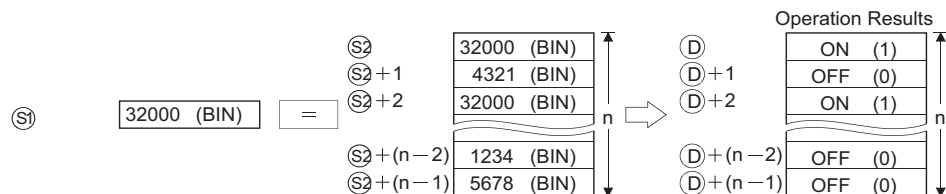
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○				—		○	—
Ⓢ <sub>2</sub>	—	○				—		—	—
ⓓ	○	○				—		—	—
n	○	○				○		○	—

## ★ Function

- (1) Compares BIN 16-bit data the nth point from the device number designated by Ⓢ<sub>1</sub> with BIN 16-bit data the nth point from the device number designated by Ⓢ<sub>2</sub>, and stores the result from the device designated by ⓓ onward.
  - (a) If the comparison condition has been met, the device designated by ⓓ will be turned ON.
  - (b) If the comparison condition has not been met, the device designated by ⓓ will be turned OFF.



- (2) The comparison operation is conducted in 16-bit units.
- (3) The constant designated by Ⓢ<sub>1</sub> can be between -32768 and 32767 (BIN 16-bit data).



6.1 Comparison Operation Instructions  
6.1.6 BIN block data comparisons (BKCMP □, BKCMP □ P)

6

(4) The results of the comparison operations for the individual instructions are as follows:

Instruction Symbols	Condition	Comparison Operation Result	Instruction Symbols	Condition	Comparison Operation Result
BKCMP=	$S2 = S1$	ON (1)	BKCMP=	$S1 \neq S2$	OFF (0)
BKCMP<>	$S1 \neq S2$		BKCMP<>	$S2 = S1$	
BKCMP>	$S1 > S2$		BKCMP>	$S1 \leq S2$	
BKCMP<=	$S1 \leq S2$		BKCMP<=	$S1 > S2$	
BKCMP<	$S1 < S2$		BKCMP<	$S1 \geq S2$	
BKCMP>=	$S1 \geq S2$		BKCMP>=	$S1 < S2$	

(5) If all comparison results stored n points from  $\textcircled{D}$  are ON (1), SM704 (block comparison signal) goes ON.



## Operation Error

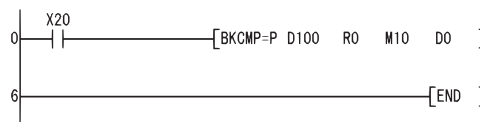
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range of the device n points from a device designated by  $\textcircled{S1}$ ,  $\textcircled{S2}$  or  $\textcircled{D}$  exceeds the relevant device. (Error code: 4101)
  - The device range for n points starting from the device designated by  $\textcircled{S1}$  overlaps with the device range for n points starting from the device designated by  $\textcircled{D}$ . (Error code: 4101)
  - The device range for n points starting from the device designated by  $\textcircled{S2}$  overlaps with the device range for n points starting from the device designated by  $\textcircled{D}$ . (Error code: 4101)



## Program Example

- (1) The following program compares, when X20 is turned ON, the data stored at D100 to D103 with the data stored at R0 to R3 and stores the operation result into the area starting from M10.

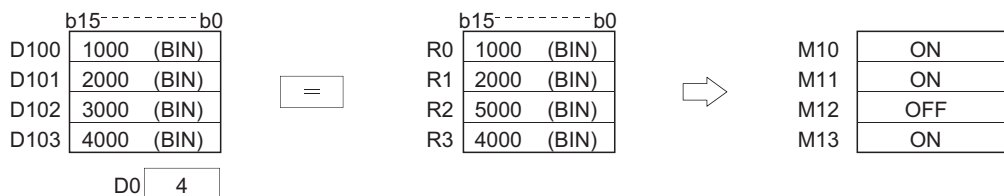
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKCMP=P	D100 R0 M10 D0
6	END	

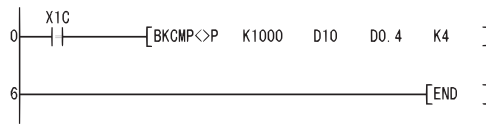
[Operation]





- (2) The following program compares, when X1C is turned ON, the constant K1000 with the data stored at D10 to D13, and stores the operation result at b4 to b7 in D0.

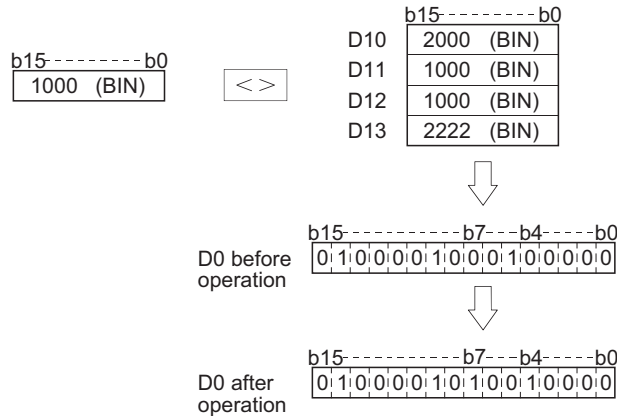
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	BKCM P > P	K1000 D10 D0.4 K4
6	END	

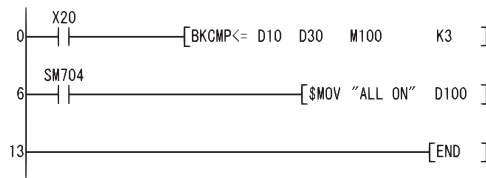
[Operation]



- (3) The following program compares, when X20 is turned ON, the data at D10 to D12 with the data at D30 to D32, and stores the operation result into the area starting from M100.

The following program transfers the character string "ALL ON" to D100 onward when all devices from M100 onward have reached the 1 "ON" state.

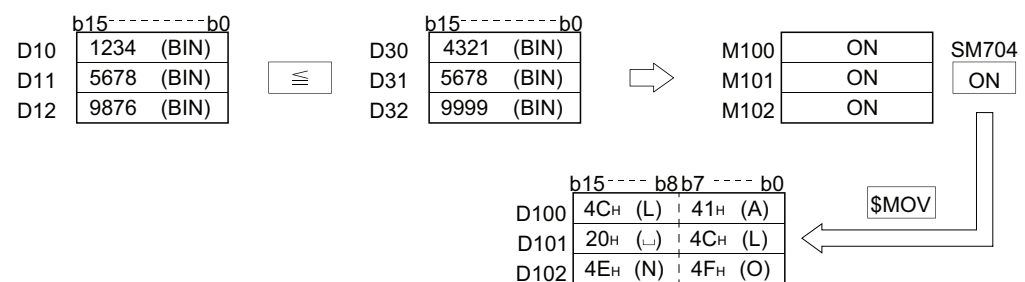
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKCM P <=	D10 D30 M100 K3
6	LD	SM704
7	\$MOV	"ALL ON" D100
13	END	

[Operation]

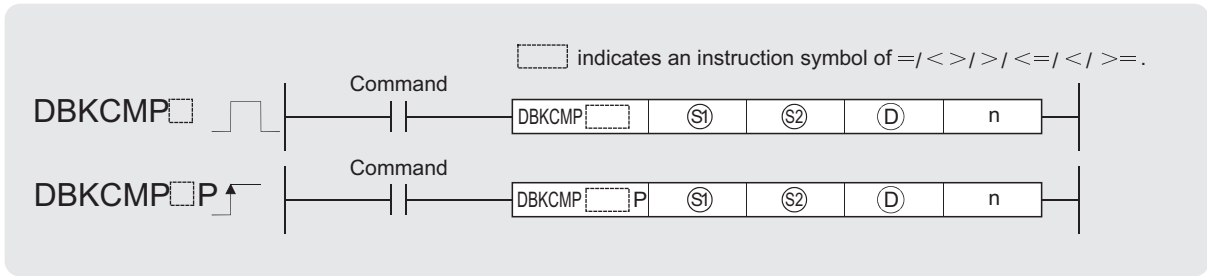


# 6.1.7 BIN 32-bit block data comparisons (DBKCMP □, DBKCMP □ P)



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.

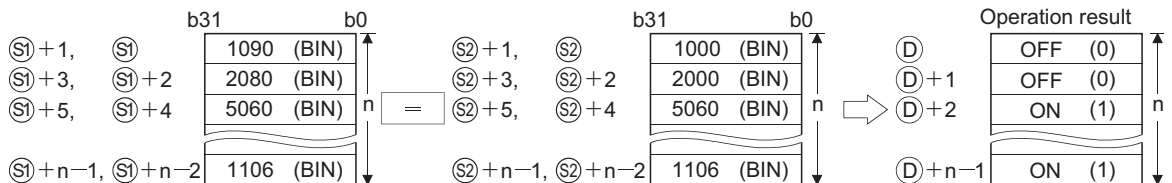


- Ⓢ1 : Data to be compared or head number of the devices where the data to be compared are stored (BIN 32 bits)
- Ⓢ2 : Head number of the devices where the comparison data are stored (BIN 32 bits)
- Ⓧ : Head number of the devices where the comparison operation result will be stored (bits)
- n : Number of comparison data blocks (BIN 16 bits)

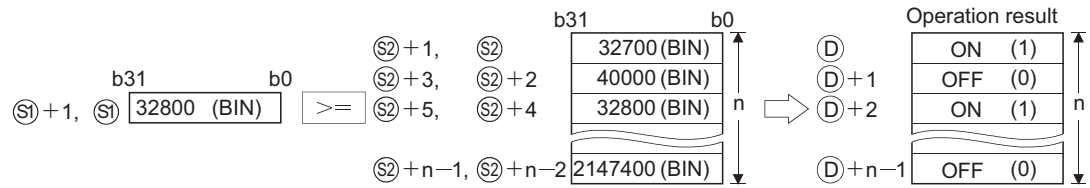
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		○	—
Ⓢ2	—	○				—		—	—
Ⓧ	○	○				—		—	—
n	○	○				○		○	—

## ★ Function

- (1) This instruction compares BIN 32-bit data stored in n-point devices starting from the device specified by Ⓢ1 with BIN 32-bit data stored in n-point devices starting from the device specified by a constant and Ⓢ2 and then stores the result into the nth device specified by Ⓧ and up.
  - (a) If the comparison condition has been met, the corresponding devices specified by Ⓧ will be turned on.
  - (b) If the comparison condition has not been met, the corresponding devices specified by Ⓧ will be turned off.



- (2) The comparison operation is executed in 32-bit units.
- (3) The constant in the device specified by  $\textcircled{S1}$  can be between  $-2147483648$  and  $2147483647$  (BIN 32-bit data).



- (4)  $\textcircled{D}$  specifies out of the device range of n-point devices starting from the device specified by  $\textcircled{S1}$  and  $\textcircled{S2}$ .
- (5) The following table shows the results of the comparison operations for each individual instruction.

Instruction Symbols	Condition	Comparison Operation Result	Instruction Symbols	Condition	Comparison Operation Result
DBKCMP=	$\textcircled{S2} = \textcircled{S1}$	ON (1)	DBKCMP=	$\textcircled{S1} \neq \textcircled{S2}$	OFF (0)
DBKCMP<>	$\textcircled{S1} \neq \textcircled{S2}$		DBKCMP<>	$\textcircled{S2} = \textcircled{S1}$	
DBKCMP>	$\textcircled{S1} > \textcircled{S2}$		DBKCMP>	$\textcircled{S1} \leq \textcircled{S2}$	
DBKCMP<=	$\textcircled{S1} \leq \textcircled{S2}$		DBKCMP<=	$\textcircled{S1} > \textcircled{S2}$	
DBKCMP<	$\textcircled{S1} < \textcircled{S2}$		DBKCMP<	$\textcircled{S1} \geq \textcircled{S2}$	
DBKCMP>=	$\textcircled{S1} \geq \textcircled{S2}$		DBKCMP>=	$\textcircled{S1} < \textcircled{S2}$	

- (6) If all comparison results stored into the devices starting from the device specified by  $\textcircled{D}$  to nth device are on(1), or one of the results is off(2), the special relays will be on or off in accordance with the conditions as follows.

No.	Number	When all results of comparison operations are on(1)			When results of comparison operations have a result of off(0)		
		Initial execution/Scan	Interrupt (other than I45)/Fixed scan execution	Interrupt(I45)	Initial execution/Scan	Interrupt (other than I45)/Fixed scan execution	Interrupt(I45)
1	SM704	ON	ON	ON	OFF	OFF	OFF
2	SM716	ON	—	—	OFF	—	—
3	SM717	—	ON	—	—	OFF	—
4	SM718	—	—	ON	—	—	OFF

In a standby program, a special relay depending on the caller program turns on or off.

- (7) If the value specified by n is 0, the instruction will be not processed.

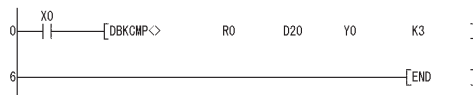
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- A negative value is specified for n. (Error code: 4100)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S1}$ ,  $\textcircled{S2}$  or  $\textcircled{D}$  exceeds the specified device range. (Error code: 4101)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S1}$  overlaps with the range of the n-point devices starting from the device specified by  $\textcircled{D}$ . (Error code: 4101)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S2}$  overlaps with the range of the n-point devices starting from the device specified by  $\textcircled{D}$ . (Error code: 4101)

## Program Example

- (1) The following program compares the value data stored at R0 to R5 with the value data stored at D20 to D25, and then stores the operation result into Y0 to Y2, when M0 is turned on,

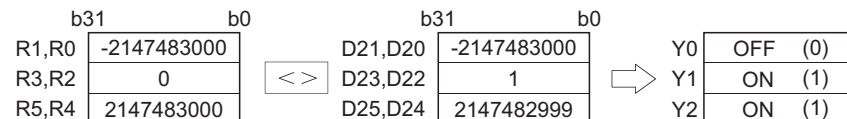
[Ladder Mode]



[List Mode]

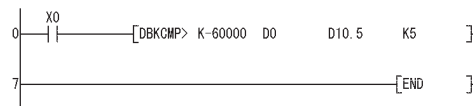
Step	Instruction	Device
0	LD	X0
1	DBKCMPC<>	R0 D20 Y0 K3
6	END	

[Operation]



- (2) The following program compares the constant with the value data stored at D0 to D9, and then stores the operation result into D10.5 to D10.9, when M0 is turned on,

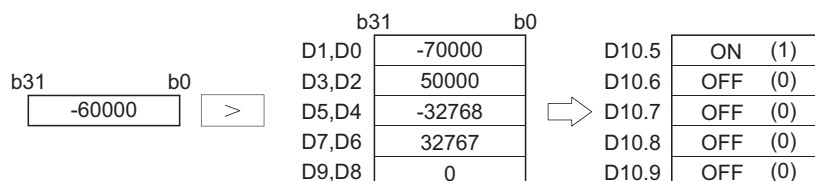
[Ladder Mode]



[List Mode]

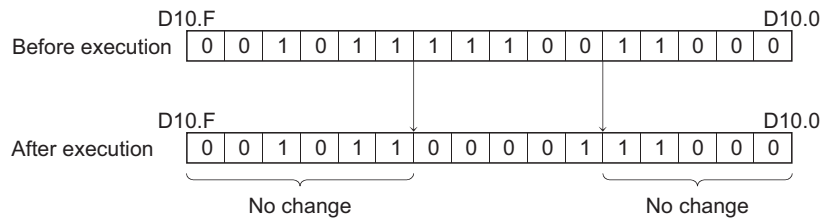
Step	Instruction	Device
0	LD	M0
1	DBKCMPC>=	K-60000 D0 D10.5 K5
7	END	

[Operation]



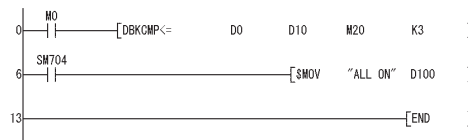
**POINT**

When certain bits are specified in a word device, bits other than the certain bits that store the operation result do not change.



- (3) The following program compares the value data stored at D0 to D5 with the value data stored at D10 to D15, and then stores the operation result into M20 to M22, when M0 is turned on. Also, the program transfers the character string "ALL ON" to D100 and up when all devices from M20 to M22 have reached the on status.

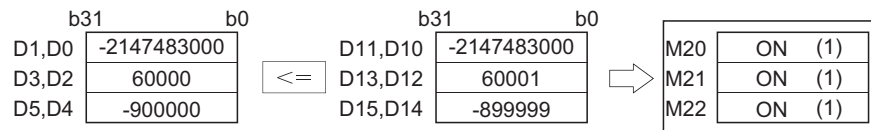
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DBKCOMP<=	D0 D10 M20 K3
6	LD	SM704
7	\$MOV	"ALLON" D100
13	END	

[Operation]



When all operation results are on(1), the special relays corresponding to each program turn on(1).  
 (Since this program examples refer to scan programs, SM704 and SM716 turn on(1), SM7171 and SM718 do not change in the scan program)

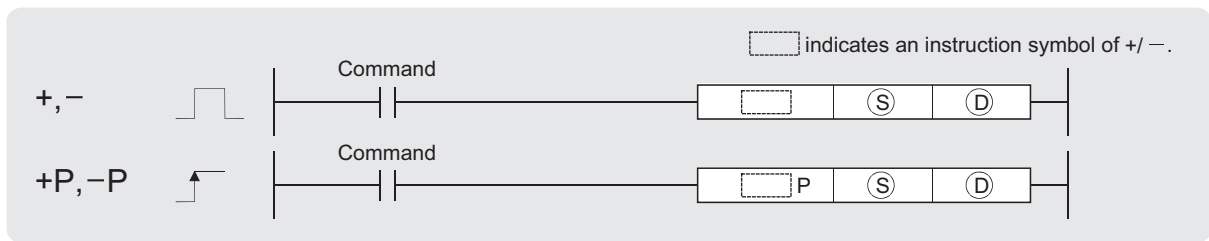
SM704	ON	(1)
SM716	ON	(1)
SM717	OFF	(0)
SM718	OFF	(0)

## 6.2 Arithmetic Operation Instructions

### 6.2.1 BIN 16-bit addition and subtraction operations (+(P),-(P))

Basic High performance Process Redundant Universal

1 When two data are set ( $\textcircled{D} + \textcircled{S} \rightarrow \textcircled{D}$ ,  $\textcircled{D} - \textcircled{S} \rightarrow \textcircled{D}$ )



$\textcircled{S}$  : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 16 bits)

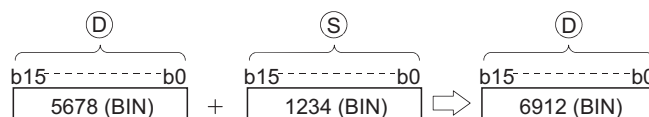
$\textcircled{D}$  : Head number of the devices where the data to be added to/subtracted from is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
$\textcircled{S}$								○	—
$\textcircled{D}$								—	—

## ★ Function

+

- (1) Adds 16-bit BIN data designated by  $\textcircled{D}$  to 16-bit BIN data designated by  $\textcircled{S}$  and stores the result of the addition at the device designated by  $\textcircled{D}$ .



- (2) Values for  $\textcircled{S}$  and  $\textcircled{D}$  can be designated between  $-32768$  and  $32767$  (BIN, 16 bits).

- (3) The judgment of whether data is positive or negative is made by the most significant bit (b15).

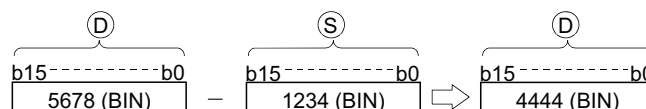
- 0: Positive
- 1: Negative

- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.

·  $K32767 + K2 \rightarrow K-32767$  ..... Since bit 15 value is "1",  
 (7FFFH) (0002H) (8001H) result of operation takes a negative value.

·  $K-32768 + K-2 \rightarrow K32766$  ..... Since bit 15 value is "0",  
 (8000H) (FFFEH) (7FFEH) result of operation takes a positive value.

- (1) Subtracts 16-bit BIN data designated by  $\textcircled{S}$  from 16-bit BIN data designated by  $\textcircled{D}$  and stores the result of the subtraction at the device designated by  $\textcircled{D}$ .



- (2) Values for  $\textcircled{S}$  and  $\textcircled{D}$  can be designated between  $-32768$  and  $32767$  (BIN, 16 bits).
- (3) The judgment of whether data is positive or negative is made by the most significant bit (b15).
- 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.

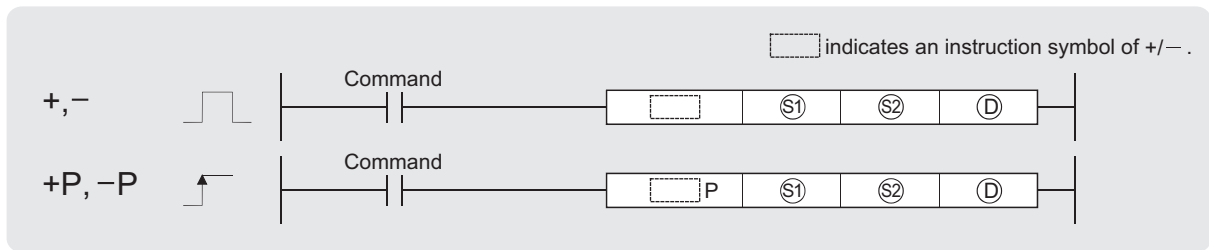
·  $K-32768 - K2 \longrightarrow K32766$  ..... Since bit 15 value is "0",  
 (8000H) (0002H) (7FFEh) result of operation takes a positive value.

·  $K32767 - K-2 \longrightarrow K-32767$  ..... Since bit 15 value is "1",  
 (7FFFH) (FFFEh) (8001H) result of operation takes a negative value.

## Operation Error

- (1) There are no operation errors associated with the +(P) or -(P) instruction.

2 When three data are set (S1 + S2 → D, S1 - S2 → D)



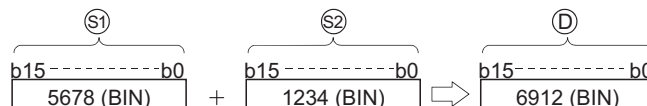
- Ⓢ<sub>1</sub> : Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BIN 16 bits)
- Ⓢ<sub>2</sub> : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 16 bits)
- ⓓ : Head number of the devices where the addition/subtraction operation result will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>					○			○	—
Ⓢ <sub>2</sub>					○			○	—
ⓓ					○			—	—

## ★ Function

+

- (1) Adds 16-bit BIN data designated by Ⓢ<sub>1</sub> to 16-bit BIN data designated by Ⓢ<sub>2</sub> and stores the result of the addition at the device designated by ⓓ .



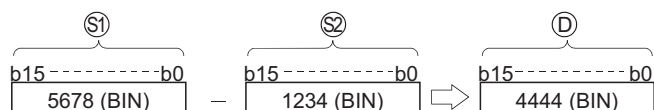
- (2) Values for Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub> ⓓ and can be designated between ⓓ - 32768 and 32767 (BIN, 16 bits).
- (3) The judgment of whether data is positive or negative is made by the most significant bit (b15).
- 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.

· K32767 +K2 → K-32767 ..... Since bit 15 value is "1",  
 (7FFFH) (0002H) (8001H) result of operation takes a negative value.

· K-32768 +K-2 → K32766 ..... Since bit 15 value is "0",  
 (8000H) (FFFEH) (7FFEH) result of operation takes a positive value.



- (1) Subtracts 16-bit BIN data designated by  $\text{S1}$  from 16-bit BIN data designated by  $\text{S2}$  and stores the result of the subtraction at the device designated by  $\text{D}$ .



- (2) Values for  $\text{S1}$ ,  $\text{S2}$ ,  $\text{D}$  and can be designated between  $\text{D} - 32768$  and  $32767$  (BIN, 16 bits).
- (3) The judgment of whether data is positive or negative is made by the most significant bit (b15).
- 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.
- $\text{K} - 32768 - \text{K}2 \longrightarrow \text{K}32766$  ..... Since bit 15 value is "0",  
(8000H) (0002H) (7FFE H) result of operation takes a positive value.
  - $\text{K}32767 - \text{K} - 2 \longrightarrow \text{K} - 32767$  ..... Since bit 15 value is "1",  
(7FFF H) (FFFE H) (8001 H) result of operation takes a negative value.

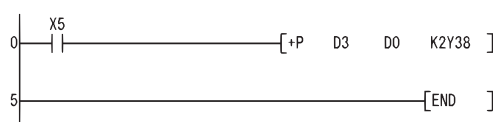
## ! Operation Error

- (1) There are no operation errors associated with the +(P) or -(P) instruction.

## Program Example

- (1) The following program adds, when X5 is turned ON, the data at D3 and D0 and outputs the operation result at Y38 to Y3F.

[Ladder Mode]

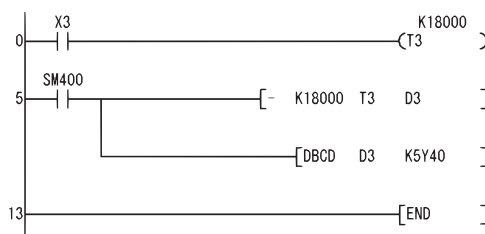


[List Mode]

Step	Instruction	Device
0	LD	X5
1	+P	D3 D0 K2Y38
5	END	

- (2) The following program outputs the difference between the set value for timer T3 and its present value in BCD to Y40 to Y53.

[Ladder Mode]



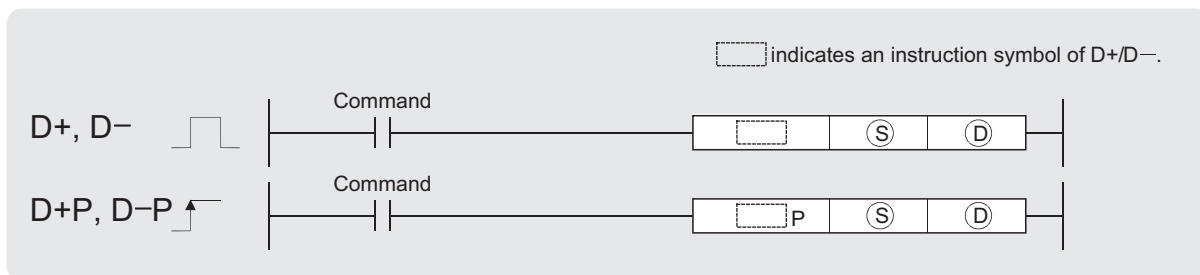
[List Mode]

Step	Instruction	Device
0	LD	X3
1	OUT	T3 K18000
5	LD	SM400
6	-	K18000 T3 D3
10	DBCD	D3 K5Y40
13	END	

## 6.2.2 BIN 32-bit addition and subtraction operations (D+(P),D-(P))

Basic High performance Process Redundant Universal

① When two data are set (( $\textcircled{D}+1, \textcircled{D}$ )+( $\textcircled{S}+1, \textcircled{S}$ )→( $\textcircled{D}+1, \textcircled{D}$ ), ( $\textcircled{D}+1, \textcircled{D}$ )-(  $\textcircled{S}+1, \textcircled{S}$  )→( $\textcircled{D}+1, \textcircled{D}$ ))



Ⓢ : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 32 bits)

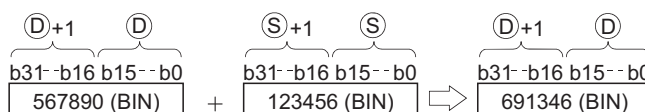
Ⓣ : Head number of the devices where the data to be added to/subtracted from is stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	JAD		UAGD	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—

### ★ Function

#### D+

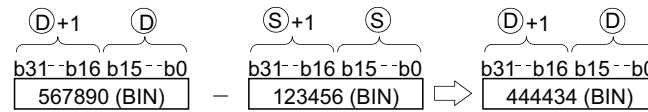
- (1) Adds 32-bit BIN data designated by  $\textcircled{D}$  to 32-bit BIN data designated by  $\textcircled{S}$ , and stores the result of the addition at the device designated by  $\textcircled{D}$ .



- (2) The values for  $\textcircled{S}$  and  $\textcircled{D}$  can be designated at between  $-2147483648$  and  $2147483647$  (BIN 32 bits).
- (3) Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
  - 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.
  - $K2147483647$  (+K2) →  $K-2147483647$  (80000001H) ..... Since bit 31 value is "1", result of operation takes a negative value.
  - $K-2147483648$  (+K-2) →  $K2147483646$  (7FFFFFFEH) ..... Since bit 31 value is "0", result of operation takes a positive value.

**D-**

- (1) Subtracts 32-bit BIN data designated by  $\textcircled{D}$  from 32-bit BIN data designated by  $\textcircled{S}$  and stores the result of the subtraction at the device designated by  $\textcircled{D}$ .



- (2) The values for  $\textcircled{S}$  and  $\textcircled{D}$  can be designated at between  $-2147483648$  and  $2147483647$  (BIN 32 bits).
- (3) Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
- 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result:  
The carry flag in this case does not go ON.

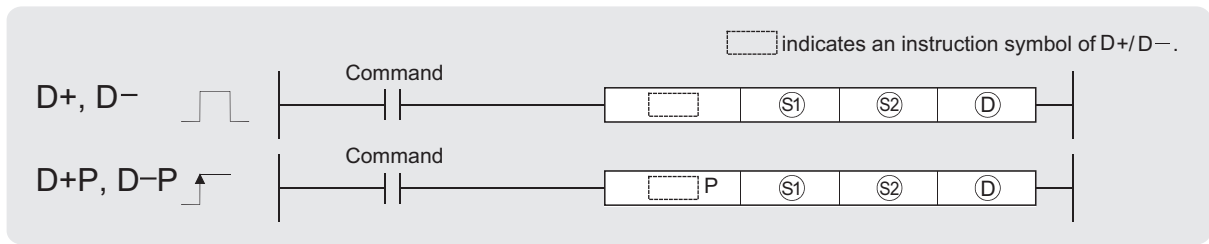
· K-2147483648 - K2 → K2147483646 ..... Since bit 31 value is "0",  
(80000000H) (00000002H) (7FFFFFFEH) result of operation takes a positive value.

· K2147483647 - K-2 → K-2147483647 ..... Since bit 31 value is "1",  
(80000000H) (FFFFFFFEH) (80000001H) result of operation takes a negative value.

**Operation Error**

- (1) There are no operation errors associated with the D+(P) or D-(P) instruction.

2 When three data are set ((S1+1,S1)+(S2+1,S2)→(D+1,D), (S1+1,S1)-(S2+1,S2)→(D+1,D))



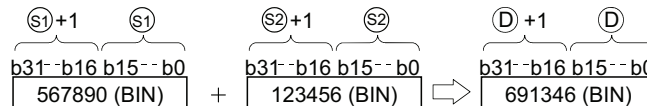
- Ⓢ<sub>1</sub> : Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BIN 32 bits)
- Ⓢ<sub>2</sub> : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 32 bits)
- ⓓ : Head number of the devices where the addition/subtraction operation result will be stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>					○			○	—
Ⓢ <sub>2</sub>					○			○	—
ⓓ					○			—	—

## ★ Function

### D+

- (1) Adds 32-bit BIN data designated by Ⓢ<sub>1</sub> to 32-bit BIN data designated by Ⓢ<sub>2</sub>, and stores the result of the addition at the device designated by ⓓ.

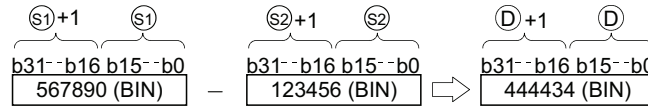


- (2) The values for Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub> and ⓓ can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- (3) Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
  - 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.

- K2147483647 (7FFFFFFFH) +K2 (00000002H) → K-2147483647 (80000001H) ... Since bit 31 value is "1", result of operation takes a negative value.
- K-2147483648 (80000000H) +K-2 (FFFFFFFEH) → K2147483646 (7FFFFFFEH) ... Since bit 31 value is "0", result of operation takes a positive value.

**D-**

- (1) Subtracts 32-bit BIN data designated by  $\text{S1}$  from 32-bit BIN data designated by  $\text{S2}$  and stores the result of the subtraction at the device designated by  $\text{D}$ .



- (2) The values for  $\text{S1}$ ,  $\text{S2}$  and  $\text{D}$  can be designated at between  $-2147483648$  and  $2147483647$  (BIN 32 bits).
- (3) Judgment of whether the data is positive or negative is made on the basis of the most significant bit (b31).
- 0: Positive
  - 1: Negative
- (4) The following will happen when an underflow or overflow is generated in an operation result: The carry flag in this case does not go ON.

$\begin{array}{l} \text{K} - 2147483648 - \text{K}2 \longrightarrow \text{K}2147483646 \cdots \cdots \text{ Since bit 31 value is "0",} \\ (80000000\text{H}) \quad (00000002\text{H}) \quad (7\text{FFFFFFE}\text{H}) \quad \text{result of operation takes a positive value.} \end{array}$

$\begin{array}{l} \text{K}2147483647 - \text{K} - 2 \longrightarrow \text{K} - 2147483647 \cdots \cdots \text{ Since bit 31 value is "1",} \\ (7\text{FFFFFFF}\text{H}) \quad (\text{FFFFFFFE}\text{H}) \quad (80000001\text{H}) \quad \text{result of operation takes a negative value.} \end{array}$

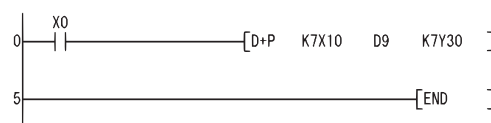
## ! Operation Error

- (1) There are no operation errors associated with the D+(P) or D-(P) instruction.

## Program Example

- (1) The following program adds 28-bit data from X10 to X2B to the data at D9 and D10 when X0 goes ON, and outputs the result of the operation to Y30 to Y4B.

[Ladder Mode]

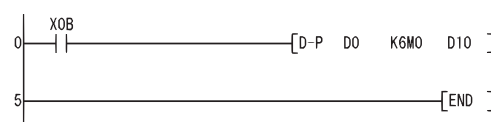


[List Mode]

Step	Instruction	Device
0	LD	X0
1	D+P	K7X10 D9 K7Y30
5	END	

- (2) The following program subtracts the data from M0 to M23 from the data at D0 and D1 when XB goes ON, and stores the result at D10 and D11.

[Ladder Mode]

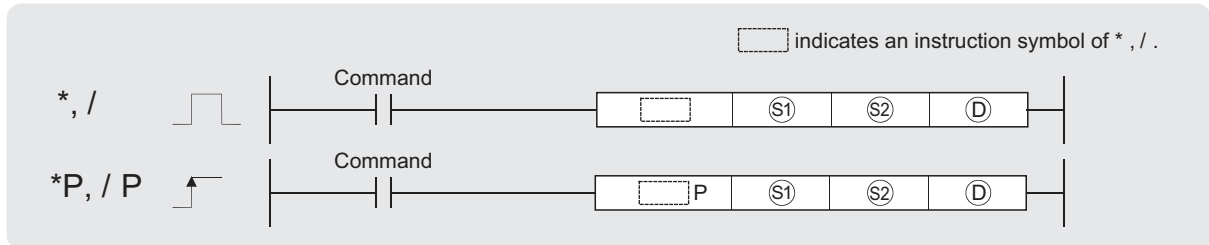


[List Mode]

Step	Instruction	Device
0	LD	X0B
1	D-P	D0 K6M0 D10
5	END	

## 6.2.3 BIN 16-bit multiplication and division operations (\*(P),/(P))

Basic High performance Process Redundant Universal



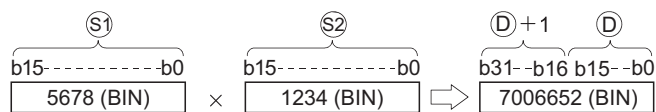
- Ⓢ<sub>1</sub> : Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BIN 16 bits)
- Ⓢ<sub>2</sub> : Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BIN 16 bits)
- Ⓣ : Head number of the devices where the multiplication/division operation result will be stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	Jn		Un	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>								○	—
Ⓢ <sub>2</sub>								○	—
Ⓣ								—	—

### ★ Function

\*

- (1) Multiplies BIN 16-bit data designated by Ⓢ<sub>1</sub> and BIN 16-bit data designated by Ⓢ<sub>2</sub>, and stores the result in the device designated by Ⓣ.



- (2) If Ⓣ is a bit device, designation is made from the lower bits.

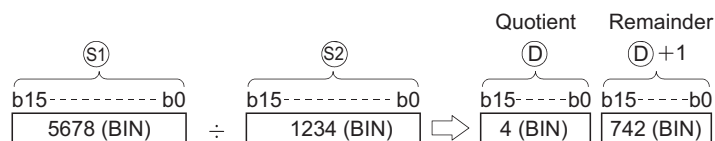
**Example**

- K1..... Lower 4 bits (b0 to b3)
- K4..... Lower 16 bits (b0 to b15)
- K8..... 32 bits (b0 to b31)

- (3) Values for Ⓢ<sub>1</sub> and Ⓢ<sub>2</sub> can be designated between -32768 and 32767 (BIN, 16 bits).
- (4) Judgments whether Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub>, and Ⓣ are positive or negative are made on the basis of the most significant bit (b15 for Ⓢ<sub>1</sub>, and Ⓢ<sub>2</sub>, for Ⓣ and b31).
  - 0: Positive
  - 1: Negative

/

- (1) Divides BIN 16-bit data designated by  $\textcircled{S1}$  and BIN 16-bit data designated by  $\textcircled{S2}$ , and stores the result in the device designated by  $\textcircled{D}$ .



- (2) If a word device has been used, the result of the division operation is stored as 32 bits, and both the quotient and remainder are stored; if a bit device has been used, 16 bits are used and only the quotient is stored.

Quotient: Stored at the lower 16 bits.

Remainder: Stored at the upper 16 bits (Stored only when using a word device).

- (3) Values for  $\textcircled{S1}$  and  $\textcircled{S2}$  can be designated between  $-32768$  and  $32767$  (BIN 16 bits).
- (4) Judgment whether values for  $\textcircled{S1}$ ,  $\textcircled{S2}$ ,  $\textcircled{D}$  and  $\textcircled{D+1}$  are positive or negative is made on the basis of the most significant bit (b15). (Sign is attached to both the quotient and remainder.)
- 0: Positive
  - 1: Negative

## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

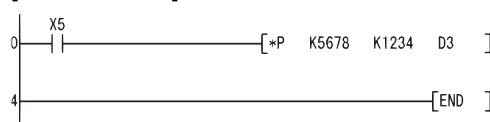
- Attempt to divide  $\textcircled{S2}$  by 0.

(Error code: 4100)

## Program Example

- (1) The following program multiplies "5678" by "1234" in BIN and stores the result at D3 and D4 when X5 turns ON.

[Ladder Mode]

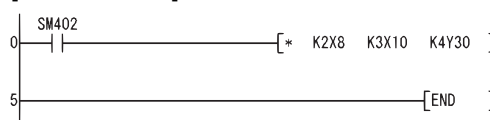


[List Mode]

Step	Instruction	Device
0	LD	X5
1	*P	K5678 K1234 D3
4	END	

- (2) The following program multiplies BIN data at X8 to XF by BIN data at X10 to X1B, and outputs the result of the multiplication to Y30 to Y3F.

[Ladder Mode]

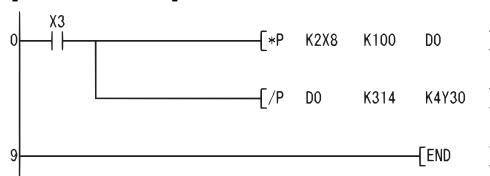


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	*	K2X8 K3X10 K4Y30
5	END	

- (3) The following program divides, when X3 is turned ON, the data at X8 to XF by 3.14 and outputs the operation result at Y30 to Y3F.

[Ladder Mode]

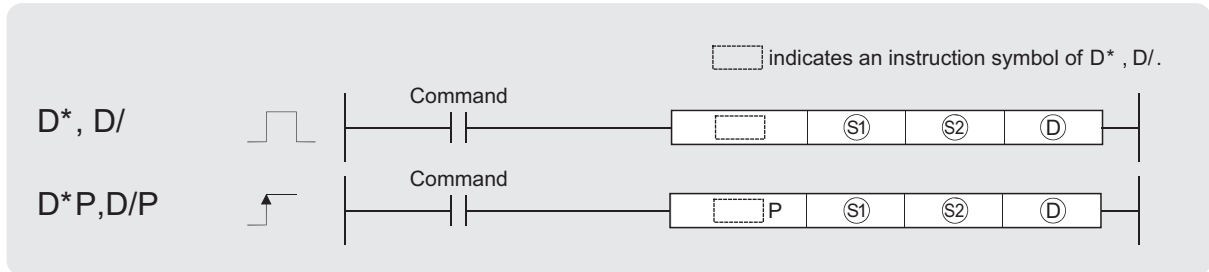


[List Mode]

Step	Instruction	Device
0	LD	X3
1	*P	K2X8 K100 D0
5	/P	D0 K314 K4Y30
9	END	

## 6.2.4 BIN 32-bit multiplication and division operations (D\*(P),D/(P))

Basic High performance Process Redundant Universal



Ⓢ<sub>1</sub> : Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BIN 32 bits)

Ⓢ<sub>2</sub> : Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BIN 32 bits)

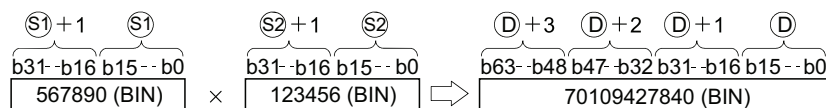
Ⓧ : Head number of the devices where the multiplication/division operation result will be stored (BIN 64 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>		○				○			—
Ⓢ <sub>2</sub>		○				○			—
Ⓧ		○				—			—

### ★ Function

#### D\*

- (1) Multiplies BIN 32-bit data designated by Ⓢ<sub>1</sub> and BIN 32-bit data designated by Ⓢ<sub>2</sub>, and stores the result in the device designated by Ⓧ.



- (2) If Ⓧ is a bit device, only the lower 32 bits of the multiplication result will be considered, and the upper 32 bits cannot be designated.

**Example** K1..... Lower 4 bits (b0 to b3)  
 K4..... Lower 16 bits (b0 to b15)  
 K8..... Lower 32 bits (b0 to b31)

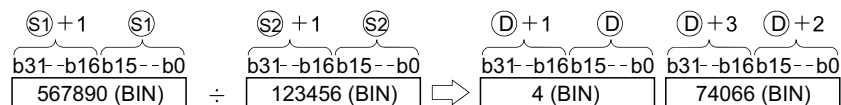
If the upper 32 bits of the bit device are required for the result of the multiplication operation, first temporarily store the data in a word device, then transfer the word device data to the bit device by designating (Ⓧ +2) and (Ⓧ +3) data.

- (3) The values for Ⓢ<sub>1</sub> and Ⓢ<sub>2</sub> can be designated at between -2147483648 and 2147483647 (BIN 32 bits).
- (4) Judgments whether Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub>, and Ⓧ are positive or negative are made on the basis of the most significant bit (b31 for Ⓢ<sub>1</sub> and Ⓢ<sub>2</sub>, b63 for Ⓧ).
  - 0: Positive
  - 1: Negative



**D/**

- (1) Divides BIN 32-bit data designated by  $\textcircled{S1}$  and BIN 32-bit data designated by  $\textcircled{S2}$ , and stores the result in the device designated by  $\textcircled{D}$ .



- (2) With a word device, the division operation result is stored in 64 bits and both the quotient and remainder are stored. With a bit device, only the quotient is stored as the operation result in 32 bits.

Quotient : Stored at the lower 32 bits.

Remainder : Stored at the upper 32 bits (Stored only when using a word device).

- (3) The values for  $\textcircled{S1}$  and  $\textcircled{S2}$  can be designated at between  $-2147483648$  and  $2147483647$  (BIN 32 bits).
- (4) Judgment whether values for  $\textcircled{S1}$ ,  $\textcircled{S2}$ ,  $\textcircled{D}$  and  $\textcircled{D}+2$  are positive or negative is made on the basis of the most significant bit (b31).  
(Sign is attached to both the quotient and remainder.)
- 0: Positive
  - 1: Negative

## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) is turned ON, and the corresponding error code is stored into SD0.

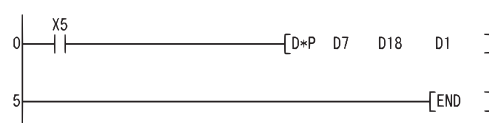
- Attempt to divide  $\textcircled{S2}$  by 0.

(Error code: 4100)

## Program Example

- (1) The following program multiplies the BIN data at D7 and D8 by the BIN data at D18 and D19 when X5 is ON, and stores the result at D1 to D4.

[Ladder Mode]

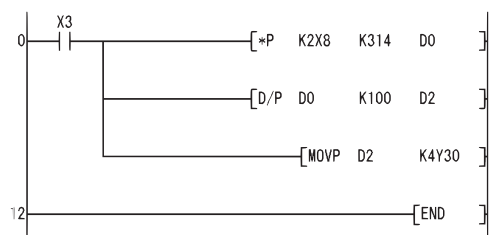


[List Mode]

Step	Instruction	Device
0	LD	X5
1	D*P	D7 D18 D1
5	END	

- (2) The following program outputs the value resulting when the data at X8 to XF is multiplied by 3.14 to Y3F when X3 is ON.

[Ladder Mode]



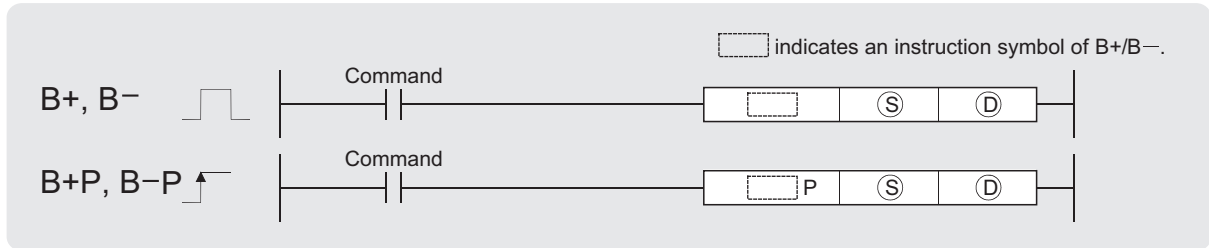
[List Mode]

Step	Instruction	Device
0	LD	X3
1	*P	K2X8 K314 D0
5	D/P	D0 K100 D2
10	MOV	D2 K4Y30
12	END	

## 6.2.5 BCD 4-digit addition and subtraction operations (B+(P),B-(P))

Basic High performance Process Redundant Universal

1 When two data are set ( $\textcircled{D} + \textcircled{S} \rightarrow \textcircled{D}$ ,  $\textcircled{D} - \textcircled{S} \rightarrow \textcircled{D}$ )



$\textcircled{S}$  : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 4 digits)

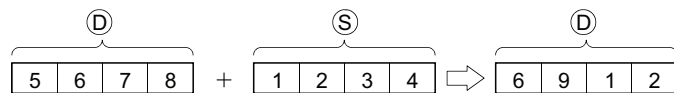
$\textcircled{D}$  : Head number of the devices where the data to be added to/subtracted from is stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	JGO		UGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
$\textcircled{S}$								○	—
$\textcircled{D}$								—	—

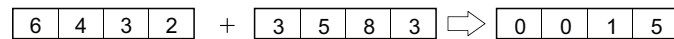
### ★ Function

#### B+

- (1) Adds the BCD 4-digit data designated by  $\textcircled{D}$  and the BCD 4-digit data designated by  $\textcircled{S}$ , and stores the result of the addition at the device designated by  $\textcircled{D}$ .

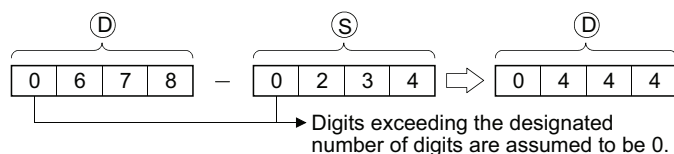


- (2) 0 to 9999 (BCD 4 digits) can be assigned to  $\textcircled{S}$  and  $\textcircled{D}$ .  
 (3) If the result of the addition operation exceeds 9999, the higher bits are ignored. The carry flag in this case does not go ON.



#### B-

- (1) Subtracts the BCD 4-digit data designated by  $\textcircled{S}$  and the BCD 4-digit data designated by  $\textcircled{D}$ , and stores the result of the subtraction at the device designated by  $\textcircled{D}$ .



- (2) 0 to 9999 (BCD 4 digits) can be assigned to  $\textcircled{S}$  and  $\textcircled{D}$ .

- (3) The following will result if an underflow is generated by the subtraction operation:  
The carry flag in this case does not go ON.

$$\begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 0 & 0 & 1 \\ \hline \end{array} - \begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 0 & 0 & 3 \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|c|c|} \hline 9 & 9 & 9 & 9 & 8 \\ \hline \end{array}$$

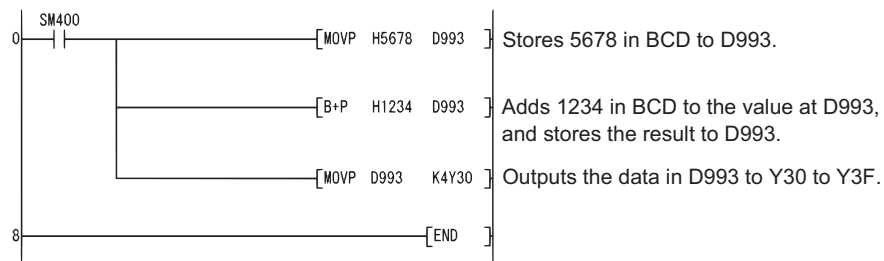
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $\textcircled{S}$  or  $\textcircled{D}$  BCD data is outside the 0 to 9999 range. (Error code: 4100)

## Program Example

- (1) The following program adds BCD data 5678 and 1234, stores it at D993, and at the same time outputs it to from Y30 to Y3F.

[Ladder Mode]

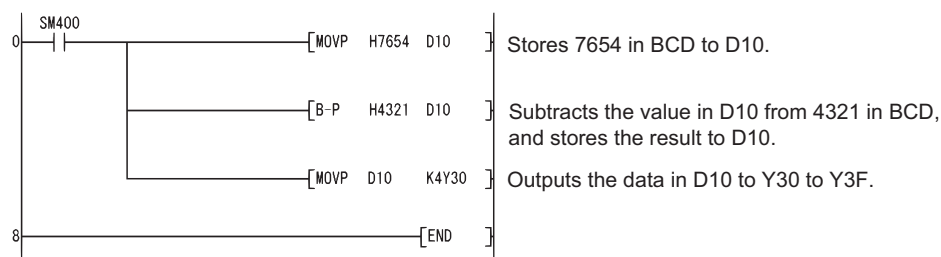


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOV P	H5678 D993
3	B+P	H1234 D993
6	MOV P	D993 K4Y30
8	END	

- (2) The following program subtracts the BCD data 4321 from 7654, stores the result at D10, and at the same time outputs it to Y30 to Y3F.

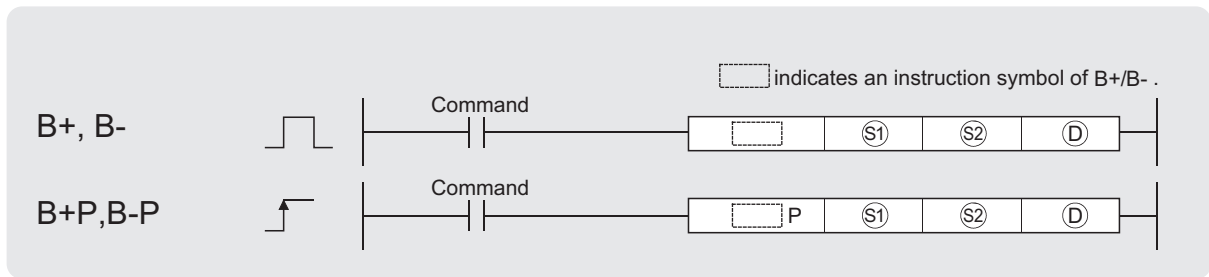
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOV P	H7654 D10
3	B-P	H4321 D10
6	MOV P	D10 K4Y30
8	END	

2 When three data are set (S1+S2 → D, S1-S2 → D)



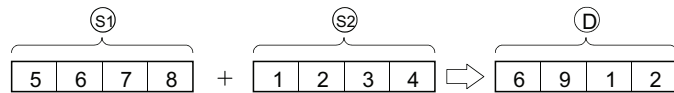
- Ⓢ<sub>1</sub> : Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BCD 4 digits)
- Ⓢ<sub>2</sub> : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 4 digits)
- ⓓ : Head number of the devices where the addition/subtraction operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>					○			○	—
Ⓢ <sub>2</sub>					○			○	—
ⓓ					○			—	—

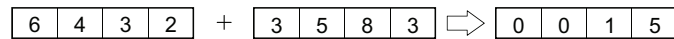
★ Function

**B+**

- (1) Adds the BCD 4-digit data designated by Ⓢ<sub>1</sub> and the BCD 4-digit data designated by Ⓢ<sub>2</sub>, and stores the result of the addition at the device designated by ⓓ.

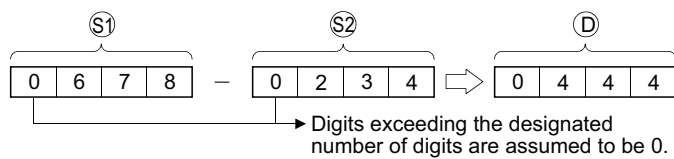


- (2) 0 to 9999 (BCD 4 digits) can be assigned to Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub> and ⓓ.  
 (3) If the result of the addition operation exceeds 9999, the higher bits are ignored. The carry flag in this case does not ON.



**B-**

- (1) Subtracts the BCD 4-digit data designated by Ⓢ<sub>1</sub> and the BCD 4-digit data designated by Ⓢ<sub>2</sub>, and stores the result of the subtraction at the device designated by ⓓ.



- (2) 0 to 9999 (BCD 4 digits) can be assigned to Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub> and ⓓ.

- (3) The following will result if an underflow is generated by the subtraction operation:  
The carry flag in this case does not go ON.

$$\begin{array}{|c|c|c|c|} \hline 0 & 0 & 0 & 1 \\ \hline \end{array} - \begin{array}{|c|c|c|c|} \hline 0 & 0 & 0 & 3 \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|c|} \hline 9 & 9 & 9 & 8 \\ \hline \end{array}$$

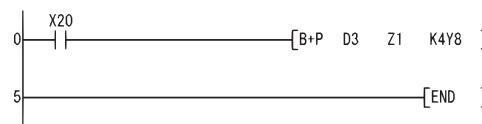
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $\text{S1}$ ,  $\text{S2}$   $\text{D}$  or BCD data is outside the 0 to 9999 range. (Error code: 4100)

## Program Example

- (1) The following program adds the D3 BCD data and the Z1 BCD data when X20 goes ON, and outputs the result to Y8 to Y17.

[Ladder Mode]

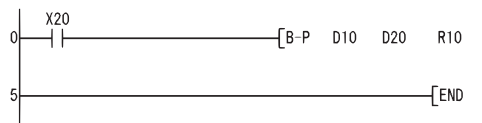


[List Mode]

Step	Instruction	Device
0	LD	X20
1	B+P	D3 Z1 K4Y8
5	END	

- (2) The following program subtracts the BCD data at D20 from the BCD data at D10 when X20 goes ON, and stores the result at R10.

[Ladder Mode]



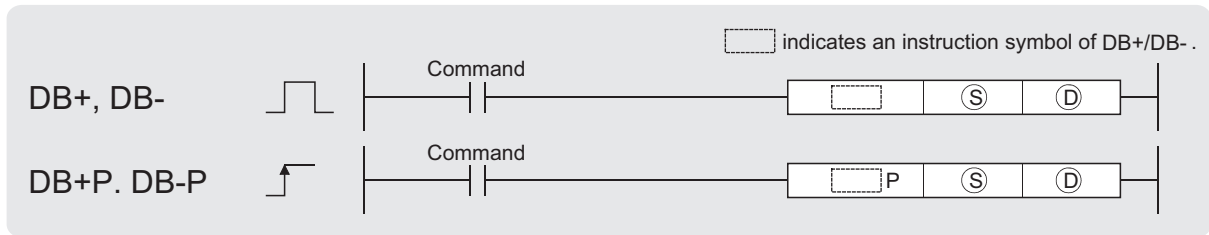
[List Mode]

Step	Instruction	Device
0	LD	X20
1	B-P	D10 D20 R10
5	END	

## 6.2.6 BCD 8-digit addition and subtraction operations (DB+(P),DB-(P))

Basic High performance Process Redundant Universal

1 When two data are set  $((D+1,D)+(S+1,S) \rightarrow (D+1,D), (D+1,D)-(S+1,S) \rightarrow (D+1,D))$



Ⓢ : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 8 digits)

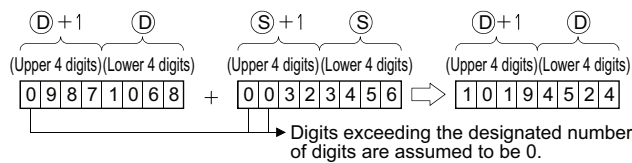
Ⓣ : Head number of the devices where the data to be added to/subtracted from is stored (BCD 8 digits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

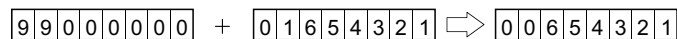
### ★ Function

#### DB+

- (1) Adds the BCD 8-digit data designated by Ⓣ and the BCD 8-digit data designated by Ⓢ, and stores the result of the addition at the device designated by Ⓣ.

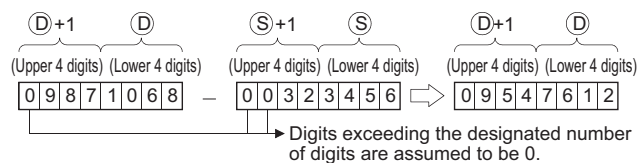


- (2) 0 to 99999999 (BCD 8 digits) can be assigned to Ⓢ and Ⓣ.
- (3) If the result of the addition operation exceeds 99999999, the upper bits will be ignored. The carry flag in this case does not go ON.



#### DB-

- (1) Subtracts the BCD 8-digit data designated by Ⓣ and the BCD 8-digit data designated by Ⓢ, and stores the result of the subtraction at the device designated by Ⓣ.



- (2) 0 to 99999999 (BCD 8 digits) can be assigned to  $\textcircled{S}$  and  $\textcircled{D}$ .
- (3) The following will result if an underflow is generated by the subtraction operation: The carry flag in this case does not go ON.

$$\begin{array}{|c|c|c|c|c|c|c|c|} \hline 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ \hline \end{array} - \begin{array}{|c|c|c|c|c|c|c|c|} \hline 1 & 2 & 3 & 4 & 5 & 6 & 7 & 9 \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|c|c|c|c|c|} \hline 9 & 9 & 9 & 9 & 9 & 9 & 9 & 9 \\ \hline \end{array}$$

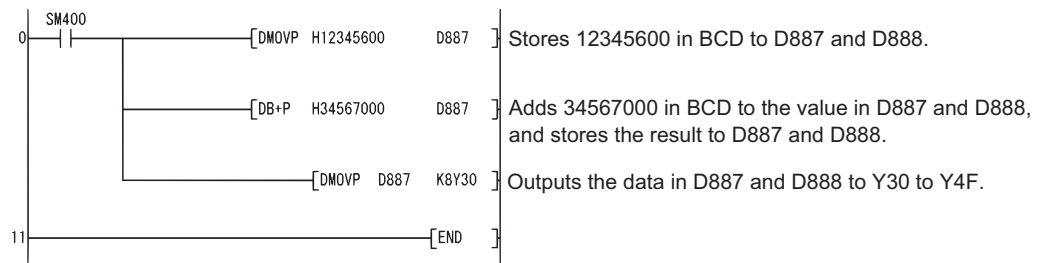
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $\textcircled{S}$  or  $\textcircled{D}$  BCD data is outside the 0 to 99999999 range. (Error code: 4100)

## Program Example

- (1) The following program adds the BCD data 12345600 and 34567000, stores the result at D887 and D888, and at the same time outputs them to from Y30 to Y4F.

[Ladder Mode]

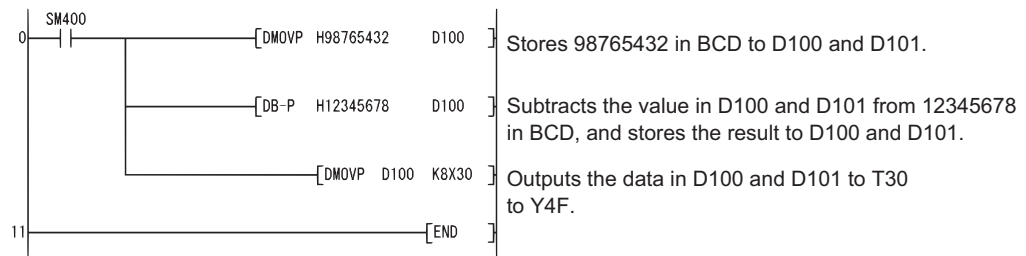


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DMOVP	H12345600 D887
4	DB+P	H34567000 D887
8	DMOVP	D887 K8Y30
11	END	

- (2) The following program subtracts the BCD data 98765432 from 12345678, stores the result at D100 and D101, and at the same time outputs it from Y30 to Y4F.

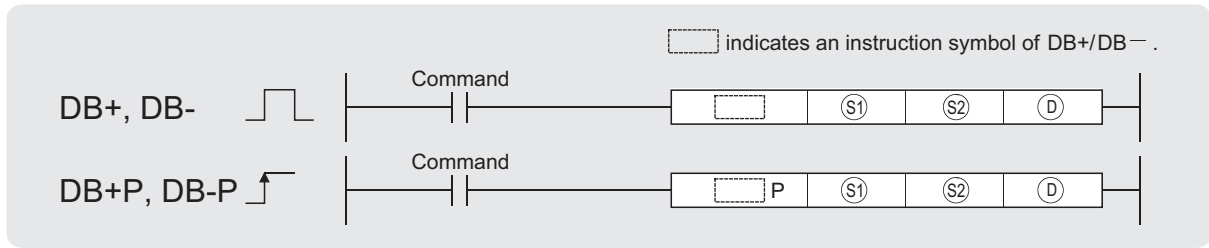
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DMOVP	H98765432 D100
4	DB-P	H12345678 D100
8	DMOVP	D100 K8Y30
11	END	

2 When three data are set ((S1+1,S1)+(S2+1,S2)→(D+1,D), (S1+1,S1)-(S2+1,S2)→(D+1,D))



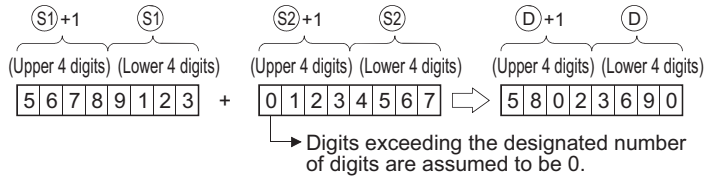
- Ⓢ1 : Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (BCD 8 digits)
- Ⓢ2 : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BCD 8 digits)
- Ⓧ : Head number of the devices where the addition/subtraction operation result is stored (BCD 8 digits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1					○			○	—
Ⓢ2					○			○	—
Ⓧ					○			—	—

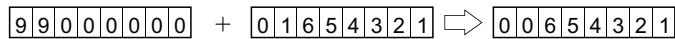
## ★ Function

### DB+

- (1) Adds the BCD 8-digit data designated by Ⓢ1 and the BCD 8-digit data designated by Ⓢ2, and stores the result of the addition at the device designated by Ⓧ.



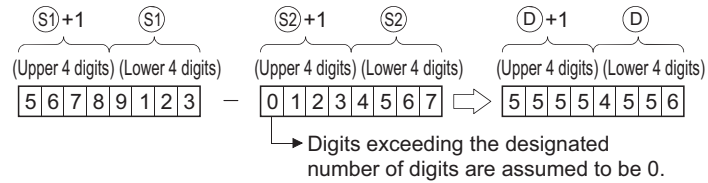
- (2) 0 to 99999999 (BCD 8 digits) can be assigned to Ⓢ1, Ⓢ2 and Ⓧ.
- (3) If the result of the addition operation exceeds 99999999, the upper bits will be ignored. The carry flag in this case does not go ON.



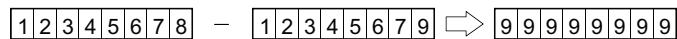


**DB-**

- (1) Subtracts the BCD 8-digit data designated by  $\textcircled{S1}$  and the BCD 8-digit data designated by  $\textcircled{S2}$ , and stores the result of the subtraction at the device designated by  $\textcircled{D}$ .



- (2) 0 to 99999999 (BCD 8 digits) can be assigned to  $\textcircled{S1}$ ,  $\textcircled{S2}$  and  $\textcircled{D}$ .
- (3) The following will result if an underflow is generated by the subtraction operation:  
The carry flag in this case does not go ON.



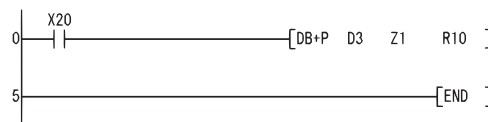
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $\textcircled{S1}$ ,  $\textcircled{S2}$   $\textcircled{D}$  or BCD data is outside the 0 to 99999999 range. (Error code: 4100)

## Program Example

- (1) The following program adds the BCD data at D3 and D4 to the BCD data at Z1 and Z2 when X20 goes ON, and stores the result at R10 and R11.

[Ladder Mode]

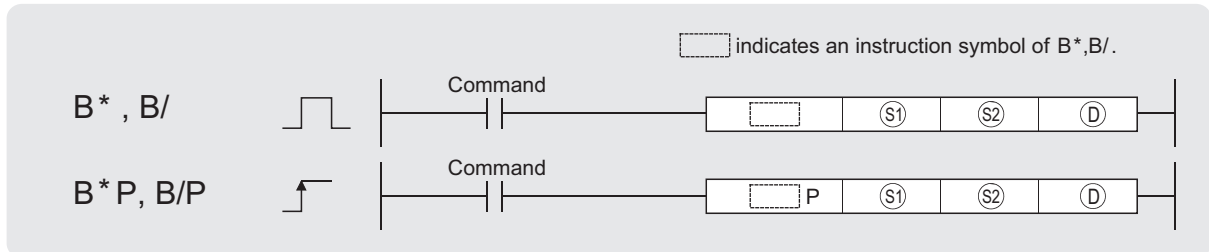


[List Mode]

Step	Instruction	Device
0	LD	X20
1	DB+P	D3 Z1 R10
5	END	

## 6.2.7 BCD 4-digit multiplication and division operations (B\*(P),B/(P))

Basic High performance Process Redundant Universal



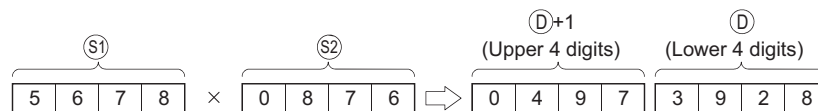
- Ⓢ<sub>1</sub> : Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BCD 4 digits)
- Ⓢ<sub>2</sub> : Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BCD 4 digits)
- Ⓣ : Head number of the devices where the multiplication/division operation result will be stored (BCD 8 digits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>					○			○	—
Ⓢ <sub>2</sub>					○			○	—
Ⓣ					○			—	—

### ★ Function

#### B\*

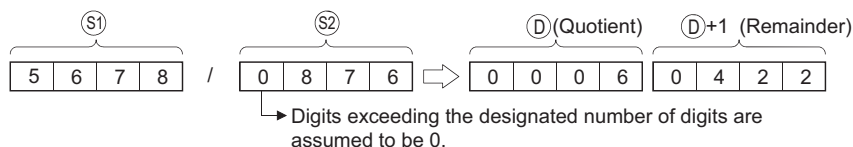
- (1) Multiplies BCD data designated by Ⓢ<sub>1</sub> and BCD data designated by Ⓢ<sub>2</sub>, and stores the result in the device designated by Ⓣ.



- (2) 0 to 9999 (BCD 4 digits) can be assigned to Ⓢ<sub>1</sub> and Ⓢ<sub>2</sub>.

#### B/

- (1) Divides BCD data designated by Ⓢ<sub>1</sub> and BCD data designated by Ⓢ<sub>2</sub>, and stores the result in the device designated by Ⓣ.



- (2) Uses 32 bits to store the result of the division as quotient and remainder  
 Quotient (BCD 4 digits) :Stored at the lower 16 bits.  
 Remainder (BCD 4 digits) :Stored at the upper 16 bits.
- (3) If Ⓣ has been designated as a bit device, the remainder of the operation will not be stored.

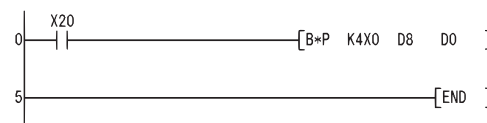
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The ⑤1 or ⑤2 BCD data is outside the 0 to 9999 range. (Error code: 4100)
  - Attempt to divide ⑤2 by 0. (Error code: 4100)

## Program Example

- (1) The following program multiplies, when X20 is turned ON, the BCD data at X0 to XF by the BCD data at D8 and stores the operation result at D0 to D1.

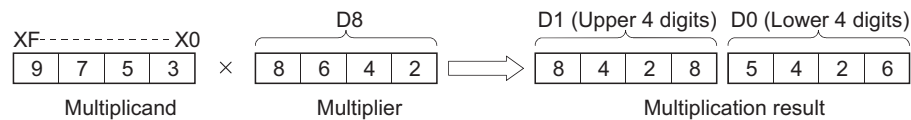
[Ladder Mode]



[List Mode]

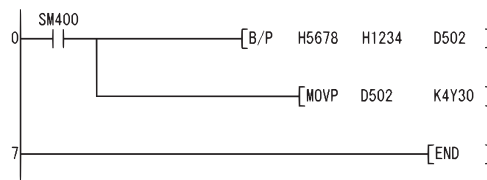
Step	Instruction	Device
0	LD	X20
1	B*P	K4X0 D8 D0
5	END	

[Operation]



- (2) The following program divides 5678 by the BCD data 1234, stores the result at D502 and D503, and at the same time outputs the quotient to Y30 to Y3F.

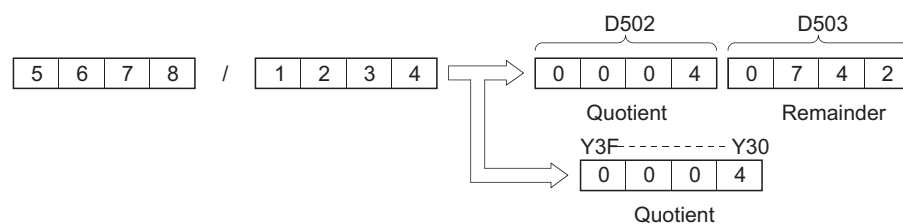
[Ladder Mode]



[List Mode]

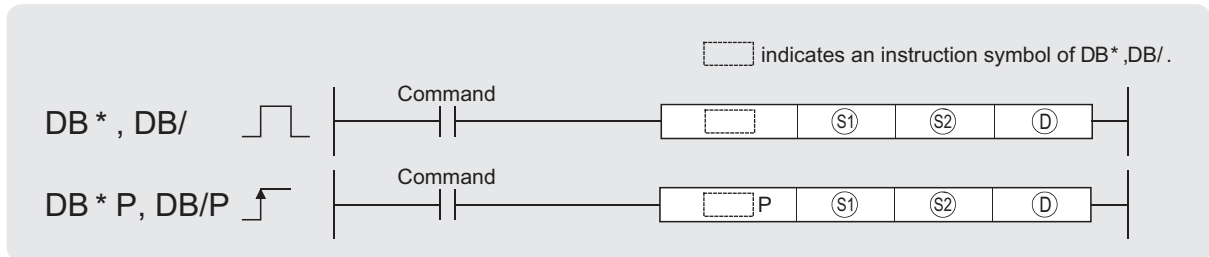
Step	Instruction	Device
0	LD	SM400
1	B/P	H5678 H1234 D502
5	MOV P	D502 K4Y30
7	END	

[Operation]



## 6.2.8 BCD 8-digit multiplication and division operations (DB\*(P),DB/(P))

Basic High performance Process Redundant Universal



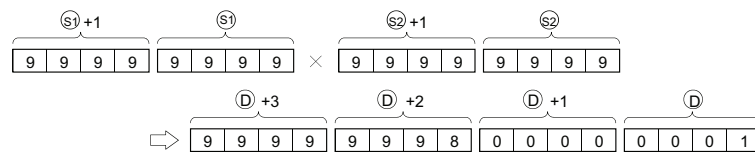
- Ⓢ1 : Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (BCD 8 digits)
- Ⓢ2 : Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (BCD 8 digits)
- Ⓣ : Head number of the devices where the multiplication/division operation result will be stored (BCD 16 digits)

Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1		○				○			—
Ⓢ2		○				○			—
Ⓣ		○				—			—

### ★ Function

#### DB\*

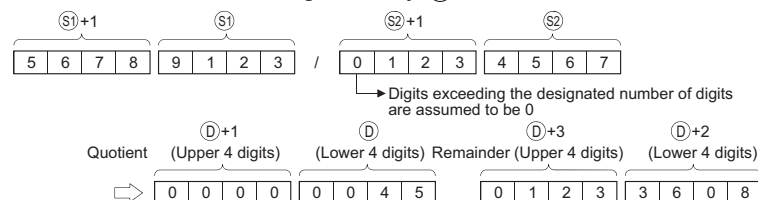
- (1) Multiplies the BCD 8-digit data designated by Ⓢ1 and the BCD 8-digit data designated by Ⓢ2, and stores the product at the device designated by Ⓣ.



- (2) If Ⓣ has designated a bit device, the lower 8 digits (lower 32 bits) will be used for the product, and the higher 8 digits (upper 32 bits) cannot be designated.  
K1 ....Lower 1 digit (b0 to 3), K4 ....Lower 4 digits (b0 to 15), K8.....Lower 8 digits (b0 to 31)
- (3) 0 to 99999999 (BCD 8 digits) can be assigned to Ⓢ1 and Ⓢ2.

#### DB/

- (1) Divides 8-digit BCD data designated by Ⓢ1 and 8-digit BCD data designated by Ⓢ2, and stores the result in the device designated by Ⓣ.



- (2) 64 bits are used for the result of the division operation, and stored as quotient and remainder.

Quotient (BCD 8 digits) : Stored at the lower 32 bits.  
 Remainder (BCD 8 digits) : Stored at the upper 32 bits.

- (3) If  $\textcircled{D}$  has been designated as a bit device, the remainder of the operation will not be stored.

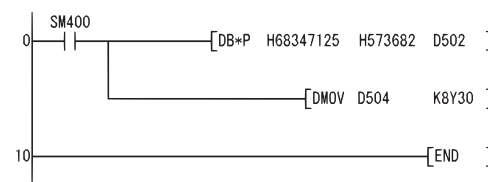
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $\textcircled{S1}$  or  $\textcircled{S2}$  BCD data is outside the 0 to 99999999 range. (Error code: 4100)
  - Attempt to divide  $\textcircled{S2}$  by 0. (Error code: 4100)

## Program Example

- (1) The following program multiplies the BCD data 67347125 and 573682, stores the result from D502 to D505, and at the same time outputs the upper 8 digits to Y30 to Y4F.

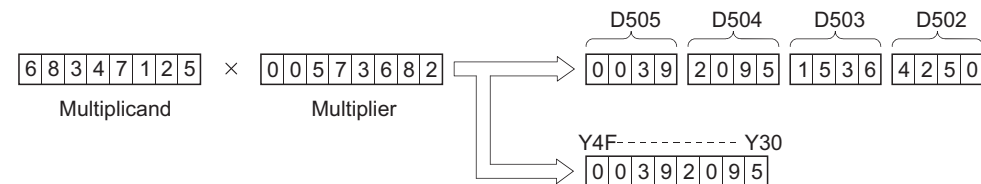
[Ladder Mode]



[List Mode]

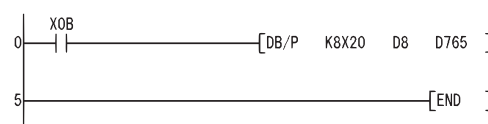
Step	Instruction	Device
0	LD	SM400
1	DB*P	H68347125 H573682 D502
7	DMOV	D504 K8Y30
10	END	

[Operation]



- (2) The following program divides the BCD data from X20 to X3F by the BCD data at D8 and D9 when X0B goes ON, and stores the result from D765 to D768.

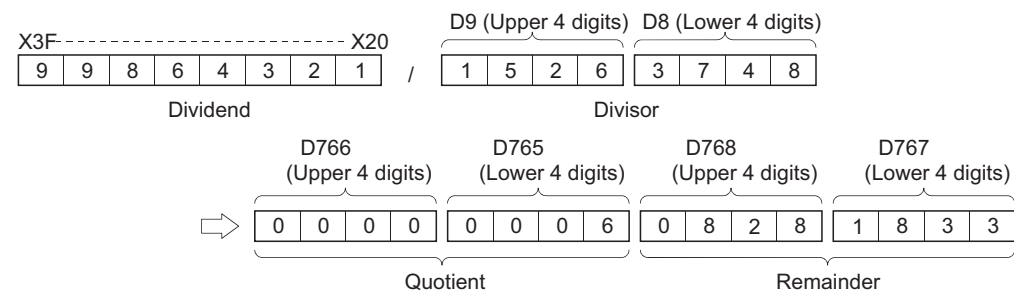
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0B
1	DB/P	K8X20 D8 D765
5	END	

[Operation]

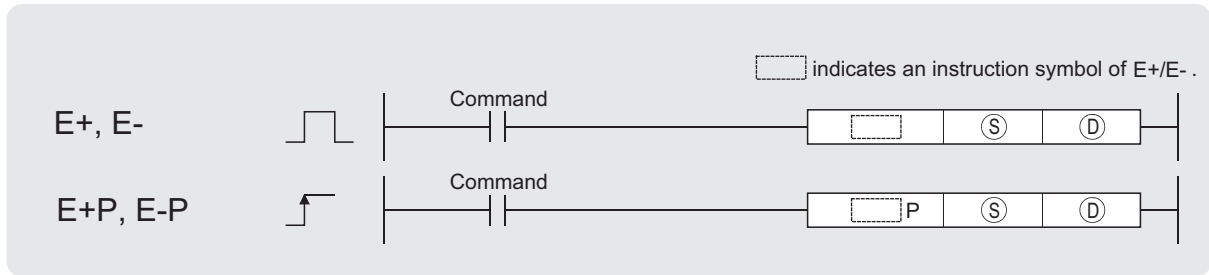


## 6.2.9 Addition and subtraction of floating decimal point data (Single precision) (E+(P),E-(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.

① When two data are set  $((\textcircled{D}+1, \textcircled{D})+(\textcircled{S}+1, \textcircled{S}) \rightarrow (\textcircled{D}+1, \textcircled{D}), (\textcircled{D}+1, \textcircled{D})-(\textcircled{S}+1, \textcircled{S}) \rightarrow (\textcircled{D}+1, \textcircled{D}))$



Ⓢ : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)

ⓓ : Head number of the devices where the data to be added to/subtracted from is stored (real number)

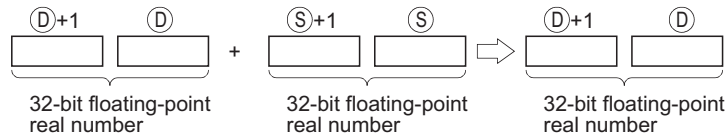
Setting Data	Internal Devices		R, ZR	JOG		UJGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	—	—	○	—	
ⓓ	—	○	—	○	—	—	—	—	

\*1: Available only in multiple Universal model QCPU

### ★ Function

#### E+

- (1) Adds the 32-bit floating decimal point type real number designated at ⓓ and the 32-bit floating decimal point type real number designated at Ⓢ, and stores the sum in the device designated at ⓓ.

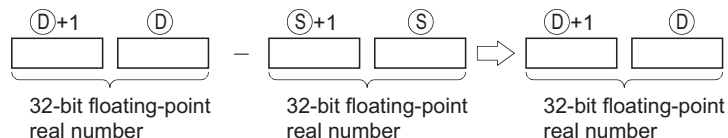


- (2) Values which can be designated at Ⓢ and ⓓ and which can be stored, are as follows:

$$0, 2^{-126} \leq | \text{Designated value (stored value)} | < 2^{128}$$

#### E-

- (1) Subtracts a 32-bit floating decimal point type real number designated by ⓓ and a 32-bit floating decimal point type real number designated by Ⓢ, and stores the result at a device designated by ⓓ.



- (2) Values which can be designated at ③ and ④ and which can be stored, are as follows:

$$0, 2^{-126} \leq | \text{Designated value (stored value)} | < 2^{128}$$

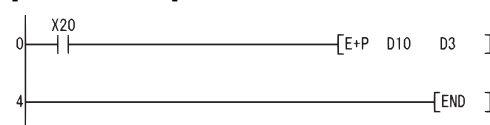
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range: (Error code: 4100)  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$   
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
  - The value of the specified device is  $-0$ .<sup>\*2</sup>  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
- <sup>\*2</sup>: There are CPU modules that will not result in an operation error if  $-0$  is specified.  
 Refer to Section 3.2.4 for details.
- The result of addition and subtraction exceeds the following range. (The overflow occurs.) (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Result of addition and subtraction} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ . (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program adds the 32-bit floating decimal point type real numbers at D3 and D4 and the 32-bit floating decimal point type real numbers at D10 and D11 when X20 goes ON, and stores the result at D3 and D4.

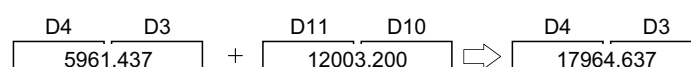
[Ladder Mode]



[List Mode]

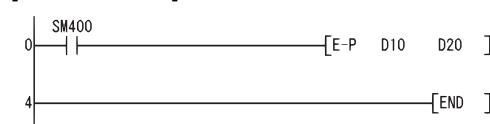
Step	Instruction	Device
0	LD	X20
1	E+P	D10 D3
4	END	

[Operation]



- (2) The following program subtracts the 32-bit floating decimal point type real number at D10 and D11 from the 32-bit floating decimal point type real numbers at D20 and D21, and stores the result of the subtraction at D20 and D21.

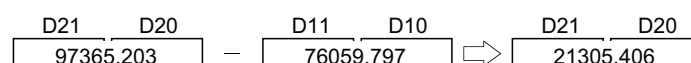
[Ladder Mode]



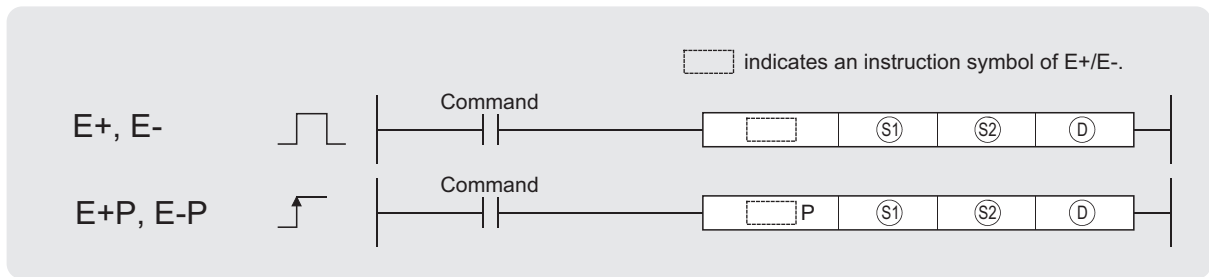
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	E-P	D10 D20
4	END	

[Operation]



2 When three data are set ((S1+1,S1)+(S2+1,S2)→(D+1,D), (S1+1,S1)-(S2+1,S2)→(D+1,D))



- Ⓢ<sub>1</sub> : Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (real number)
- Ⓢ<sub>2</sub> : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)
- ⓓ : Head number of the devices where the addition/subtraction operation result is stored (real number)

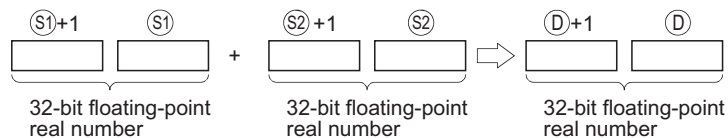
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○	—	—	○	—	○	—	
Ⓢ <sub>2</sub>	—	○	—	—	○	—	○	—	
ⓓ	—	○	—	—	○	—	—	—	

\*1: Available only in multiple Universal model QCPU

## ★ Function

### E+

- (1) Adds the 32-bit floating decimal point type real number designated at Ⓢ<sub>1</sub> and the 32-bit floating decimal point type real number designated at Ⓢ<sub>2</sub>, and stores the sum in the device designated at ⓓ.

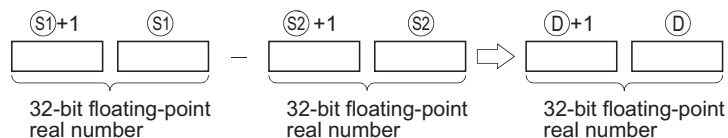


- (2) Values which can be designated at Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub> and ⓓ and which can be stored, are as follows:

$$0, 2^{-126} \leq | \text{Designated value (stored value)} | < 2^{128}$$

### E-

- (1) Subtracts a 32-bit floating decimal point type real number designated by Ⓢ<sub>1</sub> and a 32-bit floating decimal point type real number designated by Ⓢ<sub>2</sub>, and stores the result at a device designated by ⓓ.



- (2) Values which can be designated at Ⓢ<sub>1</sub> and Ⓢ<sub>2</sub> and ⓓ which can be stored, are as follows:

$$0, 2^{-126} \leq | \text{Designated value (stored value)} | < 2^{128}$$



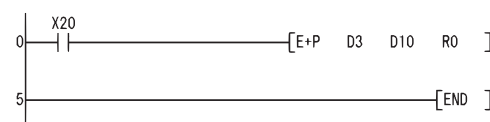
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range:  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$   
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
  - The value of the specified device is  $-0.^{*2}$   
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
- \*2: There are CPU modules that will not result in an operation error if  $-0$  is specified.  
Refer to Section 3.2.4 for details.
- The result of addition and subtraction exceeds the following range. (The overflow occurs.)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Result of addition and subtraction} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program adds the 32-bit floating decimal point type real numbers at D3 and D4 and the 32-bit floating decimal point type real numbers at D10 and D11 when X20 goes ON, and outputs the result to R0 and R1.

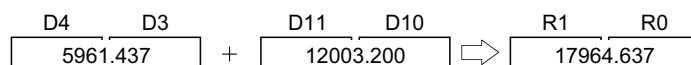
[Ladder Mode]



[List Mode]

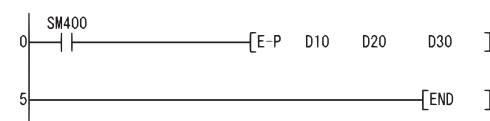
Step	Instruction	Device
0	LD	X20
1	E+P	D3 D10 R0
5	END	

[Operation]



- (2) The following programs subtracts the 32-bit floating decimal point type real numbers at D20 and D21 from the 32-bit floating decimal point type real numbers at D10 and D11, and stores the result at D30 and D31.

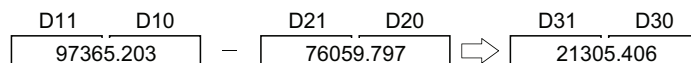
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	E-P	D10 D20 D30
5	END	

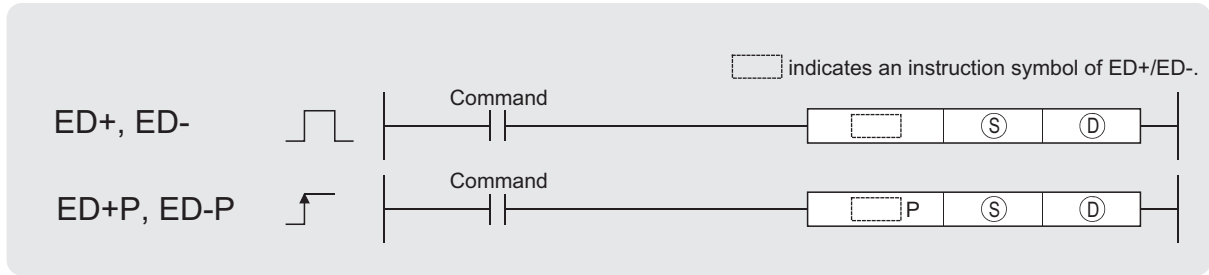
[Operation]



## 6.2.10 Addition and subtraction of floating decimal point data (Double precision) (ED+(P),ED-(P))



1 When two data are set ((D+3,D+2,D+1,D)+(S+3,S+2,S+1,S)→(D+3,D+2,D+1,D),  
(D+3,D+2,D+1,D)-(S+3,S+2,S+1,S)→(D+3,D+2,D+1,D))



Ⓢ : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)

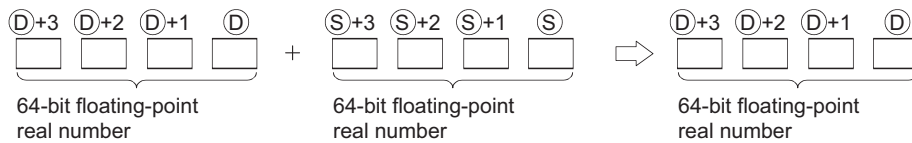
Ⓣ : Head number of the devices where the data to be added to/subtracted from is stored (real number)

Setting Data	Internal Devices		R, ZR	Jm		UAG	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

#### ED+

(1) Adds the 64-bit floating decimal point type real number designated at Ⓣ and the 64-bit floating decimal point type real number designated at Ⓢ, and stores the sum in the device designated at Ⓣ.

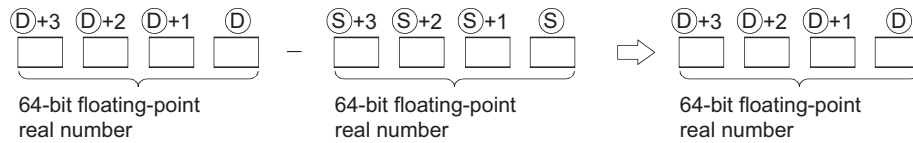


(2) Values which can be designated at Ⓢ and Ⓣ and which can be stored, are as follows:

$$0, 2^{-1022} \leq | \text{Designated value (stored value)} | < 2^{1024}$$

**ED-**

- (1) Subtracts a 64-bit floating decimal point type real number designated by  $\textcircled{D}$  and a 64-bit floating decimal point type real number designated by  $\textcircled{S}$ , and stores the result at a device designated by  $\textcircled{D}$ .



- (2) Values which can be designated at  $\textcircled{S}$  and  $\textcircled{D}$  and which can be stored, are as follows:

$$0, 2^{-1022} \leq | \text{Designated value (stored value)} | < 2^{1024}$$

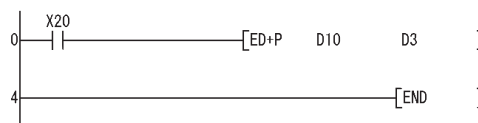
**Operation Error**

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range: (Error code: 4140)  
 $0, 2^{-1022} \leq | \text{Contents of designated device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The result of addition/subtraction exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Result of operation} |$  (Error code: 4141)

**Program Example**

- (1) The following program adds the 64-bit floating decimal point type real numbers at D3 to D6 and the 64-bit floating decimal point type real numbers at D10 to D13 when X20 goes ON, and stores the result at D3 to D6.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	ED+P	D10 D3
4	END	

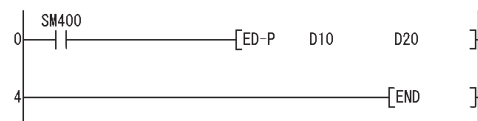
[Operation]

$$\begin{matrix} \text{D6} & \text{D5} & \text{D4} & \text{D3} \\ \square & \square & \square & \square \end{matrix} + \begin{matrix} \text{D13} & \text{D12} & \text{D11} & \text{D10} \\ \square & \square & \square & \square \end{matrix} \Rightarrow \begin{matrix} \text{D6} & \text{D5} & \text{D4} & \text{D3} \\ \square & \square & \square & \square \end{matrix}$$

5961.437      12003.200      17964.637

- (2) The following program subtracts the 64-bit floating decimal point type real number at D10 to D13 from the 64-bit floating decimal point type real numbers at D20 to D23, and stores the result of the subtraction at D20 to D23.

[Ladder Mode]



[List Mode]

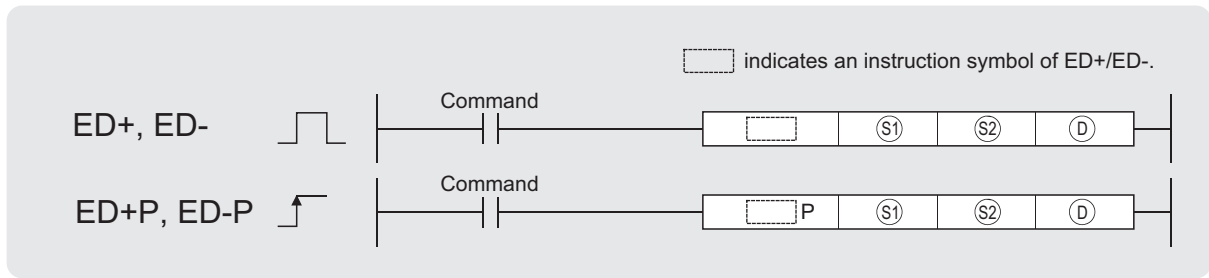
Step	Instruction	Device
0	LD	SM400
1	ED-P	D10 D20
4	END	

[Operation]

$$\begin{matrix} \text{D23} & \text{D22} & \text{D21} & \text{D20} \\ \square & \square & \square & \square \end{matrix} - \begin{matrix} \text{D13} & \text{D12} & \text{D11} & \text{D10} \\ \square & \square & \square & \square \end{matrix} \Rightarrow \begin{matrix} \text{D23} & \text{D22} & \text{D21} & \text{D20} \\ \square & \square & \square & \square \end{matrix}$$

97365.203      76059.797      21305.406

2 When three data are set  $((S1+3, S1+2, S1+1, S1) + (S2+3, S2+2, S2+1, S2) \rightarrow (D+3, D+2, D+1, D))$ ,  
 $((S1+3, S1+2, S1+1, S1) - (S2+3, S2+2, S2+1, S2) \rightarrow (D+3, D+2, D+1, D))$



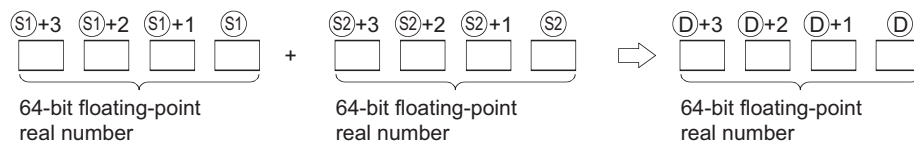
- Ⓢ1 : Data to be added to/subtracted from or head number of the devices where the data to be added to/subtracted from is stored (real number)
- Ⓢ2 : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (real number)
- Ⓧ : Head number of the devices where the addition/subtraction operation result is stored (real number)

Setting Data	Internal Devices		R, ZR	Jm		UAG	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		○	—
Ⓢ2	—	○				—		○	—
Ⓧ	—	○				—		—	—

## ★ Function

### ED+

- (1) Adds the 64-bit floating decimal point type real number designated at Ⓢ1 and the 64-bit floating decimal point type real number designated at Ⓢ2, and stores the sum in the device designated at Ⓧ.

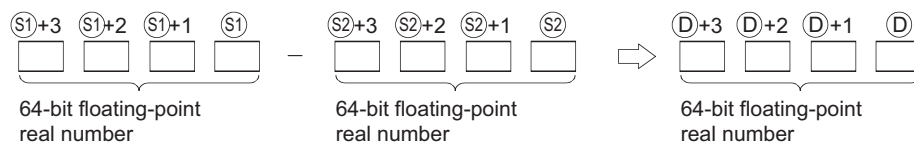


- (2) Values which can be designated at Ⓢ1, Ⓢ2 and Ⓧ and which can be stored, are as follows:

$$0, 2^{-1022} \leq | \text{Designated value (stored value)} | < 2^{1024}$$

### ED-

- (1) Subtracts a 64-bit floating decimal point type real number designated by Ⓢ1 and a 64-bit floating decimal point type real number designated by Ⓢ2, and stores the result at a device designated by Ⓧ.



- (2) Values which can be designated at Ⓢ1 and Ⓢ2 and Ⓧ which can be stored, are as follows:

$$0, 2^{-1022} \leq | \text{Designated value (stored value)} | < 2^{1024}$$

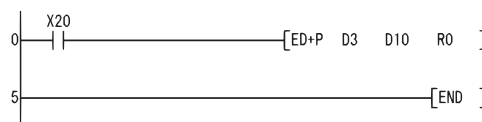
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range: (Error code: 4140)  
 $0, 2^{-1022} \leq | \text{Contents of designated device} | < 2^{1024}$
  - The value of the specified device is  $-0$ . (Error code: 4140)
  - The result of addition/subtraction exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Result of operation} |$  (Error code: 4141)

## Program Example

- (1) The following program adds the 64-bit floating decimal point type real numbers at D3 to D6 and the 64-bit floating decimal point type real numbers at D10 to D13 when X20 goes ON, and outputs the result at R0 to R3.

[Ladder Mode]



[List Mode]

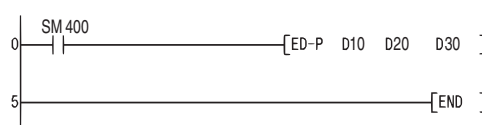
Step	Instruction	Device
0	LD	X20
1	ED+P	D3 D4 D5 D6 R0
5	END	

[Operation]

$$\begin{array}{|c|c|c|c|} \hline D6 & D5 & D4 & D3 \\ \hline \boxed{5961.437} & \boxed{ } & \boxed{ } & \boxed{ } \\ \hline \end{array} + \begin{array}{|c|c|c|c|} \hline D13 & D12 & D11 & D10 \\ \hline \boxed{12003.200} & \boxed{ } & \boxed{ } & \boxed{ } \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|c|} \hline R3 & R2 & R1 & R0 \\ \hline \boxed{17964.637} & \boxed{ } & \boxed{ } & \boxed{ } \\ \hline \end{array}$$

- (2) The following programs subtracts the 64-bit floating decimal point type real numbers at D20 to D23 from the 64-bit floating decimal point type real numbers at D10 to D13, and stores the result at D30 to D33.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	ED-P	D10 D11 D12 D13 D30
5	END	

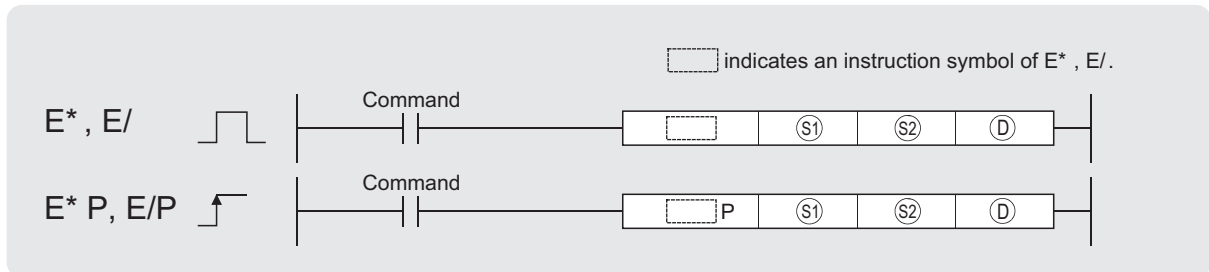
[Operation]

$$\begin{array}{|c|c|c|c|} \hline D13 & D12 & D11 & D10 \\ \hline \boxed{97365.203} & \boxed{ } & \boxed{ } & \boxed{ } \\ \hline \end{array} - \begin{array}{|c|c|c|c|} \hline D23 & D22 & D21 & D20 \\ \hline \boxed{76059.797} & \boxed{ } & \boxed{ } & \boxed{ } \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|c|} \hline D33 & D32 & D31 & D30 \\ \hline \boxed{21305.406} & \boxed{ } & \boxed{ } & \boxed{ } \\ \hline \end{array}$$

## 6.2.11 Multiplication and division of floating decimal point data (Single precision) (E\*(P),E/(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



- Ⓢ1 : Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (real number)
- Ⓢ2 : Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (real number)
- Ⓣ : Head number of the devices where the multiplication/division operation result will be stored (real number)

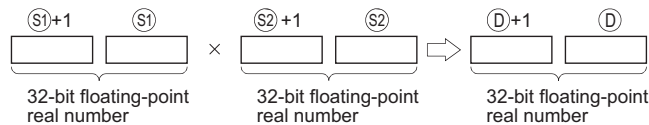
Setting Data	Internal Devices		R, ZR	JMO		UAGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○	—	○	—	○	—	○*1	—
Ⓢ2	—	○	—	○	—	○	—	○*1	—
Ⓣ	—	○	—	○	—	○	—	○*1	—

\*1: Available only in multiple Universal model QCPU

### ★ Function

#### E\*

- Multiplies the 32-bit floating decimal point real number designated by Ⓢ1 by the 32-bit floating decimal point real number designated by Ⓢ2 and stores the operation result at the device designated by Ⓣ.

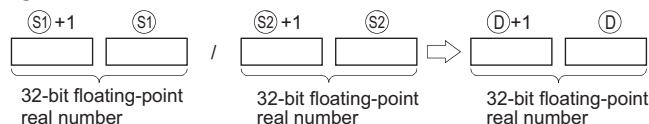


- Values which can be designated at Ⓢ1, Ⓢ2 and Ⓣ and which can be stored, are as follows:

$$0, 2^{-126} \leq | \text{Designated value (stored value)} | < 2^{128}$$

#### E/

- Divides the 32-bit floating decimal point real number designated by Ⓢ1 by the 32-bit floating decimal point real number designated by Ⓢ2 and stores the operation result at the device designated by Ⓣ.



- (2) Values which can be designated at  $\text{S}1$ ,  $\text{S}2$  and  $\text{D}$  and which can be stored, are as follows:

$$0, 2^{-126} \leq | \text{Designated value (stored value)} | < 2^{128}$$

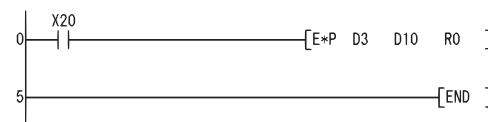
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device or the result of multiplication is not within the following range:  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$   
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
  - The value of the designated device is  $-0.^{*2}$   
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
- \*2: There are CPU modules that will not result in an operation error if -0 is specified.  
Refer to Section 3.2.4 for details.
- The result of multiplication and division exceeds the following range.  
 (The overflow occurs.)(For the Universal model QCPU only)  
 $2^{128} \leq | \text{Result of addition and subtraction} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program multiplies the 32-bit floating decimal point real numbers at D3 and D4 and the 32-bit floating decimal point real numbers at D10 and D11, and stores the result at R0 and R1.

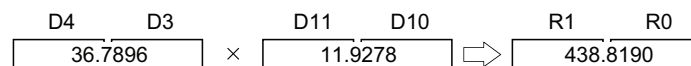
[Ladder Mode]



[List Mode]

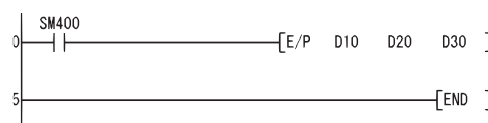
Step	Instruction	Device
0	LD	X20
1	E*P	D3 D10 R0
5	END	

[Operation]



- (2) The following program divides the 32-bit floating decimal point real numbers at D10 and D11 by the 32-bit floating decimal point real numbers at D20 and D21, and stores the result at D30 and D31.

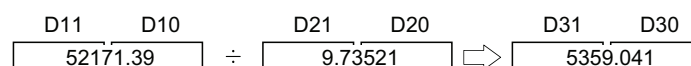
[Ladder Mode]



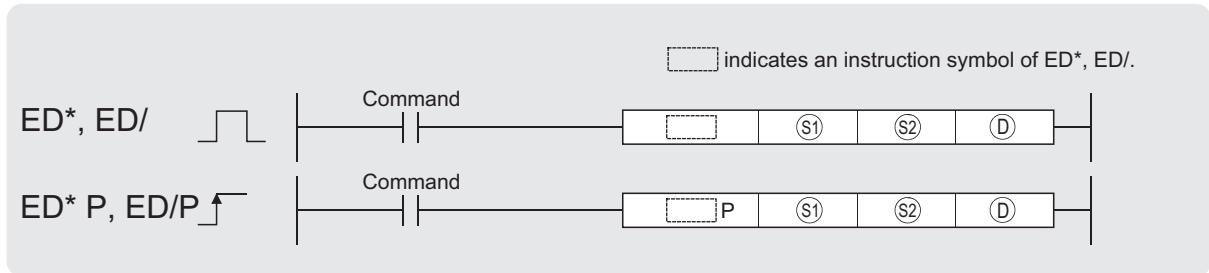
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	E/P	D10 D20 D30
5	END	

[Operation]



## 6.2.12 Multiplication and division of floating decimal point data (Double precision) (ED\*(P),ED/(P))



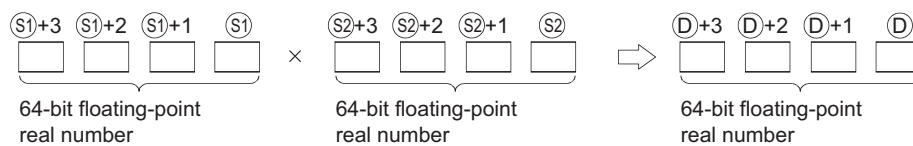
- Ⓢ<sub>1</sub> : Data to be multiplied/divided or head number of the devices where the data to be multiplied/divided is stored (real number)
- Ⓢ<sub>2</sub> : Data for multiplying/dividing or head number of the devices where the data for multiplying/dividing is stored (real number)
- ⓓ : Head number of the devices where the multiplication/division operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JAG		UAGD	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○				—		○	—
Ⓢ <sub>2</sub>	—	○				—		○	—
ⓓ	—	○				—		—	—

### ★ Function

#### ED\*

- (1) Multiplies the 64-bit floating decimal point real number designated by Ⓢ<sub>1</sub> by the 64-bit floating decimal point real number designated by Ⓢ<sub>2</sub> and stores the operation result at the device designated by ⓓ.

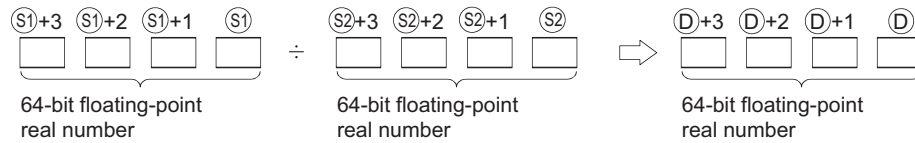


- (2) Values which can be designated at Ⓢ<sub>1</sub>, Ⓢ<sub>2</sub> and ⓓ and which can be stored, are as follows:  
 $0, 2^{-1022} \leq | \text{Designated value (stored value)} | < 2^{1024}$
- (3) When the operation results in -0 or an underflow, the result is processed as 0.



**ED/**

- (1) Divides the 64-bit floating decimal point real number designated by  $\textcircled{S1}$  by the 64-bit floating decimal point real number designated by  $\textcircled{S2}$  and stores the operation result at the device designated by  $\textcircled{D}$ .



- (2) Values which can be designated at  $\textcircled{S1}$ ,  $\textcircled{S2}$  and  $\textcircled{D}$  and which can be stored, are as follows:  
 $0, 2^{-1022} \leq | \text{Designated value (stored value)} | < 2^{1024}$
- (3) When the operation results in -0 or an underflow, the result is processed as 0.

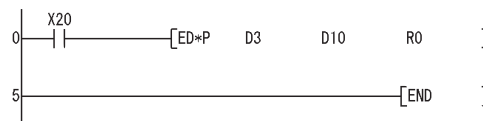
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device or the result of multiplication is not within the following range: (Error code: 4140)  
 $0, 2^{-1022} \leq | \text{Contents of designated device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The value of  $\textcircled{S2}$  at division operation is 0. (Error code: 4100)
  - The result of multiplication/division exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \leq | \text{Result of operation} |$

## Program Example

- (1) The following program multiplies the 64-bit floating decimal point real numbers at D3 to D6 and the 64-bit floating decimal point real numbers at D10 to D13, and stores the result at R0 to R3.

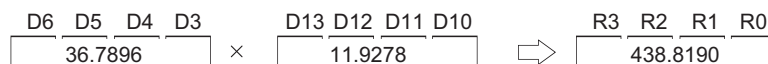
[Ladder Mode]



[List Mode]

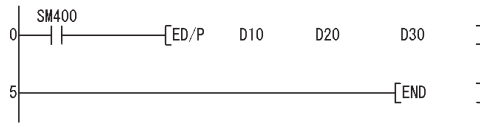
Step	Instruction	Device
0	LD	X20
1	ED*P	D3 D10 R0
5	END	

[Operation]



- (2) The following program divides the 64-bit floating decimal point real numbers at D10 to D13 by the 64-bit floating decimal point real numbers at D20 to D23, and stores the result at D30 to D33.

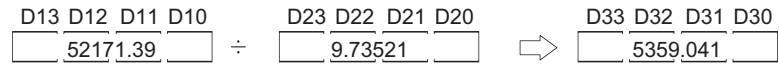
[Ladder Mode]



[List Mode]

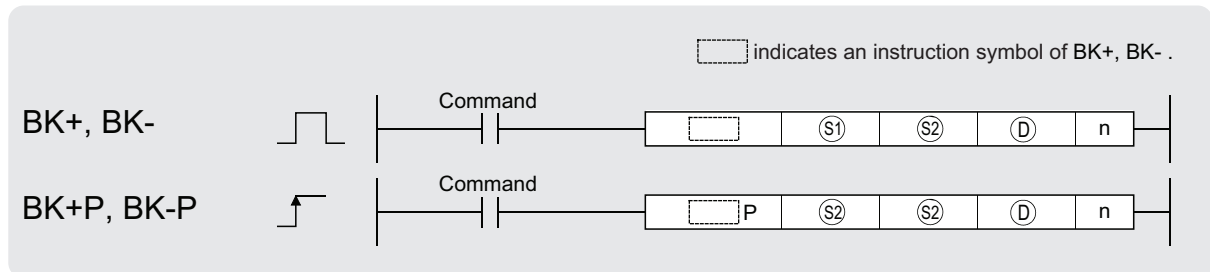
Step	Instruction	Device
0	LD	SM400
1	ED/P	D10
5	END	D30

[Operation]



## 6.2.13 Block addition and subtraction (BK+(P),BK-(P))

Basic High performance Process Redundant Universal



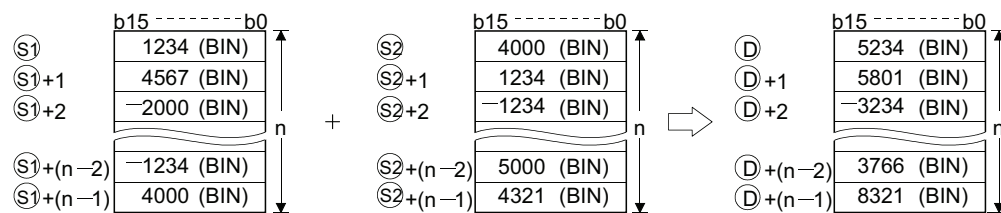
- Ⓢ<sub>1</sub> : Head number of the devices where the data to be added to/subtracted from is stored (BIN 16 bits)
- Ⓢ<sub>2</sub> : Data for adding/subtracting or head number of the devices where the data for adding/subtracting is stored (BIN 16 bits)
- ⓓ : Head number of the devices where the operation result will be stored (BIN 16 bits)
- n : Number of addition/subtraction data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○	○			—		—	—
Ⓢ <sub>2</sub>	—	○	○			—		○	—
ⓓ	—	○	○			—		—	—
n	○	○	○			○		○	—

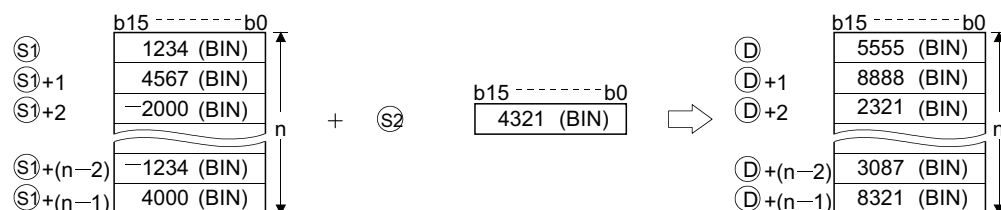
### ★ Function

#### BK+

- (1) Adds n points of BIN data from the device designated by Ⓢ<sub>1</sub> and n-points of BIN data from the device designated by Ⓢ<sub>2</sub> and stores the result from the device designated by ⓓ onward.



- (2) Block addition is performed in 16-bit units.
- (3) The constant designated by Ⓢ<sub>2</sub> can be between -32768 and 32767 (BIN 16-bit data).



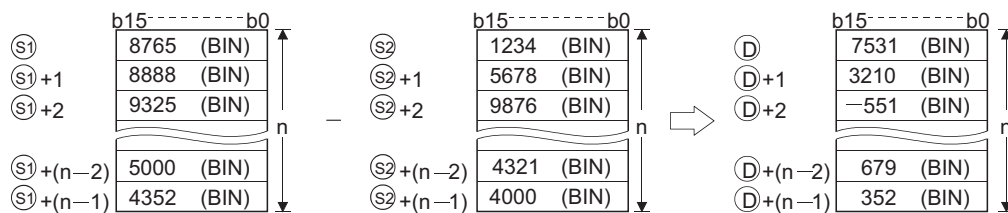
- (4) The following will happen when an underflow or overflow is generated in an operation result:  
The carry flag in this case does not go ON.

$$\begin{array}{r} \cdot K32767 + K2 \longrightarrow K-32767 \\ (7FFFH) \quad (0002H) \quad (8001H) \end{array}$$

$$\begin{array}{r} \cdot K-32767 + K-2 \longrightarrow K32767 \\ (8001H) \quad (FFFEH) \quad (7FFFH) \end{array}$$

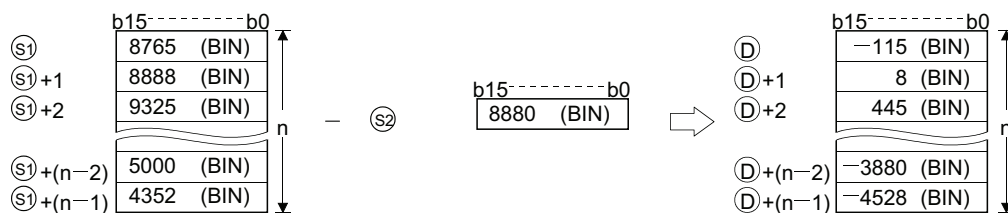
## BK-

- (1) Subtracts  $n$  points of BIN data from the device designated by  $\textcircled{S1}$  and  $n$ -points of BIN data from the device designated by  $\textcircled{S2}$  and stores the result from the device designated by  $\textcircled{D}$  onward.



- (2) Block subtraction is performed in 16-bit units.

- (3) The constant designated by  $\textcircled{S2}$  can be between  $-32768$  and  $32767$  (BIN 16-bit data).



- (4) The following will happen when an underflow or overflow is generated in an operation result:  
The carry flag in this case does not go ON.

$$\begin{array}{r} \cdot K-32768 - K2 \longrightarrow K32766 \\ (8000H) \quad (0002H) \quad (7FFE H) \end{array}$$

$$\begin{array}{r} \cdot K32767 - K-2 \longrightarrow -32767 \\ (7FFFH) \quad (FFFEH) \quad (8001H) \end{array}$$

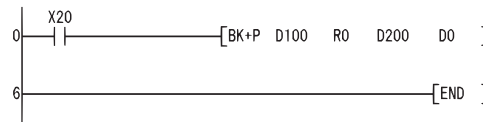
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $n$ -bit range from the  $\textcircled{S1}$ ,  $\textcircled{S2}$  or  $\textcircled{D}$  device exceeds the range of that device. (Error code: 4101)
  - The device ranges of  $\textcircled{S1}$  and  $\textcircled{D}$  overlap. (Except when the same device is assigned to  $\textcircled{S1}$  and  $\textcircled{D}$ ) (Error code: 4101)
  - The device ranges of  $\textcircled{S2}$  and  $\textcircled{D}$  overlap. (Except when the same device is assigned to  $\textcircled{S2}$  and  $\textcircled{D}$ ) (Error code: 4101)

## Program Example

- (1) The following program adds, when X20 is turned ON, the data stored at D100 to D103 to the data stored at R0 to R3 and stores the operation result into the area starting from D200.

[Ladder Mode]



[List Mode]

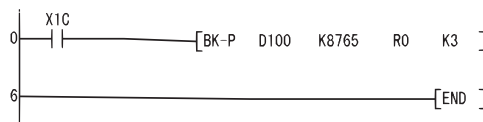
Step	Instruction	Device
0	LD	X20
1	BK+P	D100 R0 D200 D0
6	END	

[Operation]

b15-----b0			b15-----b0			b15-----b0	
D100	6789 (BIN)	+	R0	1234 (BIN)	⇒	D200	8023 (BIN)
D101	7821 (BIN)		R1	2032 (BIN)		D201	9853 (BIN)
D102	5432 (BIN)		R2	-3252 (BIN)		D202	2180 (BIN)
D103	3520 (BIN)		R3	-1000 (BIN)		D203	2520 (BIN)
D0							4

- (2) The following program subtracts, when X1C is turned ON, the constant 8765 from the data at D100 to D102 and stores the operation result into the area starting from R0.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	BK-P	D100 K8765 R0 K3
6	END	

[Operation]

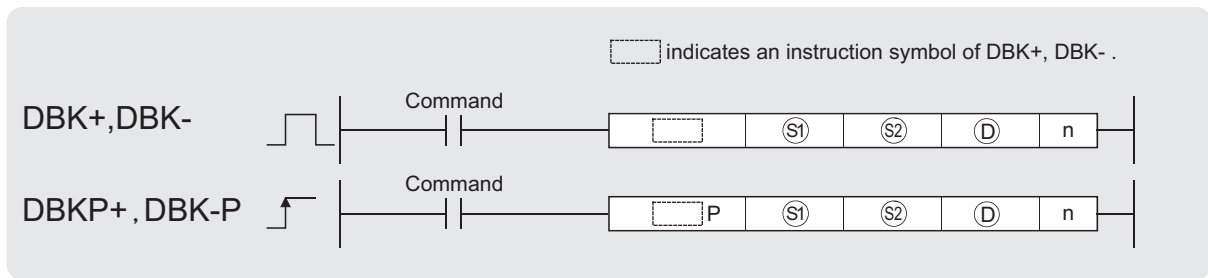
b15-----b0			b15-----b0			b15-----b0	
D100	12345 (BIN)	-		8765 (BIN)	⇒	R0	3580 (BIN)
D101	8701 (BIN)		R1	-64 (BIN)			
D102	3502 (BIN)		R2	-5263 (BIN)			

## 6.2.14 BIN 32-bit data block addition and subtraction operations (DBK+(P),DBK-(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ1: Head number of the devices where the data to be added and subtracted are stored (BIN 32 bits)

Ⓢ2: Addition and subtraction data or head number of the devices where the addition and subtraction data are stored (BIN 32 bits)

Ⓧ: Head number of the devices where the addition and subtraction operation result will be stored (BIN 32 bits)  
n: Number of addition and subtraction data blocks (BIN 16 bits)

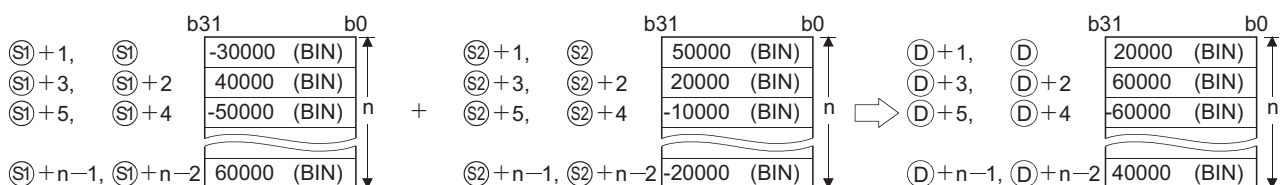
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K,H	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		—	—
Ⓢ2	—	○				—		○	—
Ⓧ	—	○				—		—	—
n	—	○				○		○	—

### ★ Function

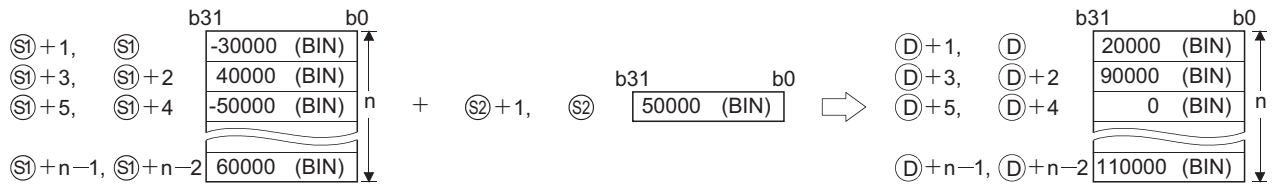
#### DBK+

- (1) This instruction adds BIN 32-bit data stored in n-point devices starting from the device specified by Ⓢ1 to BIN 32-bit data stored in n-point devices starting from the device specified by Ⓢ2 or a constant. and then stores the operation result into the nth device specified by Ⓧ and up,

When a device is specified for Ⓢ2



When a constant is specified for  $\textcircled{S2}$



- (2) Block addition is executed in 32-bit units.
- (3) The constant in the device specified by  $\textcircled{S2}$  can be between  $-2147483648$  to  $2147483647$  (BIN 32-bit data).
- (4) If the value specified by  $n$  is 0, the instruction will be not processed.
- (5) The following will happen if an overflow occurs in an operation result:  
The carry flag in this case is not turned on.

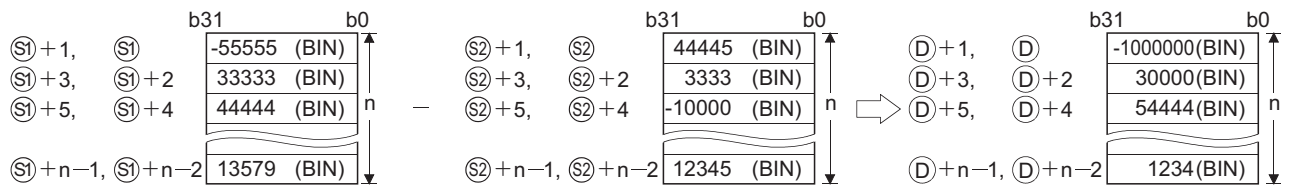
·  $K2147483647+K2 \longrightarrow K-2147483647$   
(7FFFFFFFH) (00000002H) (80000001H)

·  $K-2147483647+K-2 \longrightarrow K2147483647$   
(80000001H) (FFFFFFFEH) (7FFFFFFFH)

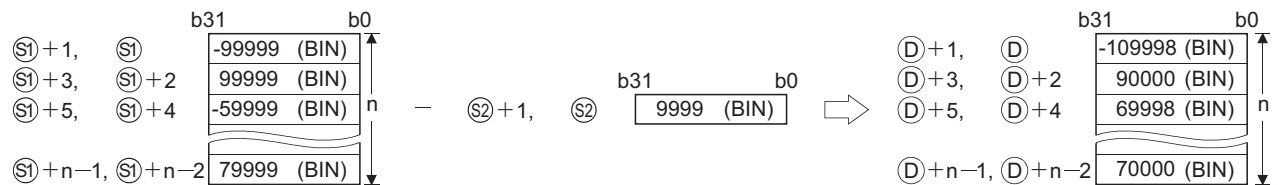
## DBK-

- (1) This instruction subtracts BIN 32-bit data stored in the  $n$ -point devices starting from the device specified by  $\textcircled{S2}$  or a constant from BIN 32-bit data stored in  $n$ -point devices starting from the device specified by  $\textcircled{S1}$ , and then stores the operation result into the  $n$ th device specified by  $\textcircled{D}$  and up,

When a device is specified for  $\textcircled{S2}$



When a constant is specified for  $\textcircled{S2}$



- (2) Block subtraction is executed in 32-bit units.
- (3) The constant in the device specified by  $\textcircled{S2}$  can be between  $-2147483648$  to  $2147483647$  (BIN 32-bit data).
- (4) If the value specified by  $n$  is 0, the instruction will be not processed.
- (5)  $\textcircled{D}$  specifies out of the range of  $n$ -point devices starting from the device specified by  $\textcircled{S1}$  and  $\textcircled{S2}$ .  
However,  $\textcircled{S1}$  and  $\textcircled{S2}$  can specify the same device.

- (6) The following will happen if an overflow occurs in an operation result:  
The carry flag in this case is not turned on.

- $K2147483647 - K2 \longrightarrow K-2147483647$   
(7FFFFFFFH)(00000002H) (80000001H)
- $K-2147483647 - K2 \longrightarrow K2147483647$   
(80000001H) (FFFFFFFEH) (7FFFFFFFH)

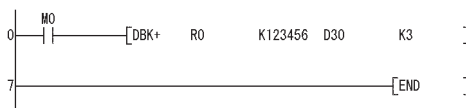


## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- A negative value is specified for n. (Error code: 4100)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S1}$ ,  $\textcircled{S2}$ , or  $\textcircled{D}$  exceeds the specified device range. (Error code: 4101)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S1}$  overlaps with the range of the n-point devices starting from the device specified by  $\textcircled{D}$ . (Exclude the case that  $\textcircled{S1}$  and  $\textcircled{D}$  specify the same device. (Error code: 4101)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S2}$  overlaps with the range of the n-point devices starting from the device specified by  $\textcircled{D}$ . (Error code: 4101)

- (1) The following program adds the value data stored at R0 to R5 to the constant, and then stores the operation result into D30 to D35, when M0 is turned on.

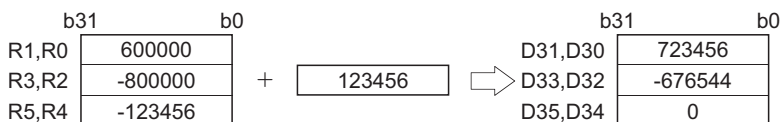
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DBK+	R0 K123456 D30 K3
7	END	

[Operation]



- (2) The following program subtracts the value data stored at D50 to D59 from the value data stored at D100 to D109, and then stores the operation result into R100 to R109, when M0 is turned on.

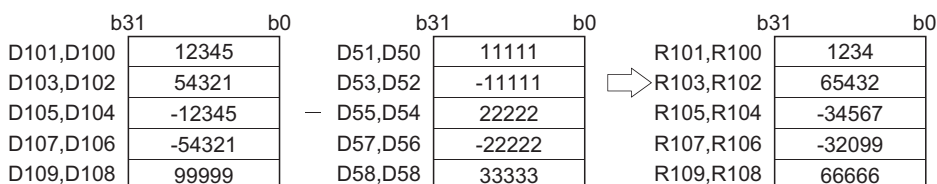
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DBK-	D100 D50 R100 K5
6	END	

[Operation]

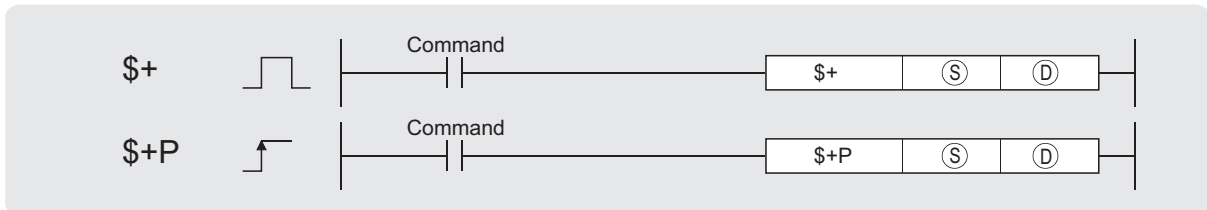




## 6.2.15 Linking character strings (\$+(P))



1 When two data are set ( $\text{Ⓢ} + \text{Ⓣ} \rightarrow \text{Ⓣ}$ )

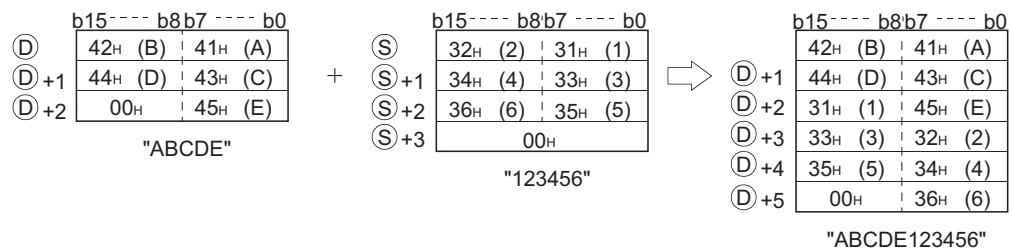


Ⓢ : Data for linking or head number of the devices where the data for linking is stored (character string)  
 Ⓣ : Head number of the devices where the data to be linked is stored (character string)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- Links the character string data designated by Ⓢ after the character string data designated by Ⓣ and stores the result into the area starting with the device number designated by Ⓣ. The object of character string data is that character string data stored from device numbers designated at Ⓣ and Ⓢ to that stored at "00H".



- When character strings are linked, the "00H", which indicates the end of character string data designated at Ⓣ, is ignored, and the character string designated at Ⓢ is appended to the last character of the Ⓣ string.

6

6.2 Arithmetic Operation Instructions  
 6.2.15 Linking character strings (\$+(P))

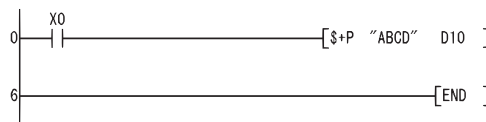
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The entire character string linked from the device number designated by ⑤ to the final device number of the relevant device cannot be stored. (Error code: 4101)
  - The storage device numbers for the character strings designated by ⑤ and ⑥ overlap. (Error code: 4101)
  - The character string of ⑤ and ⑥ exceeds 16383 characters. (Error code: 4101)

## Program Example

- (1) The following program links the character string stored from D10 to D12 to the character string "ABCD" when X0 is ON.

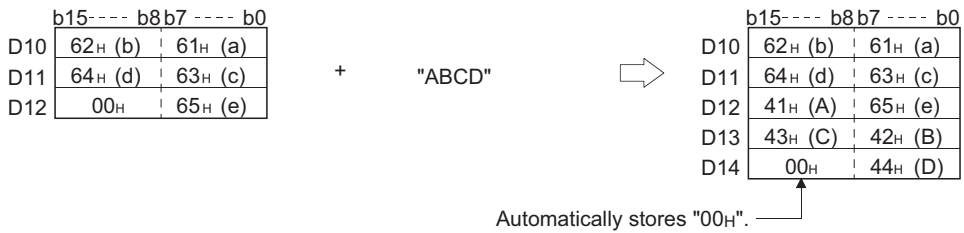
[Ladder Mode]



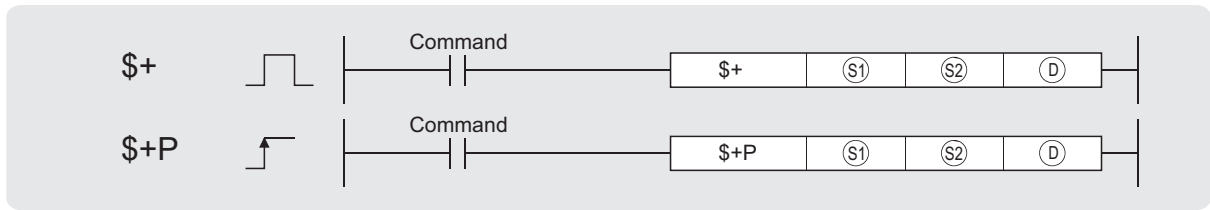
[List Mode]

Step	Instruction	Device
0	LD	X0
1	\$+P	"ABCD" D10
6	END	

[Operation]



2 When three data are set (S1+S2 → D)

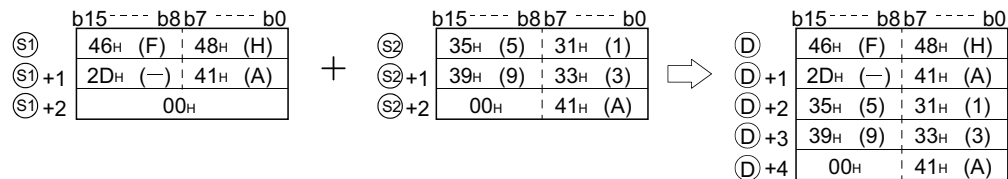


- Ⓢ<sub>1</sub> : Data for linking or head number of the devices where the data for linking is stored (character string)
- Ⓢ<sub>2</sub> : Data to be linked or head number of the devices where the data to be linked is stored (character string)
- ⓓ : Head number of the devices where the linking result will be stored (character string)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○				—		○	—
Ⓢ <sub>2</sub>	—	○				—		○	—
ⓓ	—	○				—		—	—

★ Function

- (1) Links the character string data designated by Ⓢ<sub>2</sub> after the character string data designated by Ⓢ<sub>1</sub> and stores the result into the area starting with the device number designated by ⓓ.



- (2) When character strings are linked, the "00H" which indicates the end of character string data indicated by Ⓢ<sub>1</sub>, is ignored, and the character string indicated by Ⓢ<sub>2</sub> is appended to the last character of the Ⓢ<sub>1</sub> string.

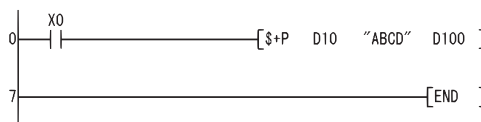
## ⚠ Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The entire character string linked from the device number designated by ① to the final device number of the relevant device cannot be stored. (Error code: 4101)
  - The storage device numbers for the character strings designated by ① and ② overlap. (Error code: 4101)
  - The storage device numbers for the character strings designated by ② and ③ overlap. (Error code: 4101)
  - The character string of ①, ② and ③ exceeds 16383 characters. (Error code: 4101)

## 📄 Program Example

- (1) The following program links the character string stored from D10 to D12 with the character string "ABCD" when X0 is ON, and stores them in D100 onwards.

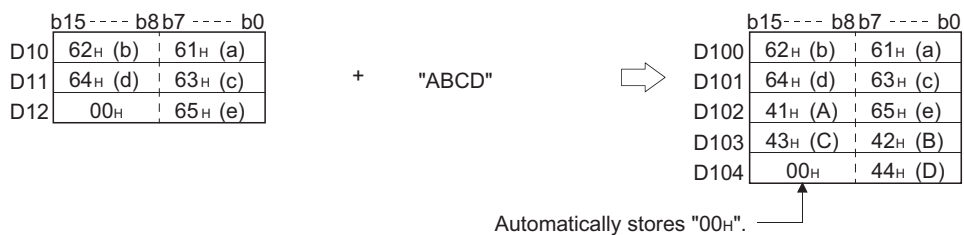
[Ladder Mode]



[List Mode]

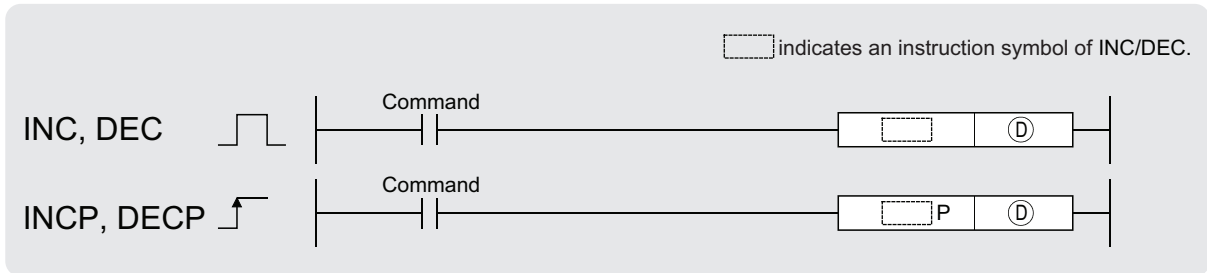
Step	Instruction	Device
0	LD	X0
1	S+P	D10 "ABCD" D100
7	END	

[Operation]



## 6.2.16 Incrementing and decrementing 16-bit BIN data (INC(P),DEC(P))

Basic High performance Process Redundant Universal



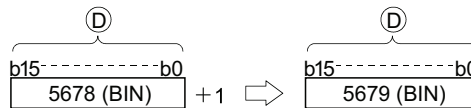
Ⓧ : Head number of devices for INC (+1)/DEC (-1) operation (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ				○					—

### ★ Function

#### INC

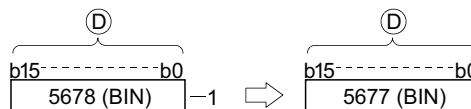
- (1) Adds 1 to the device designated by Ⓧ (16-bit data).



- (2) When INC/INCP operation is executed for the device designated by Ⓧ, whose content is 32767, the value -32768 is stored at the device designated by Ⓧ.

#### DEC

- (1) Subtracts 1 from the device designated by Ⓧ (16-bit data).



- (2) When DEC/DECP operation is executed for the device designated by Ⓧ, whose content is -32768, the value 32767 is stored at the device designated by Ⓧ.

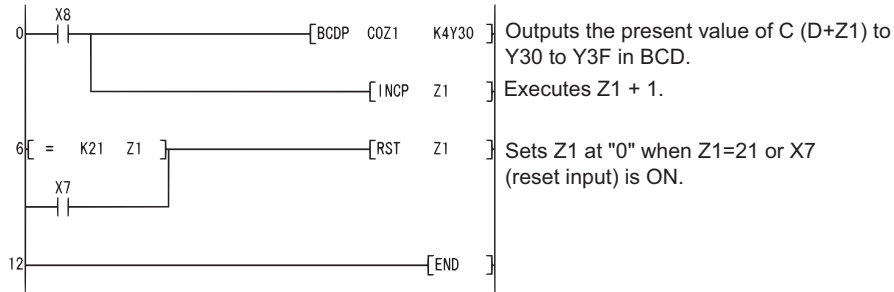
### ! Operation Error

- (1) There are no operation errors associated with the INC(P)/DEC(P) instruction.

## Program Example

- (1) The following program outputs the present value at the counter C0 to C20 to the area Y30 to Y3F in BCD, every time X8 is turned ON. (When present value is less than 9999)

[Ladder Mode]

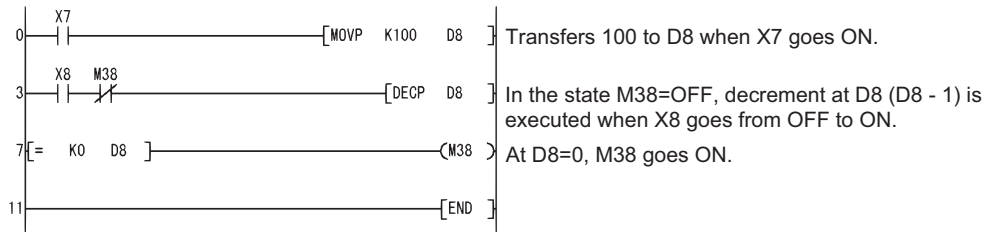


[List Mode]

Step	Instruction	Device
0	LD	X8
1	BCDP	COZ1 K4Y30
4	INCP	Z1
6	LD=	K21 Z1
9	OR	X7
10	RST	Z1
12	END	

- (2) The following is a down counter program.

[Ladder Mode]

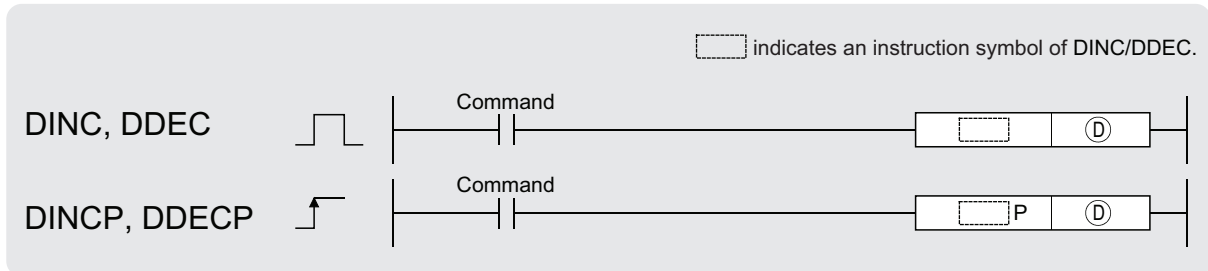


[List Mode]

Step	Instruction	Device
0	LD	X7
1	MOV P	K100 D8
3	LD	X8
4	ANI	M38
5	DECP	D8
7	LD=	K0 D8
10	OUT	M38
11	END	

## 6.2.17 Incrementing and decrementing 32-bit BIN data (DINC(P),DDEC(P))

Basic High performance Process Redundant Universal



Ⓧ : Head number of devices for DINC(+1) or DDEC(-1) operation (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ									—

### ★ Function

#### DINC

- (1) Adds 1 to the device designated by Ⓧ (32-bit data).

$$\begin{array}{|c|c|} \hline \text{Ⓧ+1} & \text{Ⓧ} \\ \hline \text{b31--b16} & \text{b15--b0} \\ \hline \text{73500 (BIN)} & \end{array} + 1 \Rightarrow \begin{array}{|c|c|} \hline \text{Ⓧ+1} & \text{Ⓧ} \\ \hline \text{b31--b16} & \text{b15--b0} \\ \hline \text{73501 (BIN)} & \end{array}$$

- (2) When DINC/DINCP operation is executed for the device designated by Ⓧ, whose content is 2147483647, the value -2147483648 is stored at the device designated by Ⓧ.

#### DDEC

- (1) Subtracts -1 from the device designated by Ⓧ (32-bit data).

$$\begin{array}{|c|c|} \hline \text{Ⓧ+1} & \text{Ⓧ} \\ \hline \text{b31--b16} & \text{b15--b0} \\ \hline \text{73500 (BIN)} & \end{array} - 1 \Rightarrow \begin{array}{|c|c|} \hline \text{Ⓧ+1} & \text{Ⓧ} \\ \hline \text{b31--b16} & \text{b15--b0} \\ \hline \text{73499 (BIN)} & \end{array}$$

- (2) When DDEC/DDECP operation is executed for the device designated by Ⓧ, whose content is 0, the value -1 is stored at the device designated by Ⓧ.

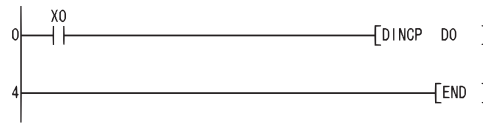
### ! Operation Error

- (1) There are no operation errors associated with the DINC(P) or DDEC(P).

## Program Example

- (1) The following program adds 1 to the data at D0 and D1 when X0 is ON.

[Ladder Mode]

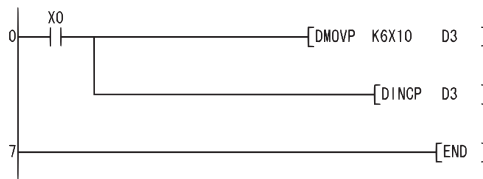


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DINC P	D0
4	END	

- (2) The following program adds 1 to the data set at X10 to X27 when X0 goes ON, and stores the result at D3 and D4.

[Ladder Mode]

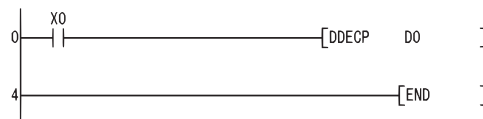


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DMOV P	K6X10 D3
4	DINC P	D3
7	END	

- (3) The following program subtracts 1 from the data at D0 and D1 when X0 goes ON.

[Ladder Mode]

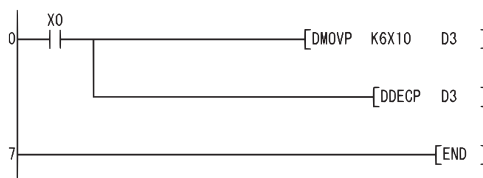


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DDEC P	D0
4	END	

- (4) The following program subtracts 1 from the data set at X10 to X27 when X0 goes ON, and stores the result at D3 and D4.

[Ladder Mode]



[List Mode]

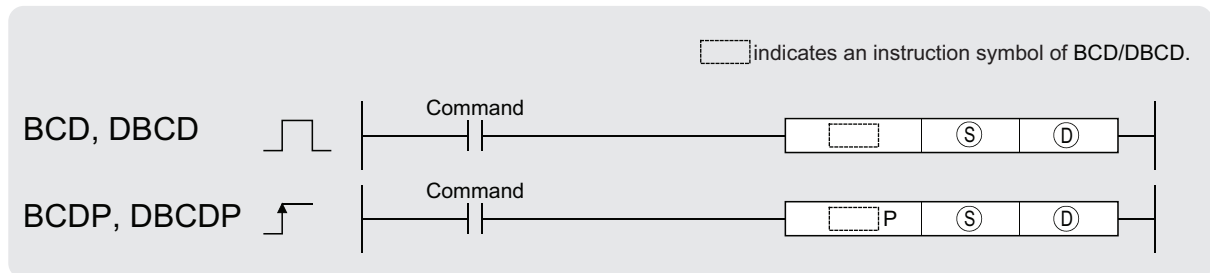
Step	Instruction	Device
0	LD	X0
1	DMOV P	K6X10 D3
4	DDEC P	D3
7	END	



## 6.3 Data conversion instructions

### 6.3.1 Conversion from BIN data to 4-digit and 8-digit BCD (BCD(P), DBCD(P))

Basic High performance Process Redundant Universal



Ⓢ : BIN data or head number of the devices where the BIN data is stored (BIN 16/32 bits)

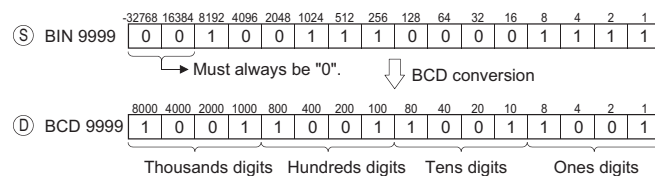
Ⓣ : Head number of the devices where BCD data will be stored (BCD 4/8 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—

## ★ Function

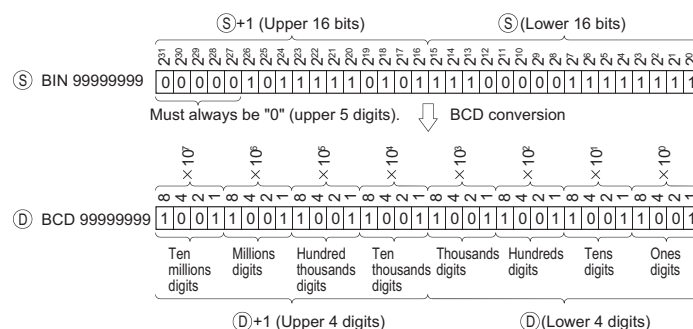
### BCD

Converts BIN data (0 to 9999) at the device designated by Ⓢ to BCD data, and stores it at the device designated by Ⓣ.



### DBCD

Converts BIN data (0 to 99999999) at the device designated by Ⓢ to BCD data, and stores it at the device designated by Ⓣ.

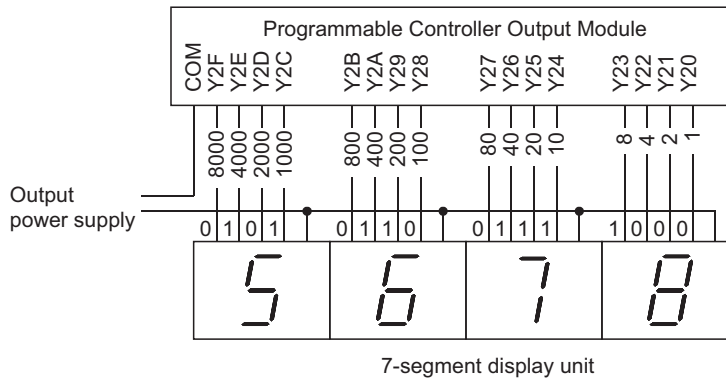


## Operation Error

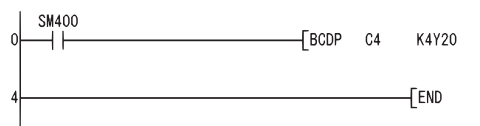
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data of (S) is other than 0 to 9999 at BCD instruction. (Error code: 4100)
  - The data of (S) or (S) +1 is other than 0 to 99999999 at DBCD instruction. (Error code: 4100)

## Program Example

- (1) The following program outputs the present value of C4 from Y20 to Y2F to the BCD display device.



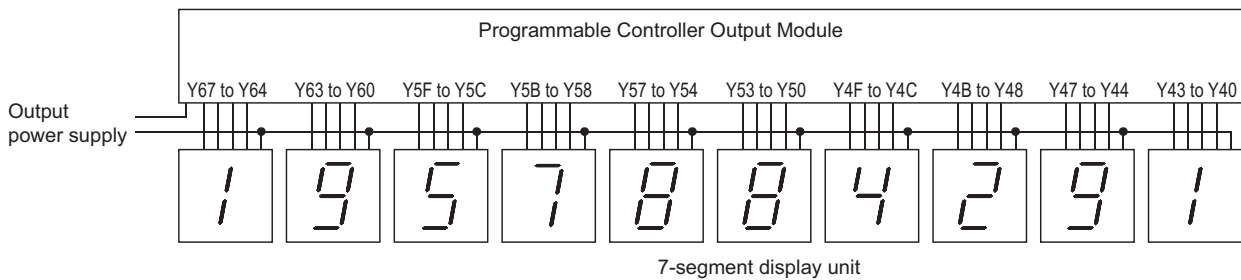
[Ladder Mode]



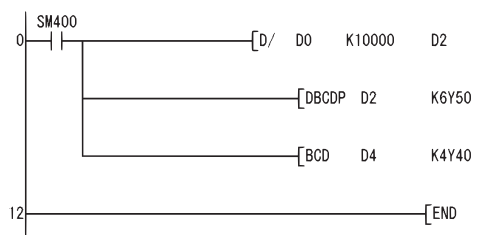
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BCDP	C4
2	AND	K4Y20
3	END	

- (2) The following program outputs 32-bit data from D0 to D1 to Y40 to Y67.



[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	D/	D0
2	DBCDP	D2
3	BCD	D4
4	END	

## 6.3.2 Conversion from BCD 4-digit and 8-digit data to BIN data (BIN(P),DBIN(P))

Basic High performance Process Redundant Universal



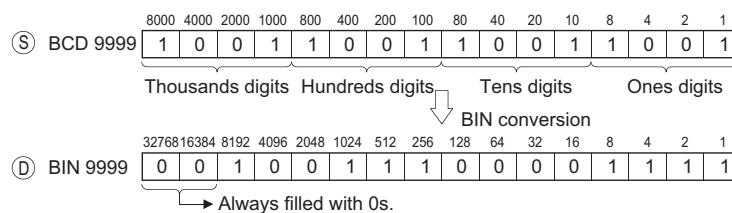
- Ⓢ : BCD data or head number of the devices where the BCD data is stored (BCD 4/8 digits)
- Ⓣ : Head number of the devices where BIN data will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

### ★ Function

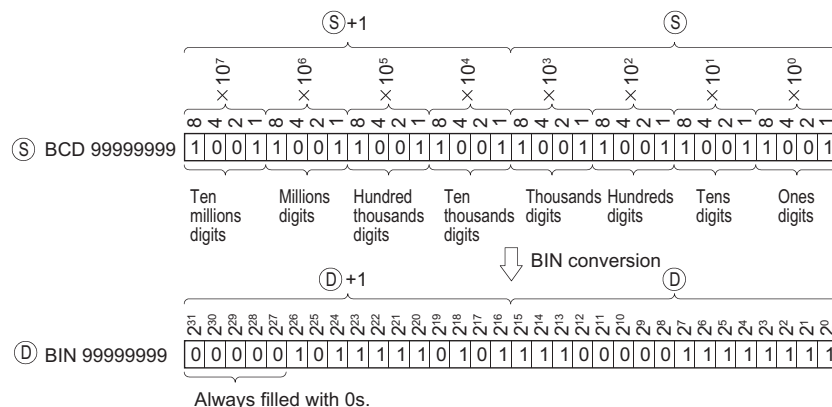
#### BIN

Converts BCD data (0 to 9999) at device designated by Ⓢ to BIN data, and stores at the device designated by Ⓣ.



#### DBIN

Converts BCD data (0 to 99999999) at device designated by Ⓢ to BIN data, and stores at the device designated by Ⓣ.



6

6.3 Data conversion instructions  
6.3.2 Conversion from BCD 4-digit and 8-digit data to BIN data (BIN(P),DBIN(P))

## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- When values other than 0 to 9 are designated to any digits of  $\text{Ⓢ}$

(Error code: 4100)

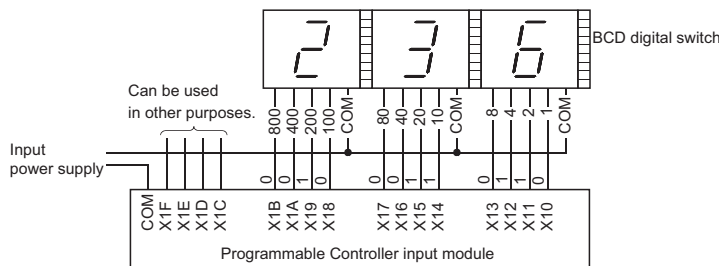
When the QCPU is used, the error above can be suppressed by turning ON SM722.

However, the instruction is not executed regardless of whether SM722 is turned ON or OFF if the designated value is out of the available range.

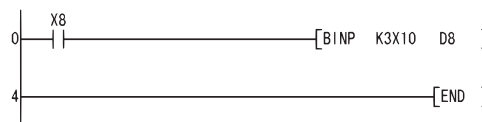
For the BINP/DBINP instruction, the next operation will not be performed until the command (execution condition) is turned from OFF to ON regardless of the presence/absence of an error.

## Program Example

- (1) The following program converts the BCD data at X10 to X1B to BIN when X8 is ON, and stores it at D8.



### [Ladder Mode]

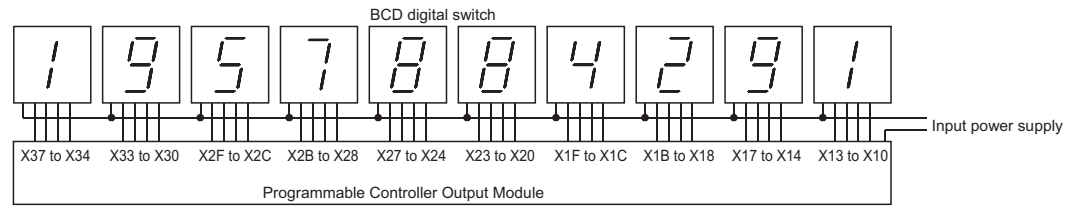


### [List Mode]

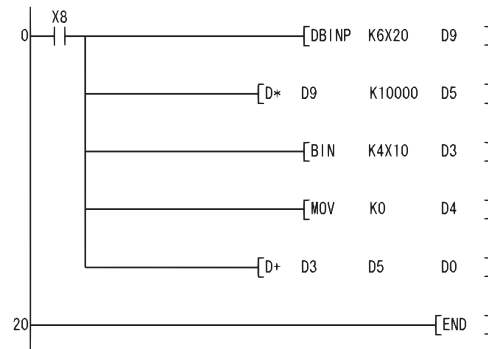
Step	Instruction	Device
0	LD	X8
1	BINP	K3X10 D8
4	END	

- (2) The following program converts the BCD data at X10 to X37 to BIN when X8 is ON, and stores it at D0 and D1.

(Addition of the BIN data converted from BCD at X20 to X37 and the BIN data converted from BCD at X10 to X1F)



[Ladder Mode]



[List Mode]

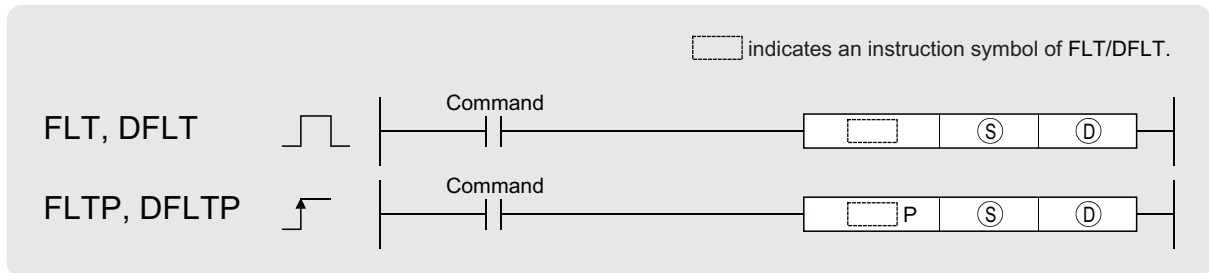
Step	Instruction	Device
0	LD	X8
1	DBINP	K6X20 D9
4	D*	D9 K10000 D5
9	BIN	K4X10 D3
12	MOV	K0 D4
14	D+	D3 D5 D0
20	END	

If the data set at X10 to X37 is a BCD value which exceeds 2147483647, the value at D0 and D1 will be a negative value, because it exceeds the range of numerical values that can be handled by a 32-bit device.

### 6.3.3 Conversion from BIN 16 and 32-bit data to floating decimal point (Single precision) (FLT(P),DFLT(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



- Ⓢ : Integer data to be converted to 32-bit floating decimal point data or head number of the devices where the integer data is stored (BIN 16/32 bits)
- Ⓣ : Head number of the devices where the converted 32-bit floating decimal point data will be stored (real number)

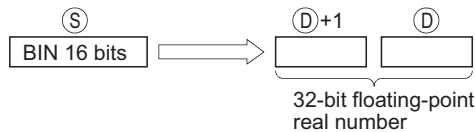
Setting Data	Internal Devices		R, ZR	JMO		UAGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○		○	○		○		—
Ⓣ	—	○		—	○		○*1		—

\*1: Available only in multiple Universal model QCPU

## ★ Function

### FLT

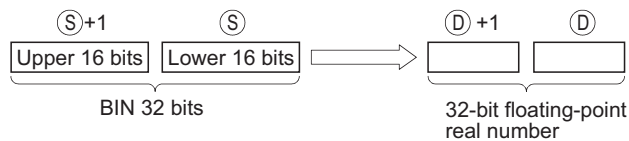
- (1) Converts 16-bit BIN data designated by Ⓢ to 32-bit floating decimal point type real number, and stores at device number designated by Ⓣ.



- (2) BIN values between -32768 to 32767 can be designated by Ⓢ.

### DFLT

- (1) Converts 32-bit BIN data designated by Ⓢ to 32-bit floating decimal point type real number, and stores at device number designated by Ⓣ.

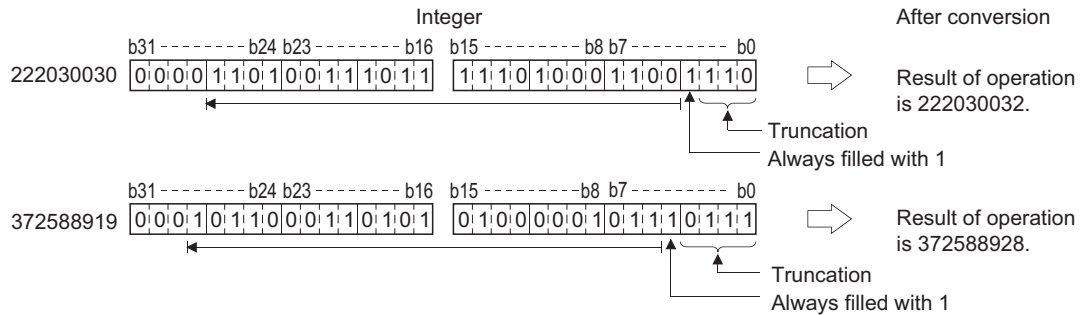


- (2) BIN values between -2147483648 to 2147483647 can be designated by Ⓢ+1 and Ⓢ.

- (3) Due to the fact that 32-bit floating decimal point type real numbers are processed by simple 32-bit processing, the number of significant digits is 24 bits if the display is binary and approximately 7 digits if the display is decimal.

For this reason, if the integer exceeds the range of -16777216 to 16777215 (24-bit BIN value), errors can be generated in the conversion value.

As for the conversion result, the 25th bit from the upper bit of the integer is always filled with 1 and 26th bit and later bits are truncated.



## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The result exceeds the following range (The overflow occurs.)  
(For the Universal model QCPU only)

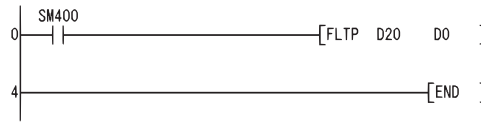
$$2^{128} \cong | \text{Operation result} |$$

(Error code: 4141)

## Program Example

- (1) The following program converts the BIN 16-bit data at D20 to a 32-bit floating decimal point type real number and stores the result at D0 and D1.

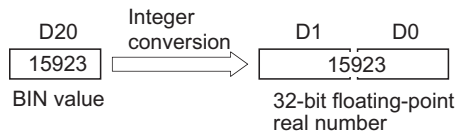
[Ladder Mode]



[List Mode]

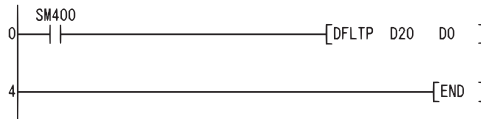
Step	Instruction	Device
0	LD	SM400
1	FLTP	D20 D0
4	END	

[Operation]



- (2) The following program converts the BIN 32-bit data at D20 and D21 to a 32-bit floating decimal point type real number, and stores the result at D0 and D1.

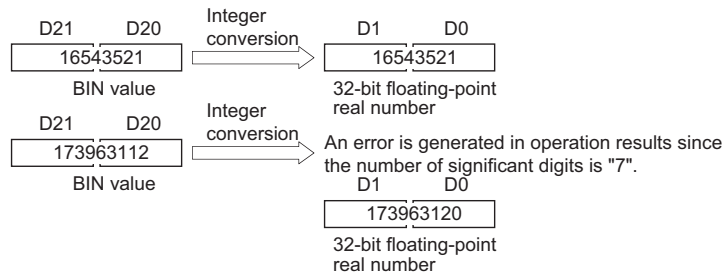
[Ladder Mode]



[List Mode]

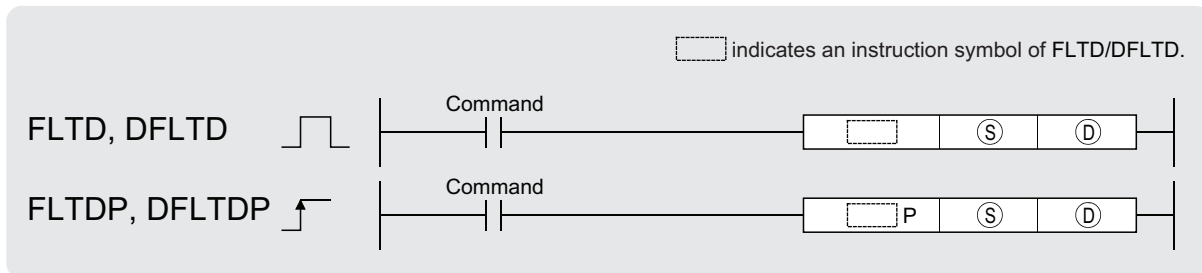
Step	Instruction	Device
0	LD	SM400
1	DFLTP	D20 D0
4	END	

[Operation]





### 6.3.4 Conversion from BIN 16 and 32-bit data to floating decimal point (Double precision) (FLTD(P),DFLTD(P))



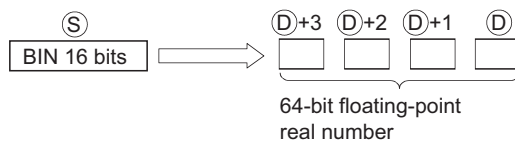
- Ⓢ : Integer data to be converted to 64-bit floating decimal point data or head number of the devices where the integer data is stored (BIN 16/32 bits)
- Ⓣ : Head number of the devices where the converted 64-bit floating decimal point data will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:V:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○		—			○		—
Ⓣ	—	○		—			—		—

## ★ Function

### FLTD

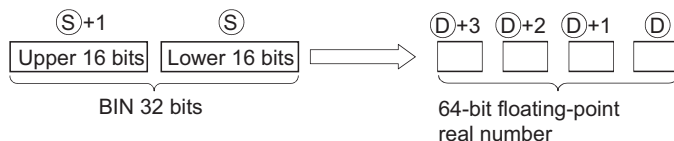
- (1) Converts 16-bit BIN data designated by Ⓢ to 64-bit floating decimal point type real number, and stores at device number designated by Ⓣ.



- (2) BIN values between -32768 to 32767 can be designated by Ⓢ.

### DFLTD

- (1) Converts 32-bit BIN data designated by Ⓢ to 64-bit floating decimal point type real number, and stores at device number designated by Ⓣ.



- (2) BIN values between -2147483648 to 2147483647 can be designated by Ⓢ +1 and Ⓢ.

6.3 Data conversion instructions  
6.3.4 Conversion from BIN 16 and 32-bit data to floating decimal point (Double precision) (FLTD(P),DFLTD(P))

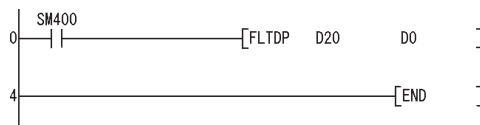
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The result exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Operation result} |$  (Error code: 4141)

## Program Example

- (1) The following program converts the BIN 16-bit data at D20 to a 64-bit floating decimal point type real number and stores the result at D0 to D3.

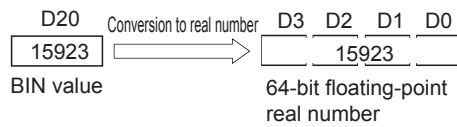
[Ladder Mode]



[List Mode]

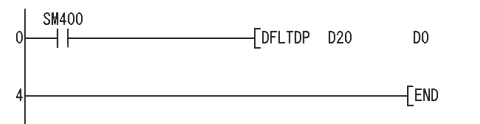
Step	Instruction	Device
0	LD	SM400
1	FLTDP	D20 D0
4	END	

[Operation]



- (2) The following program converts the BIN 32-bit data at D20 and D21 to a 64-bit floating decimal point type real number, and stores the result at D0 to D3.

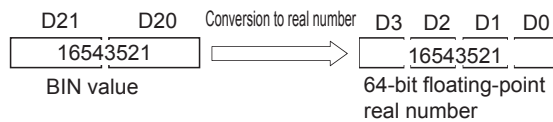
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DFLTDP	D20 D0
4	END	

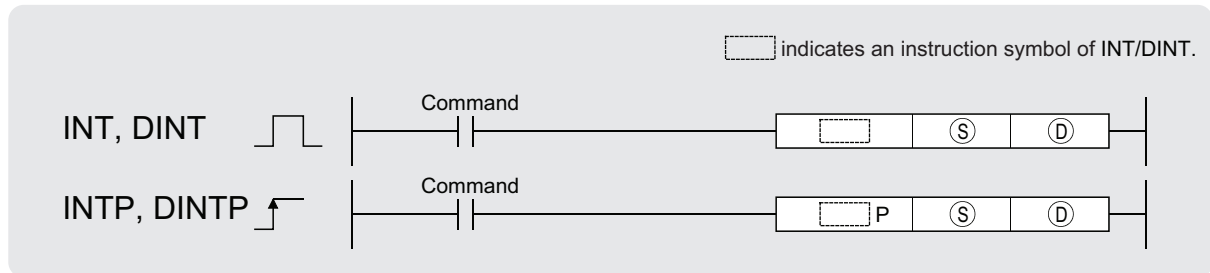
[Operation]



## 6.3.5 Conversion from floating decimal point data to BIN16- and 32-bit data (Single precision) (INT(P),DINT(P))

Ver.  
Basic High performance Process Redundant Universal

Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : 32-bit floating decimal point data to be converted to BIN value or head number of the devices where the floating decimal point data is stored (real number)

Ⓣ : Head number of the devices where the converted BIN value will be stored (BIN 16/32 bits)

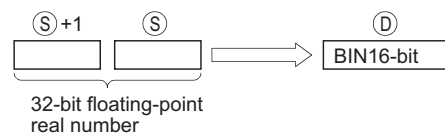
Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	—	○	○*1	○	—	—
Ⓣ	○	○	○	○	○	○	—	—	—

\*1: Available only in multiple Universal model QCPU

### ★ Function

#### INT

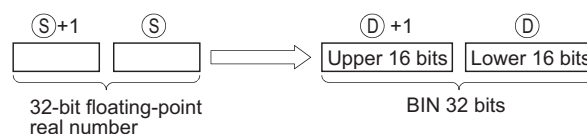
- Converts the 32-bit floating decimal point real number designated at Ⓢ into BIN 16-bit data and stores it at the device number designated at Ⓣ.



- The range of 32-bit floating decimal point type real numbers that can be designated at Ⓢ + 1 or Ⓢ is from  $-32768$  to  $32767$ .
- Stores integer values stored at Ⓣ as BIN 16-bit values.
- After conversion, the first digit after the decimal point of the real number is rounded off.

#### DINT

- Converts 32-bit floating decimal point type real number designated by Ⓢ to BIN 32-bit data, and stores the result at the device number designated by Ⓣ.



- The range of 32-bit floating decimal point type real numbers that can be designated at Ⓢ + 1 or Ⓢ is from  $-2147483648$  to  $2147483647$ .

- (3) The integer value stored at  $\text{D} + 1$  and  $\text{D}$  is stored as BIN 32 bits.
- (4) After conversion, the first digit after the decimal point of the real number is rounded off.



## Operation Error

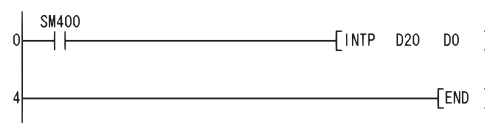
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range(For the Universal model QCPU only):  
 $0, 2^{-126} \leq |\text{Contents of designated device}| < 2^{128}$  (Error code: 4140)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)
  - The 32-bit floating decimal point type data designated by  $\text{S}$  when the INT instruction was used was outside the  $-31768$  to  $32767$  range. (Error code: 4100)
  - The 32-bit floating decimal point type data designated by  $\text{S}$  when the DINT instruction was used was outside the  $-2147483648$  to  $2147483647$  range. (Error code: 4100)



## Program Example

- (1) The following program converts the 32-bit floating decimal point type real number at D20 and D21 to BIN 16-bit data, and stores the result at D0.

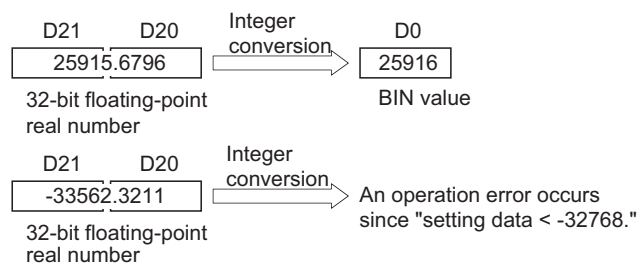
[Ladder Mode]



[List Mode]

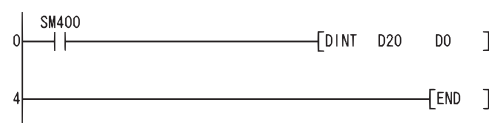
Step	Instruction	Device
0	LD	SM400
1	INTP	D20 D0
4	END	

[Operation]



- (2) The following program converts the 32-bit floating decimal point type real number at D20 and D21 to BIN 32-bit data and stores the result at D0 and D1.

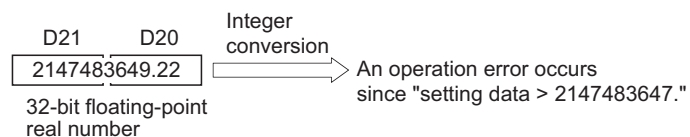
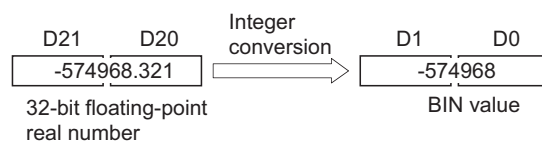
[Ladder Mode]



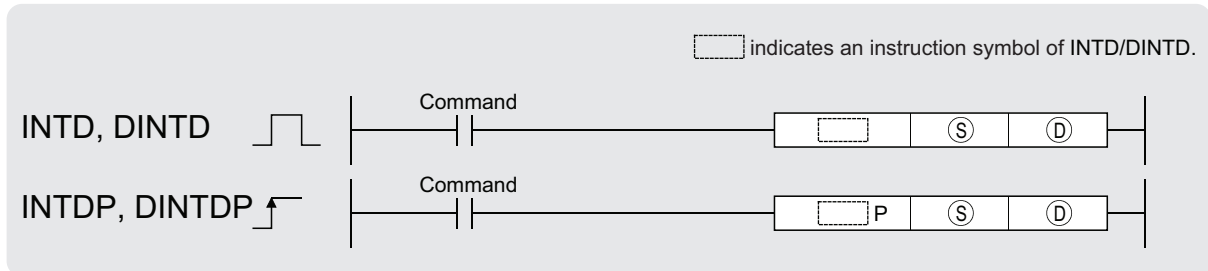
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DINT	D20 D0
4	END	

[Operation]



## 6.3.6 Conversion from floating decimal point data to BIN16- and 32-bit data (Double precision) (INTD(P),DINTD(P))



Ⓢ : 64-bit floating decimal point data to be converted to BIN value or head number of the devices where the floating decimal point data is stored (real number)

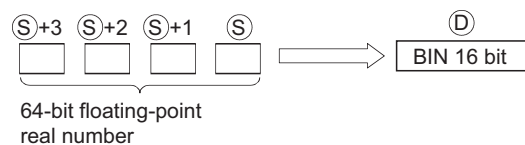
Ⓣ : Head number of the devices where the converted BIN value will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	JMO		UAGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○			—		—	○	—
Ⓣ	—	○			—		○	—	—

### ★ Function

#### INTD

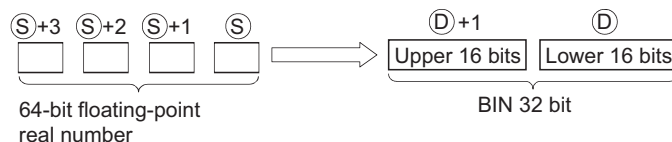
- (1) Converts the 64-bit floating decimal point real number designated at Ⓢ into BIN 16-bit data and stores it at the device number designated at Ⓣ.



- (2) The range of 64-bit floating decimal point type real numbers that can be designated at Ⓢ+3, Ⓢ+2, Ⓢ+1 or Ⓢ is from  $-32768$  to  $32767$ .
- (3) Stores integer values stored at Ⓣ as BIN 16-bit values.
- (4) The converted data is the value rounded 64-bit floating-point real number to the first digit after the decimal point.

#### DINTD

- (1) Converts 64-bit floating decimal point type real number designated by Ⓢ to BIN 32-bit data, and stores the result at the device number designated by Ⓣ.



- (2) The range of 64-bit floating decimal point type real numbers that can be designated at  $\textcircled{S}+3, \textcircled{S}+2, \textcircled{S}+1$  or  $\textcircled{S}$  is from  $-2147483648$  to  $2147483647$ .
- (3) The integer value stored at  $\textcircled{D}+1$  and  $\textcircled{D}$  is stored as BIN 32 bits.
- (4) The converted data is the value rounded 64-bit floating-point real number to the first digit after the decimal point.

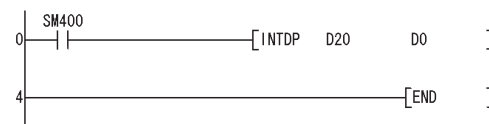
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0, 2^{-1022} \leq |\text{value of specified device}| \leq 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The 64-bit floating decimal point type data designated by  $\textcircled{S}$  when the INTD instruction was used was outside the  $-31768$  to  $32767$  range. (Error code: 4100)
  - The 64-bit floating decimal point type data designated by  $\textcircled{S}$  when the DINTD instruction was used was outside the  $-2147483648$  to  $2147483647$  range. (Error code: 4100)

## Program Example

- (1) The following program converts the 64-bit floating decimal point type real number at D20 to D23 with BIN 16-bit data, and stores the result at D0.

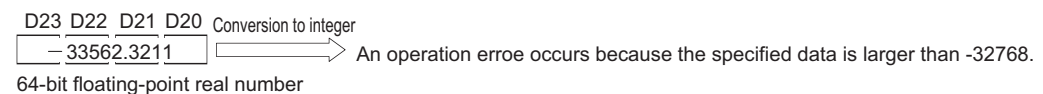
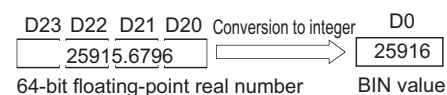
[Ladder Mode]



[List Mode]

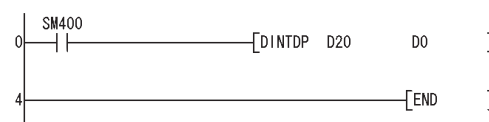
Step	Instruction	Device
0	LD	SM400
1	INTDP	D20 D0
4	END	

[Operation]



- (2) The following program converts the 64-bit floating decimal point type real number at D20 to D23 with BIN 32-bit data and stores the result at D0 and D1.

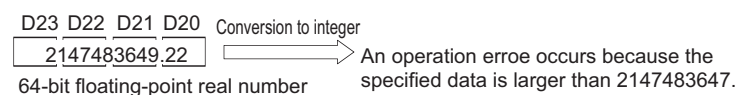
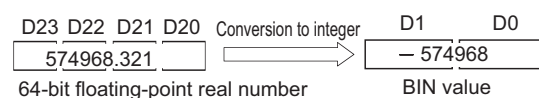
[Ladder Mode]



[List Mode]

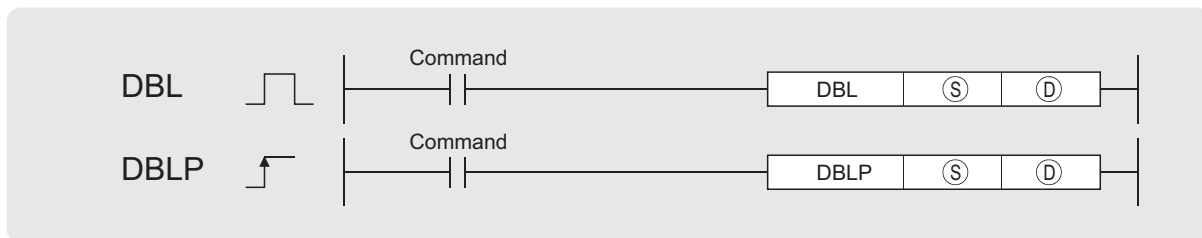
Step	Instruction	Device
0	LD	SM400
1	DINTDP	D20 D0
4	END	

[Operation]



## 6.3.7 Conversion from BIN 16-bit to BIN 32-bit data (DBL(P))

Basic High performance Process Redundant Universal

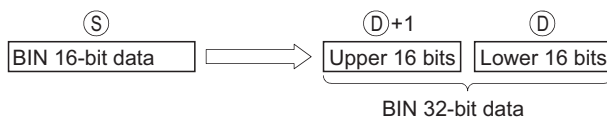


Ⓢ : BIN 16-bit data or head number of the devices where the BIN 16-bit data is stored (BIN 16 bits)  
 Ⓣ : Head number of the devices where the converted BIN 32-bit data will be stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

### ★ Function

Converts BIN 16-bit data at device designated by Ⓢ to BIN 32-bit data with sign, and stores the result at a device designated by Ⓣ.



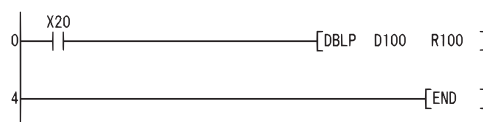
### ! Operation Error

(1) There are no errors associated with the DBL(P) instruction.

### 📄 Program Example

(1) The following program converts the BIN 16-bit data stored at D100 to BIN 32-bit data when X20 is ON, and stores at R100 and R101.

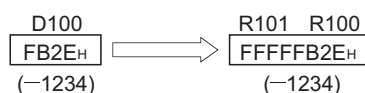
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	DBLP	D100 R100
4	END	

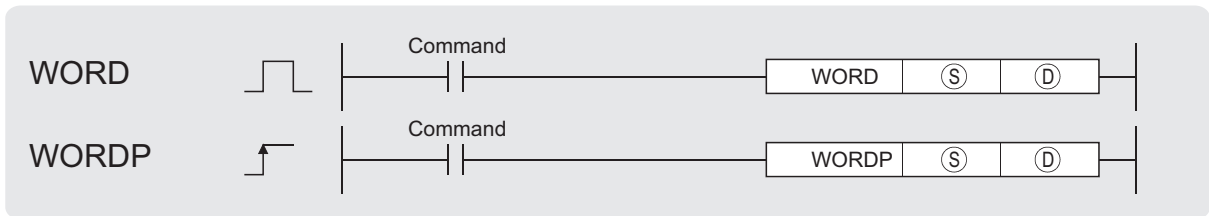
[Operation]





# 6.3.8 Conversion from BIN 32-bit to BIN 16-bit data (WORD(P))

Basic High performance Process Redundant Universal



Ⓢ : BIN 32-bit data or head number of the devices where the BIN 32-bit data is stored (BIN 32 bits)

Ⓣ : Head number of the devices where the converted BIN 16-bit data will be stored (BIN 16 bits)

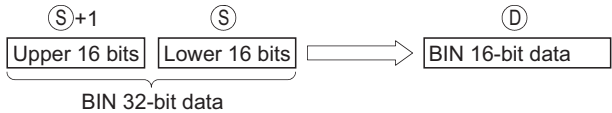
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—

6

## ★ Function

Converts BIN 32-bit data at device designated by Ⓢ to BIN 16-bit data with sign, and stores the result at a device designated by Ⓣ.

Devices can be designated in the range from -32768 to 32767.



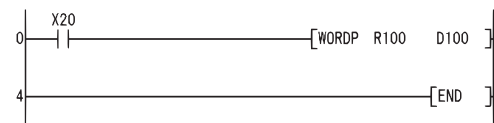
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The contents of the data designated by Ⓢ+1 and Ⓢ are outside the range of -32768 to 32767. (Error code: 4100)

## 📄 Program Example

- (1) The following program converts the BIN 32-bit data at R100 and R101 to BIN 16-bit data when X20 is ON, and stores it at D100.

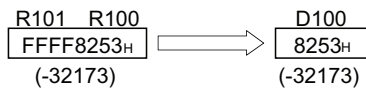
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	WORDP	R100 D100
4	END	

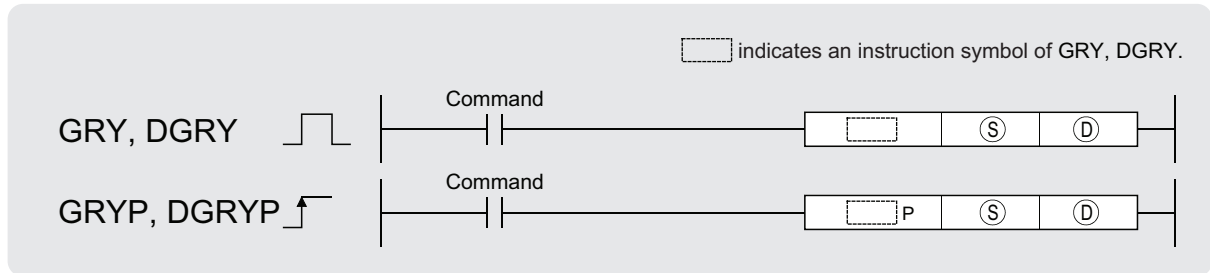
[Operation]



6.3 Data conversion instructions  
6.3.8 Conversion from BIN 32-bit to BIN 16-bit data (WORD(P))

## 6.3.9 Conversion from BIN 16 and 32-bit data to Gray code (GRY(P),DGRY(P))

Basic High performance Process Redundant Universal



Ⓢ : BIN data or head number of the devices where the BIN data is stored (BIN 16/32 bits)

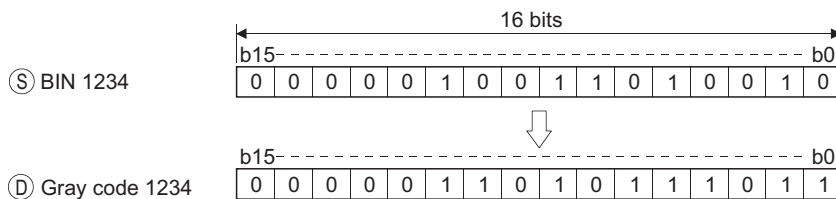
Ⓣ : Head number of the devices where the converted Gray code will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	JAG		UAGD	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—

### ★ Function

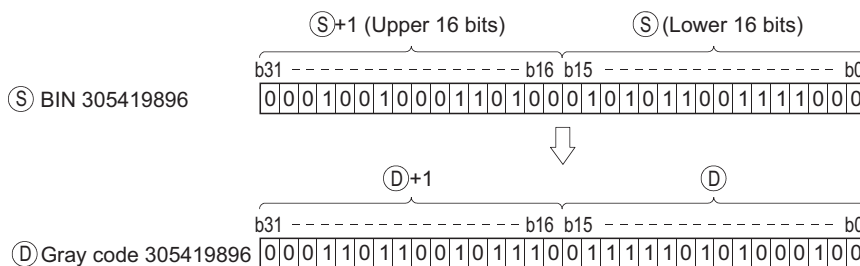
#### GRY

Converts BIN 16-bit data at the device designated by Ⓢ to Gray code, and stores result at device designated by Ⓣ.



#### DGRY

Converts BIN 32-bit data at the device designated by Ⓢ to Gray code, and stores result at device designated by Ⓣ.



## ! Operation Error

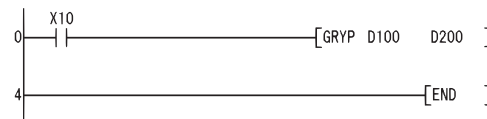
(1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The data at ⑤ is a negative number. (Error code: 4100)

## Program Example

(1) The following program converts the BIN data at D100 to Gray code when X10 is ON, and stores result at D200.

[Ladder Mode]

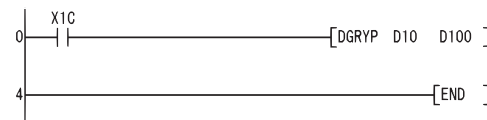


[List Mode]

Step	Instruction	Device
0	LD	X10
1	GRYP	D100 D200
4	END	

(2) The following program converts the BIN data at D10 and D11 to Gray code when X1C is ON, and stores it at D100 and D101.

[Ladder Mode]

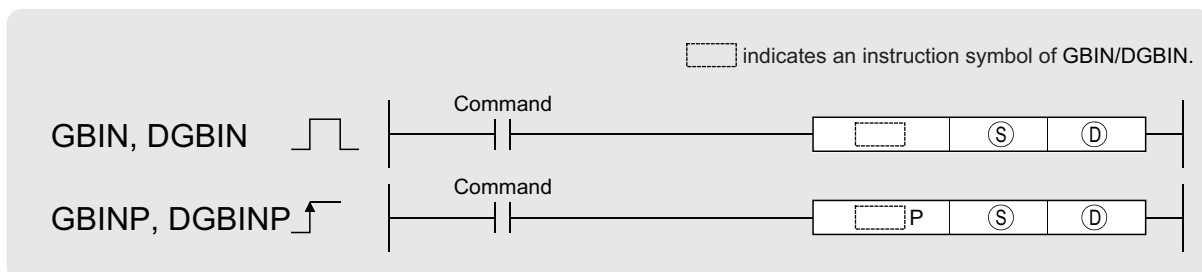


[List Mode]

Step	Instruction	Device
0	LD	X1C
1	DGRYP	D10 D100
4	END	

## 6.3.10 Conversion of Gray code to BIN 16 and 32-bit data (GBIN(P),DGBIN(P))

Basic High performance Process Redundant Universal



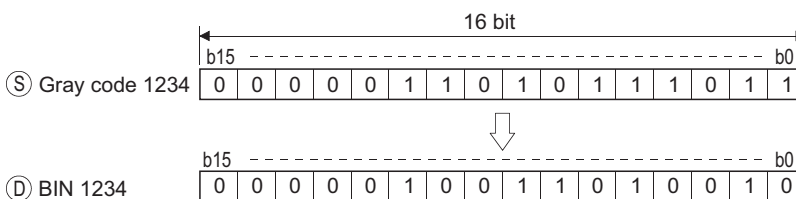
Ⓢ : Gray code data or head number of the devices where the Gray code data is stored (BIN 16/32 bits)  
 Ⓣ : Head number of the devices where the converted BIN data will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—

### ★ Function

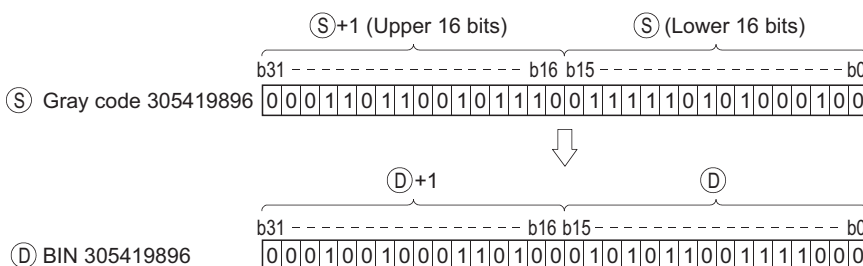
#### GBIN

Converts Gray code data at device designated by Ⓢ to BIN 16-bit data and stores at device designated by Ⓣ.



#### DGBIN

Converts Gray code data at device designated by Ⓢ to BIN 32-bit data and stores at device designated by Ⓣ.



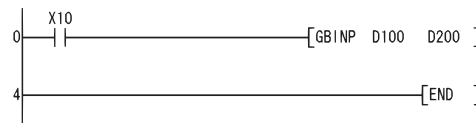
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Data at (S) when the GBIN instruction was issued is outside the 0 to 32767 range.  
(Error code: 4100)
  - Data at (S) when the DGBIN instruction was issued is outside the 0 to 2147483647 range.  
(Error code: 4100)

## Program Example

- (1) The following program converts the Gray code data at D100 when X10 is ON to BIN data, and stores the result at D200.

[Ladder Mode]

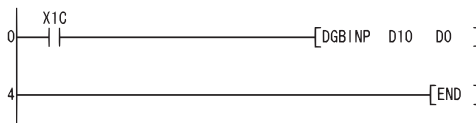


[List Mode]

Step	Instruction	Device
0	LD	X10
1	GBINP	D100 D200
4	END	

- (2) The following program converts the Gray code data at D10 and D11 to BIN data when X1C is ON, and stores the result at D0 and D1.

[Ladder Mode]

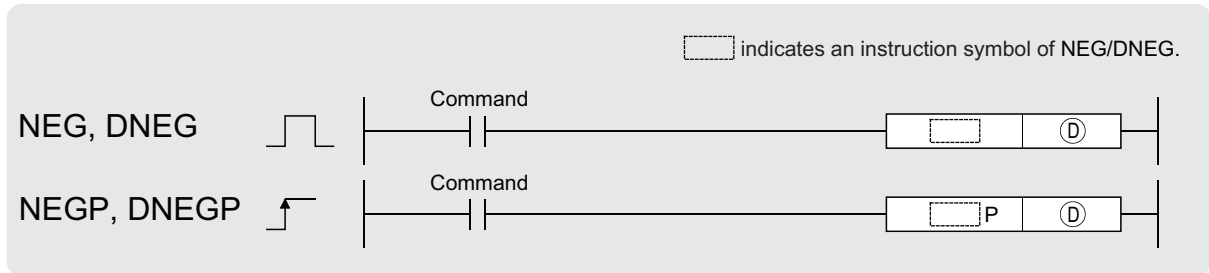


[List Mode]

Step	Instruction	Device
0	LD	X1C
1	DGBINP	D10 D11 D0 D1
4	END	

### 6.3.11 Complement of 2 of BIN 16- and 32-bit data (sign reversal) (NEG(P),DNEG(P))

Basic High performance Process Redundant Universal



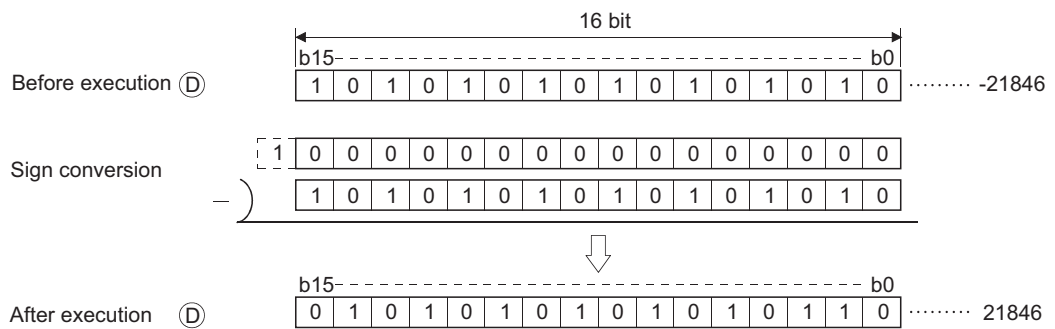
Ⓧ : Head number of the devices where the data for which complement of 2 is performed is stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ					○				—

## ★ Function

### NEG

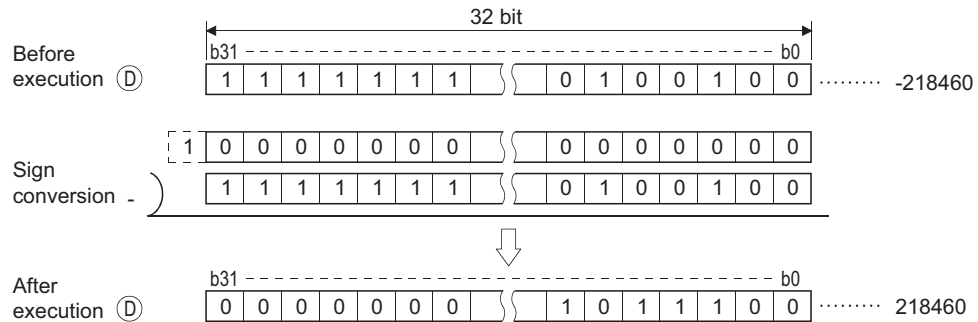
- (1) Reverses the sign of the 16-bit device designated by Ⓧ and stores at the device designated by Ⓧ.



- (2) Used when reversing positive and negative signs.

## DNEG

- (1) Reverses the sign of the 32-bit device designated by  $\text{D}$  and stores at the device designated by  $\text{D}$ .



- (2) Used when reversing positive and negative signs.

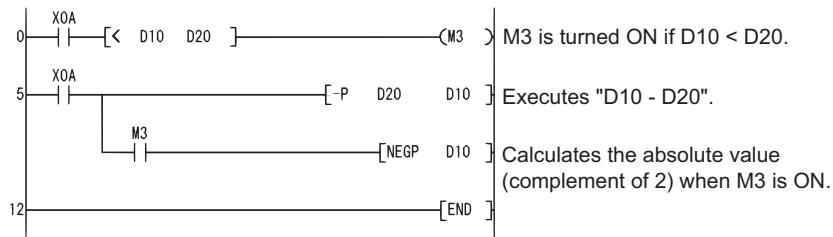
## ! Operation Error

- (1) There are no operation errors associated with the NEG(P) or DNEG(P) instruction.

## Program Example

- (1) The following program calculates a total for the data at D10 through D20 when XA goes ON, and seeks an absolute value if the result is negative.

[Ladder Mode]



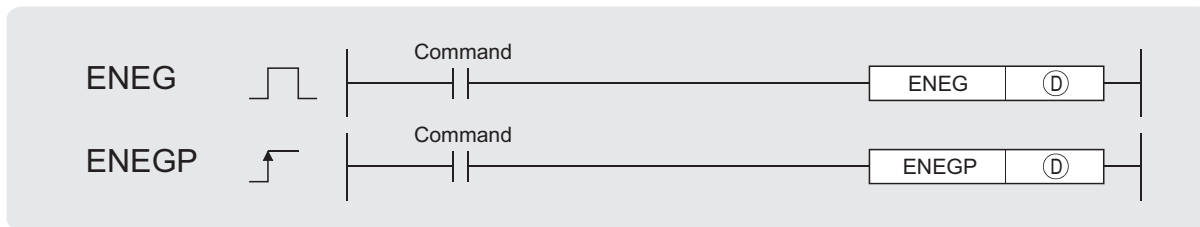
[List Mode]

Step	Instruction	Device
0	LD	XOA
1	AND<	D10 D20
4	OUT	M3
5	LD	XOA
6	-P	D20 D10
9	AND	M3
10	NEG P	D10
12	END	

## 6.3.12 Floating-point sign inversion (Single precision) (ENEG(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓧ : Head number of the devices where the 32-bit floating decimal point data whose sign is to be reversed is stored (real number)

Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ	—	○		—	○		○*1		—

\*1: Available only in multiple Universal model QCPU

### ★ Function

- Reverses the sign of the 32-bit floating decimal point type real number data designated by Ⓧ, and stores at the device designated by Ⓧ.
- Used when reversing positive and negative signs.

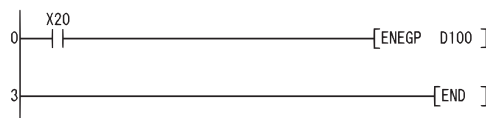
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The contents of the designated device or the result of the addition are not "0", or not within the following range (For the Universal model QCPU only):  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$  (Error code: 4140)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ . (For the Universal model QCPU only) (Error code: 4140)

### 📄 Program Example

- The following program inverts the sign of the 32-bit floating decimal point type real number data at D100 and D101 when X20 goes ON, and stores result at D100 and D101.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	ENEGP	D100
3	END	

[Operation]





## 6.3.13 Floating-point sign inversion (Double precision) (EDNEG(P))



Ⓧ : Head number of the devices where the 64-bit floating decimal point data whose sign is to be reversed is stored (real number)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ	—	○					—		

### ★ Function

- Reverses the sign of the 64-bit floating decimal point type real number data designated by Ⓧ, and stores at the device designated by Ⓧ.
- Used when reversing positive and negative signs.

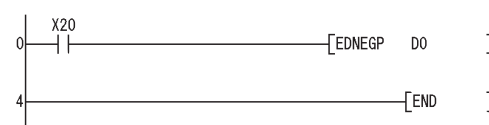
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0, 2^{-1022} \leq |\text{value of specified device}| < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)

### 📄 Program Example

- The following program inverts the sign of the 64-bit floating decimal point type real number data at D100 to D103 when X20 goes ON, and stores result at D100 to D103.

[Ladder Mode]



[List Mode]

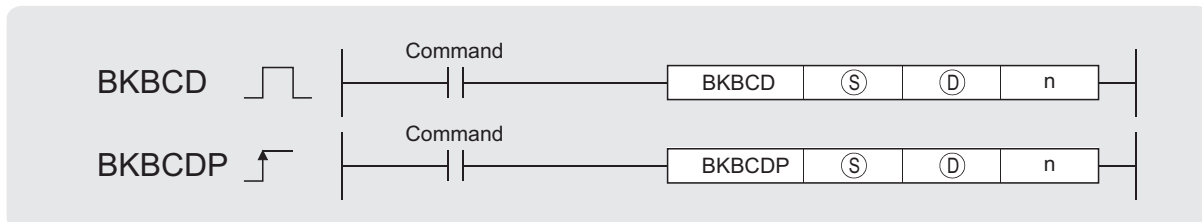
Step	Instruction	Device
0	LD	X20
1	EDNEGP	D0
3	END	

[Operation]



# 6.3.14 Conversion from block BIN 16-bit data to BCD 4-digit data (BKBCD(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the devices where BIN data is stored (BIN 16 bits)

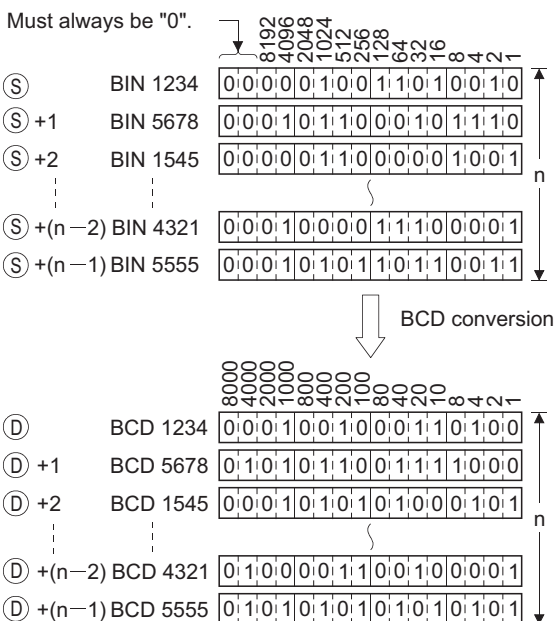
Ⓣ : Head number of the devices where the converted BCD data will be stored (BCD 4 digits)

n : Number of variable data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
Ⓣ	—	○				—			—
n	○	○				○			—

## ★ Function

- (1) Converts BIN data (0 to 9999) n points from device designated by Ⓢ to BCD, and stores result following the device designated by Ⓣ.



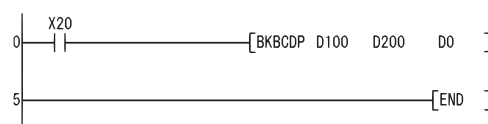
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range of the device n points from a device designated by  $\textcircled{S}$ ,  $\textcircled{D}$  or exceeds the relevant device. (Error code: 4101)
  - The data n points from the device designated by  $\textcircled{S}$  is outside the 0 to 9999 range. (Error code: 4100)
  - The  $\textcircled{S}$  and  $\textcircled{D}$  devices overlap. (Error code: 4101)

## Program Example

- (1) The following program converts, when X20 is turned ON, the BIN data stored at D100 to D102 to BCD and stores the operation result into the area starting from D200.

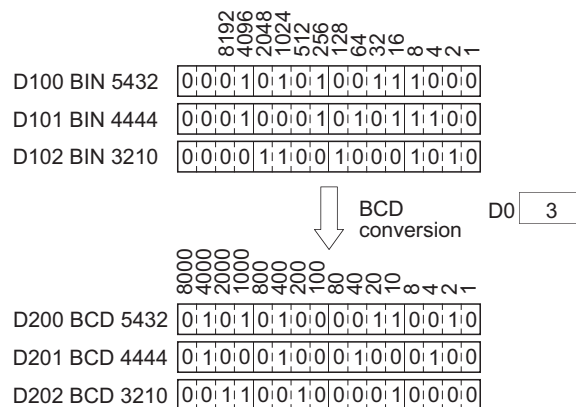
[Ladder Mode]



[List Mode]

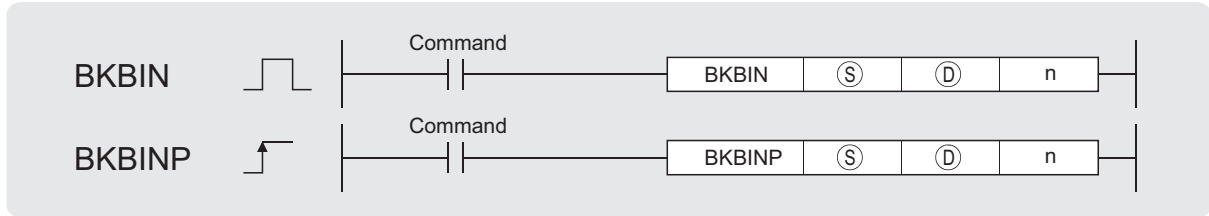
Step	Instruction	Device
0	LD	X20
1	BKBCDP	D100 D200 D0
5	END	

[Operation]



# 6.3.15 Conversion from block BCD 4-digit data to block BIN 16-bit data (BKBIN(P))

Basic High performance Process Redundant Universal

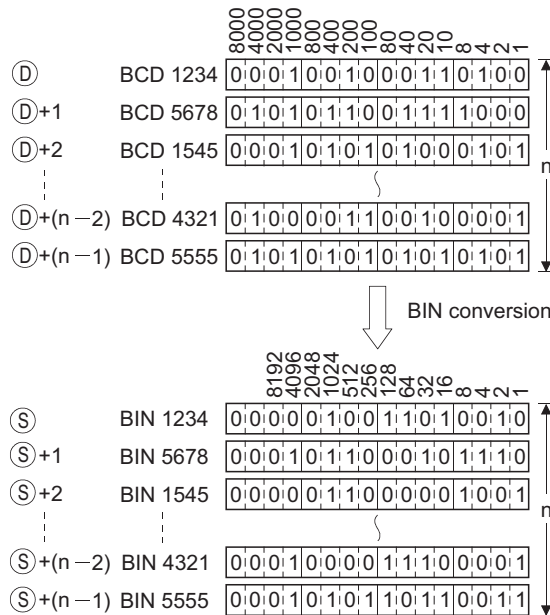


Ⓢ : Head number of the devices where BCD data is stored (BCD 4 digits)  
 Ⓣ : Head number of the devices where the converted BIN data will be stored (BIN 16 bits)  
 n : Number of variable data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
Ⓣ	—	○				—			—
n	○	○				○			—

## ★ Function

- Converts BCD data (0 to 9999) n points from device designated by Ⓢ to BIN, and stores result following the device designated by Ⓣ.



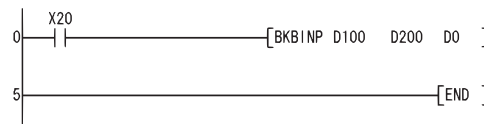
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The n-bit range from the (S), (D), or device exceeds the range of that device. (Error code: 4101)
  - The data n points at the (S) device is outside the 0 to 9999 range. (Error code: 4100)
  - The (S) and (D) devices overlap. (Error code: 4101)

## Program Example

- (1) The following program converts, when X20 is turned ON, the BCD data stored at D100 to D102 to BIN and stores the operation result into the area starting from D200.

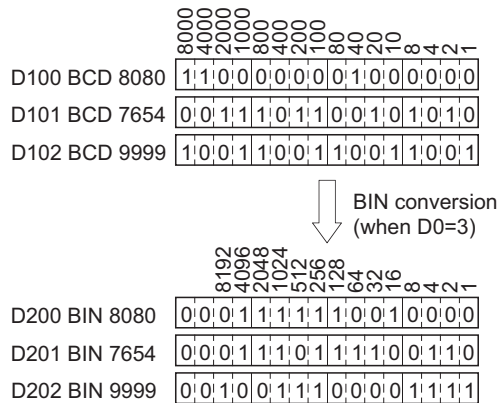
[Ladder Mode]



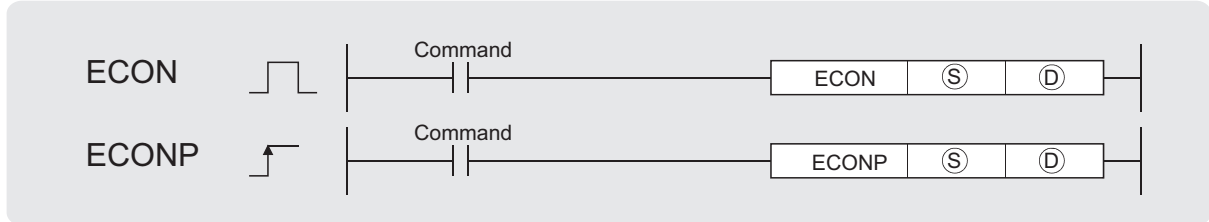
[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKBINP	D100 D200 D0
5	END	

[Operation]



## 6.3.16 Single precision to Double precision conversion (ECON(P))



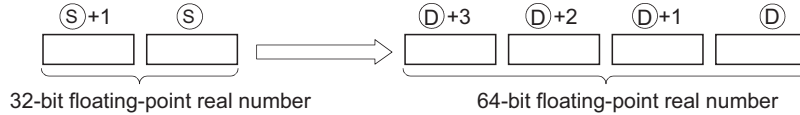
Ⓢ : Conversion source data, or head number of the device where conversion source data is stored (Real number (single precision))

Ⓣ : Head number of the device where the converted data is stored (Real number (double precision))

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

Converts 32-bit floating-point real number specified for Ⓢ into 64-bit floating-point real number, and stores the conversion result to the device specified for Ⓣ.



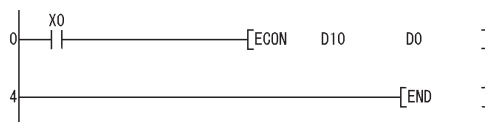
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0, 2^{-126} \leq | \text{value of specified device} | < 2^{128}$
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ . (Error code: 4140)

## Program Example

- (1) The program which converts 32-bit floating-point real number of the devices, D10 to D11, into 64-bit floating-point real number when X0 turns ON, and outputs the conversion result to the devices, D0 to D3.

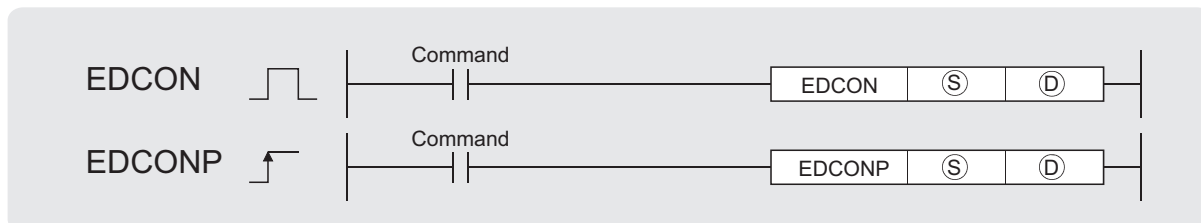
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ECON	D10 D0
4	END	

## 6.3.17 Double precision to Single precision conversion (EDCON(P))



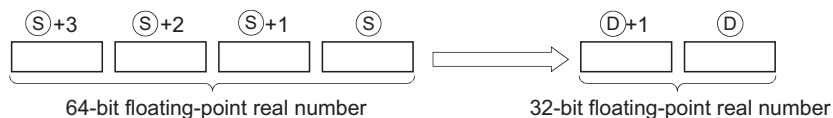
Ⓢ : Conversion source data, or head number of the device where conversion source data is stored (Real number (double precision))

ⓓ : Head number of the device where the converted data is stored (Real number (single precision))

Setting Data	Internal Devices		R, ZR	J00		U0G0	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○			—		—	○	—
ⓓ	—	○			—		○	—	—

### ★ Function

Converts 64-bit floating-point real number specified for Ⓢ into 32-bit floating-point real number, and stores the conversion result to the device specified for ⓓ.



### ! Operation Error

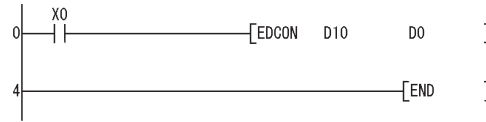
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The result exceeds the following range (Conversion results in an overflow): (Error code: 4141)  
 $2^{128} \leq | \text{Conversion result} |$



## Program Example

- (1) The program which converts 64-bit floating-point real number of the devices, D10 to D13, into 32-bit floating-point real number when X0 turns ON, and outputs the conversion result to the devices, D0 to D1.

[Ladder Mode]



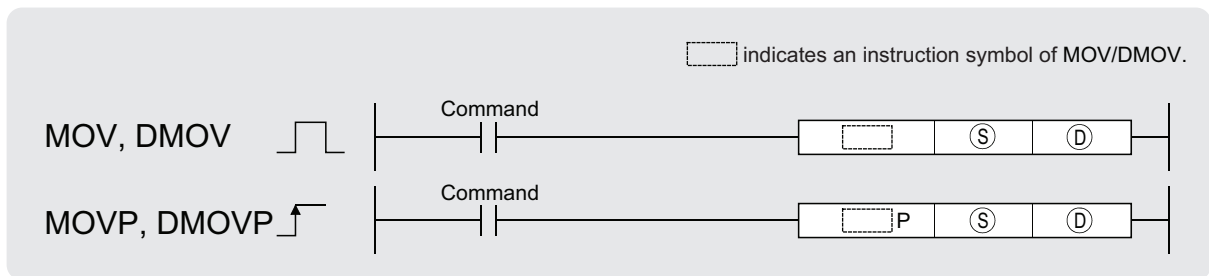
[List Mode]

Step	Instruction	Device
0	LD	X0
1	EDCON	D10 D0
4	END	

## 6.4 Data Transfer Instructions

### 6.4.1 16-bit and 32-bit data transfers (MOV(P),DMOV(P))

Basic High performance Process Redundant Universal



Ⓢ: Data to be transferred or the number of the device where the data to be transferred is stored (BIN 16/32 bits)

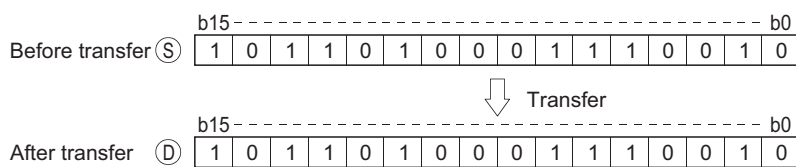
Ⓣ: Number of the device where the data will be transferred (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—

## ★ Function

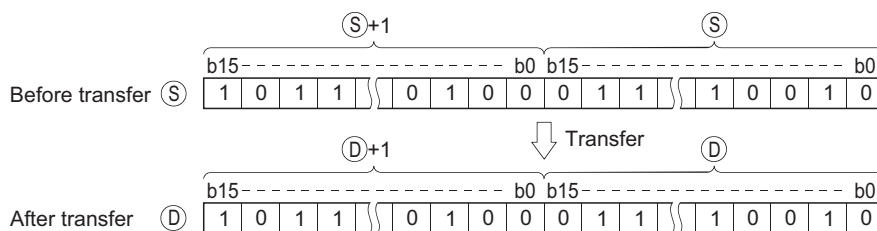
### MOV

- (1) Transfers the 16-bit data from the device designated by Ⓢ to the device designated by Ⓣ.



### DMOV

- (1) Transfers 32-bit data at the device designated by Ⓢ to the device designated by Ⓣ.



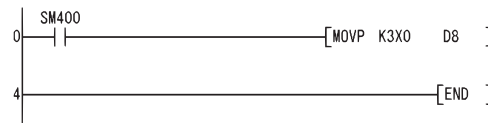
## ! Operation Error

- (1) There are no operation errors associated with the MOV(P) or DMOV(P) instruction.

## Program Example

- (1) The following program stores input data from X0 to XB at D8.

[Ladder Mode]

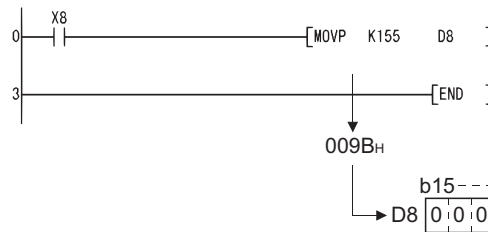


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOV P	K3X0 D8
4	END	

- (2) The following program stores the constant K155 at D8 when X8 goes ON.

[Ladder Mode]

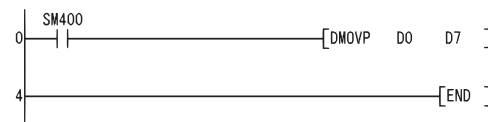


[List Mode]

Step	Instruction	Device
0	LD	X8
1	MOV P	K155 D8
3	END	

- (3) The following program stores the data from D0 and D1 at D7 and D8.

[Ladder Mode]

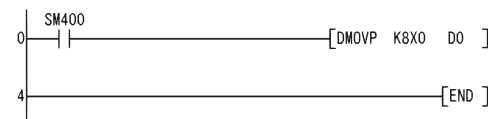


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DMOV P	D0 D7
4	END	

- (4) The following program stores the data from X0 to X1F at D0 and D1.

[Ladder Mode]



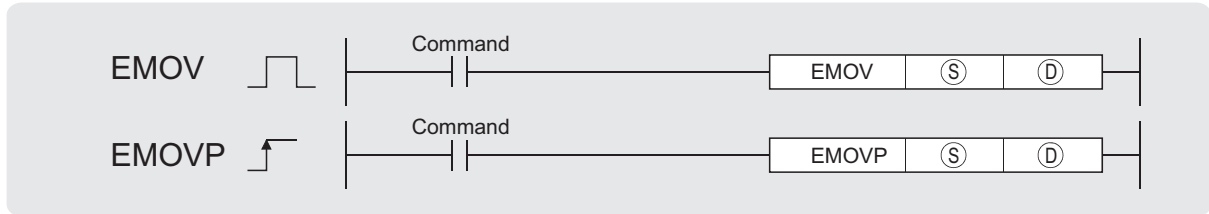
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DMOV P	K8X0 D0
4	END	

## 6.4.2 Floating-point data transfer (Single precision) (EMOV(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Data to be transferred or number of the device to which the data to be transferred is stored (real number)

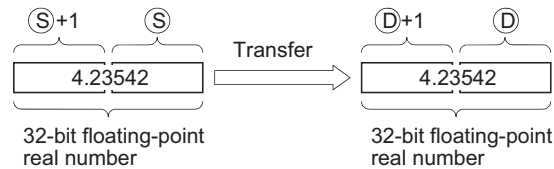
Ⓣ : The number of the device to which the transferred data will be stored (real number)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○		—		○	○*1	○	—
Ⓣ	—	○		—		○	○*1	—	—

\*1: Available only in multiple Universal model QCPU

### ★ Function

Transfers 32-bit floating decimal point type real number data being stored at the device designated by Ⓢ to a device designated by Ⓣ.



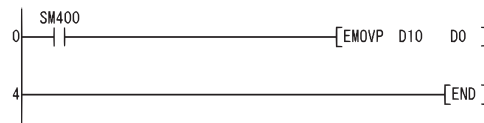
### ! Operation Error

- (1) There are no operation errors associated with the EMOV(P) instruction.

## Program Example

- (1) The following program stores the real numbers at D10 and D11 at D0 and D1.

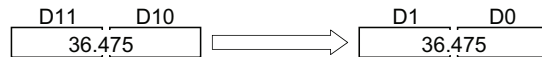
[Ladder Mode]



[List Mode]

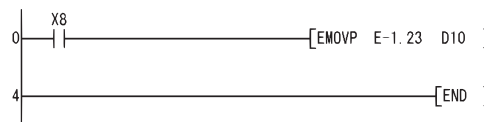
Step	Instruction	Device
0	LD	SM400
1	EMOVP	D10 D11 D0
4	END	

[Operation]



- (2) The following program stores the real number  $-1.23$  at D10 and D11 when X8 is ON.

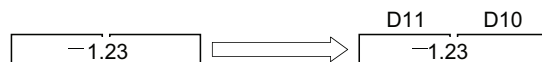
[Ladder Mode]



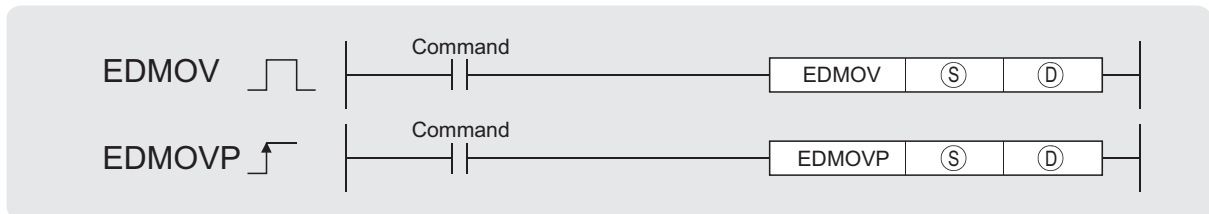
[List Mode]

Step	Instruction	Device
0	LD	X8
1	EMOVP	E-1.23 D10 D11
4	END	

[Operation]



## 6.4.3 Floating-point data transfer (Double precision) (EDMOV(P))



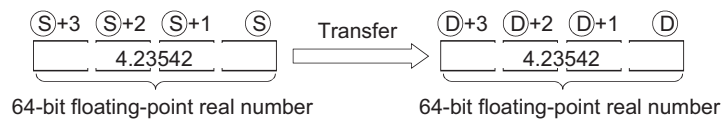
Ⓢ : Data to be transferred or number of the device to which the data to be transferred is stored (real number)

ⓓ : The number of the device to which the transferred data will be stored (real number)

Setting Data	Internal Devices		R, ZR	JGD		UAG	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
ⓓ	—	○				—		—	—

### ★ Function

Transfers 64-bit floating decimal point type real number data being stored at the device designated by Ⓢ to a device designated by ⓓ.



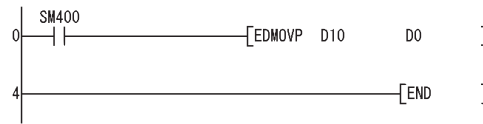
### ! Operation Error

- (1) There are no operation errors associated with the EDMOV(P) instruction.

## Program Example

- (1) The following program stores the 64-bit floating decimal point type real number at D10 to D13 at D0 to D3.

[Ladder Mode]



[List Mode]

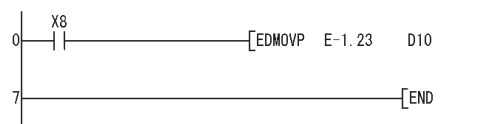
Step	Instruction	Device
0	LD	SM400
1	EDMOV P	D10 D0
4	END	

[Operation]



- (2) The following program stores the real number  $-1.23$  at D10 to D13 when X8 is ON.

[Ladder Mode]



[List Mode]

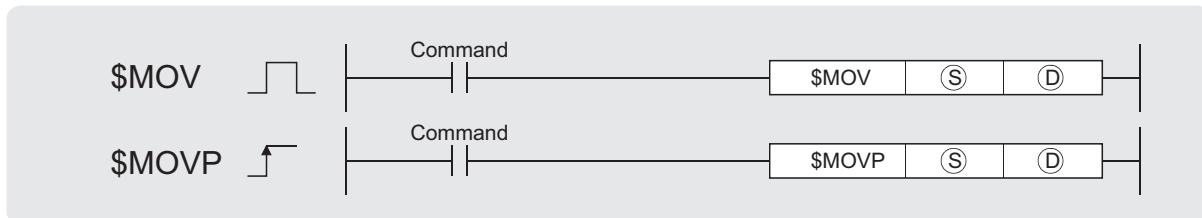
Step	Instruction	Device
0	LD	X8
1	EDMOV P	E-1.23 D10
7	END	

[Operation]



## 6.4.4 Character string transfers (\$MOV(P))

Basic High performance Process Redundant Universal



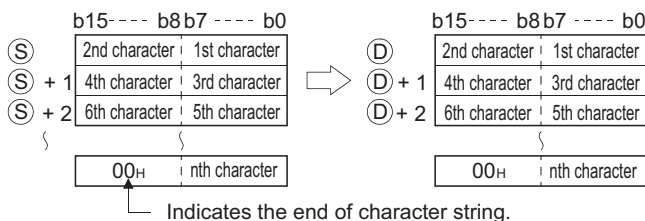
Ⓢ : Character string to be transferred (maximum string length: 32 characters) or head number of the devices where the character string to be transferred is stored (character string)

Ⓣ : Head number of the devices where the transferred character string will be stored (character string)

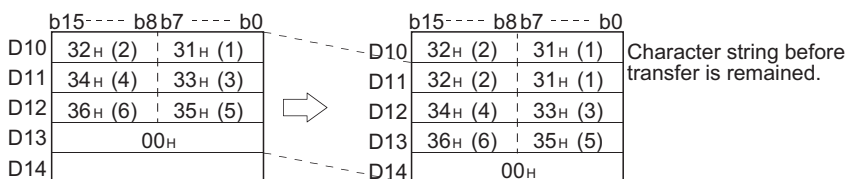
Setting Data	Internal Devices		R, ZR	JDO		U <sub>0</sub> G <sub>0</sub>	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- Transfers the character string data designated by Ⓢ to the devices from the device designated by Ⓣ and onward. The character string data enclosed in " (double quotes) or devices from the number specified by Ⓢ to the device number storing "00<sub>H</sub>" are transferred all at once.

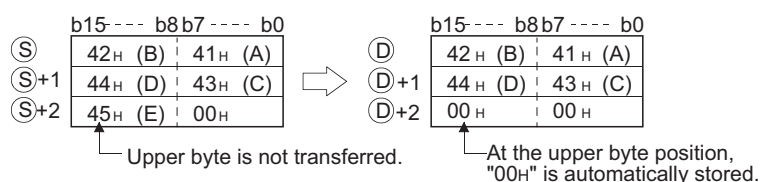


- Processing will be performed without error even in cases where the range for the devices storing the character data to be transferred (Ⓢ to Ⓢ+n) overlaps with the range of the devices which will store the character string data after it has been transferred (Ⓣ to Ⓣ+n). The following occurs when the character string data that had been stored from D10 to D13 is transferred to D11 to D14:





- (3) If the "00H" code is being stored at lower bytes of  $\textcircled{S} + n$ , "00H" will be stored at both the higher bytes and the lower bytes of  $\textcircled{D} + n$ .



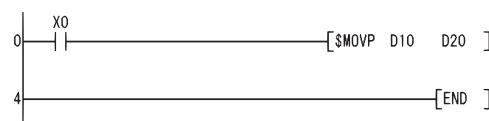
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- There is no "00H" code stored between the device number designated by  $\textcircled{S}$  and the relevant device. (Error code: 4101)
  - The entire character string linked from the device number designated by  $\textcircled{D}$  to the final device number of the relevant device cannot be stored. (Error code: 4101)
  - The character string of  $\textcircled{S}$  exceeds 16383 characters. (Error code: 4101)

## Program Example

- (1) The character string data stored in D10 to D12 is transferred to D20 to D22 when X0 goes ON.

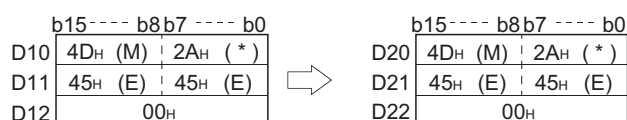
[Ladder Mode]



[List Mode]

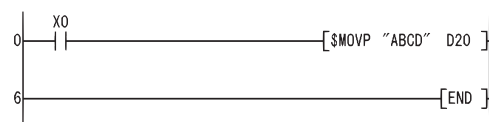
Step	Instruction	Device
0	LD	X0
1	\$MOV P	D10 D20
4	END	

[Operation]



- (2) When X is turned ON, the character string "ABCD" is transferred to D20 and D21.

[Ladder Mode]

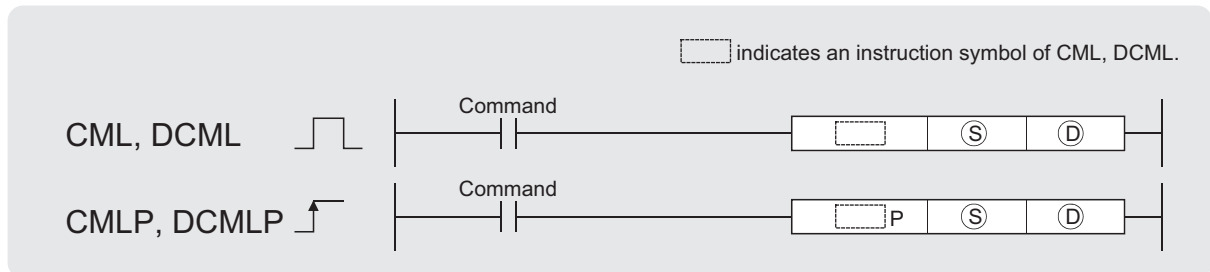


[List Mode]

Step	Instruction	Device
0	LD	X0
1	\$MOV P	"ABCD" D20
6	END	

## 6.4.5 16-bit and 32-bit negation transfers (CML(P),DCML(P))

Basic High performance Process Redundant Universal



Ⓢ : Data to be reversed or the number of the device where data to be reversed is stored (BIN 16/32 bits)

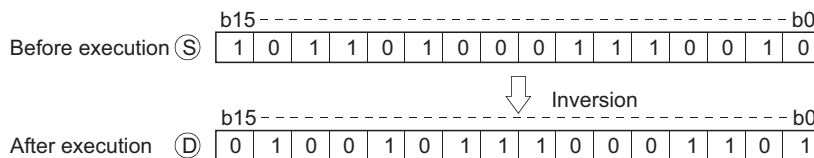
Ⓣ : Number of the device where the reversing result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

### ★ Function

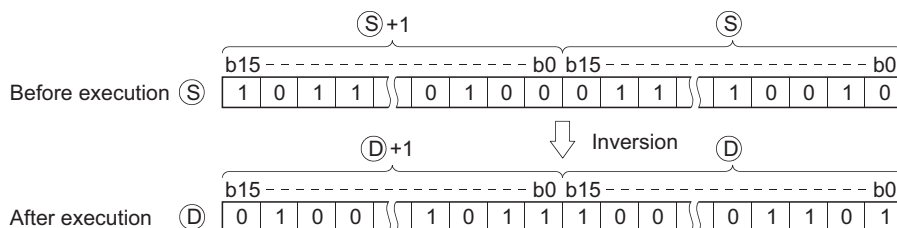
#### CML

- (1) Inverts 16-bit data designated by Ⓢ bit by bit, and transfers the result to the device designated by Ⓣ.



#### DCML

- (1) Inverts 32-bit data designated by Ⓢ bit by bit, and transfers the result to the device designated by Ⓣ.



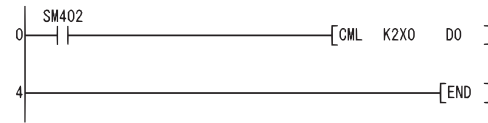
### ! Operation Error

- (1) There are no operation errors associated with the CML(P) or DCML(P) instruction.

## Program Example

- (1) The following program inverts the data from X0 to X7, and transfers result to D0.

[Ladder Mode]

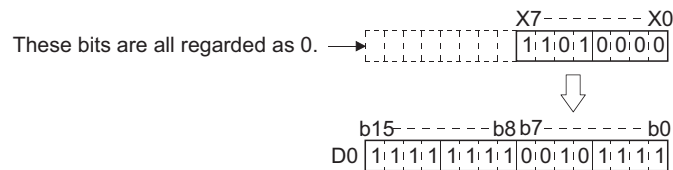


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	CML	K2X0 D0
4	END	

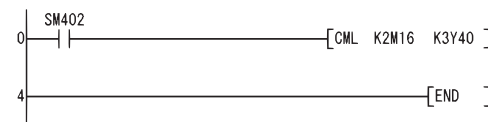
[Operation]

If "Number of bits of (S) < Number of bits of (D)"



- (2) The following program inverts the data at M16 to M23, and transfers the result to Y40 to Y47.

[Ladder Mode]

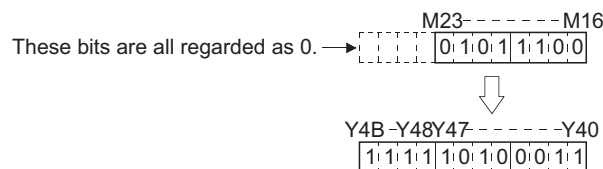


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	CML	K2M16 K3Y40
4	END	

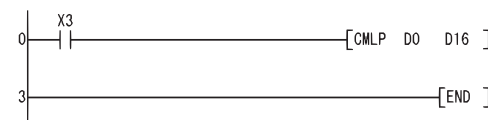
[Operation]

If "Number of bits of (S) < Number of bits of (D)"



- (3) The following program inverts the data at D0 when X3 is ON, and stores the result at D16.

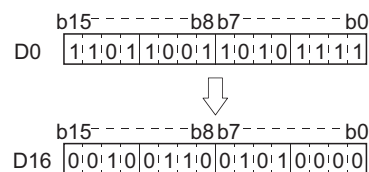
[Ladder Mode]



[List Mode]

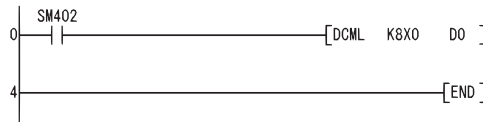
Step	Instruction	Device
0	LD	X3
1	CMLP	D0 D16
3	END	

[Operation]



(4) The following program inverts the data at X0 to X1F, and transfers results to D0 and D1.

[Ladder Mode]

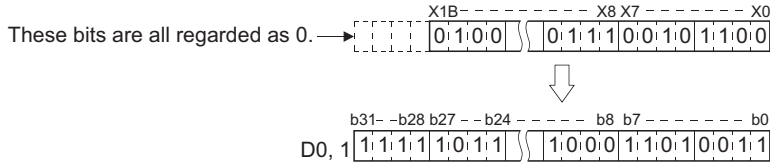


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	DCML	K8X0 D0
4	END	

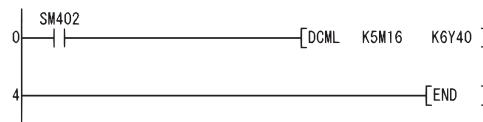
[Operation]

If "Number of bits of (S) < Number of bits of (D)"



(5) The following program inverts the data at M16 to M35, and transfers it to Y40 to Y63.

[Ladder Mode]

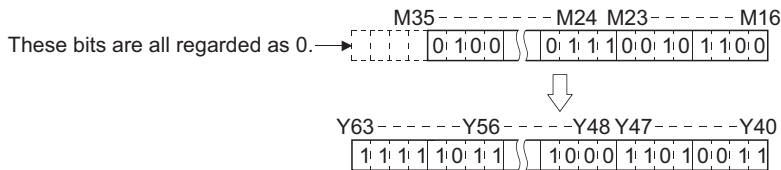


[List Mode]

Step	Instruction	Device
0	LD	SM402
1	DCML	K5M16 K6Y40
4	END	

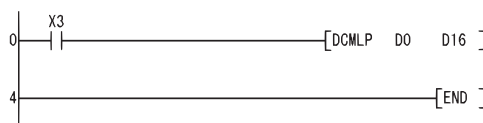
[Operation]

If "Number of bits of (S) < Number of bits of (D)"



(6) Inverts the data at D0 and D1 when X3 is ON, and stores the result at D16 and D17.

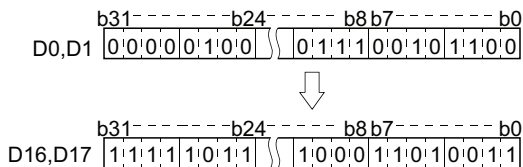
[Ladder Mode]



[List Mode]

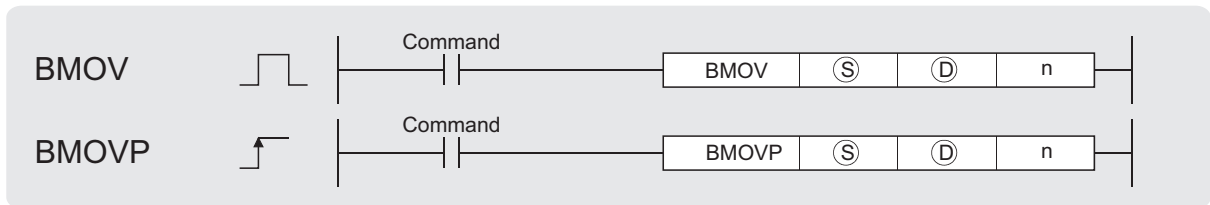
Step	Instruction	Device
0	LD	X3
1	DCMLP	D0 D16
4	END	

[Operation]



## 6.4.6 Block 16-bit data transfers (BMOV(P))

Basic High performance Process Redundant Universal

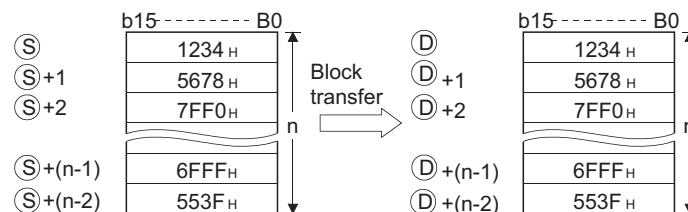


- Ⓢ : Head number of the devices where the data to be transferred is stored (BIN 16 bits)
- Ⓣ : Head number of the devices of transfer destination (BIN 16 bits)
- n : Number of transfers (when using an intelligent function module device (U<sub>IG</sub>): 1 to 6144 (QnA only)) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J <sub>IG</sub>		U <sub>IG</sub>	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ							—		—
Ⓣ							—		—
n							○		—

### ★ Function

- (1) Transfers in batch 16-bit data of n points from the device designated by Ⓢ to location n points from the device designated by Ⓣ.



- (2) Transfers can be accomplished even in cases where there is an overlap between the source and destination device.  
 In the case of transmission to the smaller device number, transmission is from Ⓢ; for transmission to the larger device number, transmission is from Ⓢ + (n - 1).  
 However, as shown in the example below, when transferring data from R to ZR, or from ZR to R, the range to be transferred (source) and the range of destination must not overlap.  
 Transfer from R to R, or from ZR to ZR can be performed without any problem.

- ZR transfer range ((specified head No. of ZR) to (specified head No. of ZR + the number of transfers - 1))
- R transfer range ((specified head No. of R + file register block No. × 32768) to (specified head No. of R + file register block No. × 32768 + the number of transfers - 1))

6

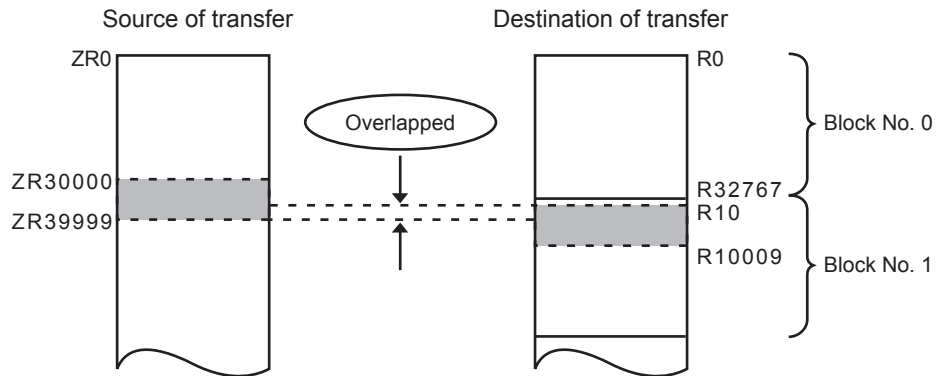
6.4 Data Transfer Instructions  
6.4.6 Block 16-bit data transfers (BMOV(P))

**Example**

Transfer ranges of ZR and R overlap when transferring 10000 blocks of data from ZR30000 (source) to R10 (block No.1 of the destination).

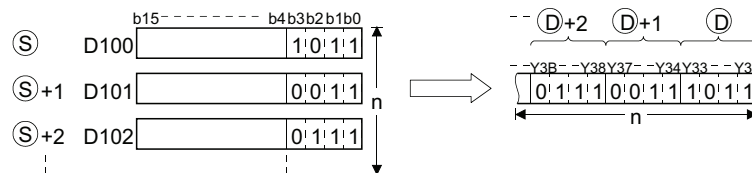
- ZR transfer range → (30000) to (30000+10000-1) → (30000) to (39999)
- R transfer range → (10+(1 × 32768)) to (10+(1 × 32768)+10000-1)  
→ (32778) to (42777)

Therefore, the range 32778 to 39999 overlaps and the data is not correctly transferred.



- (3) When ⑤ is a word device and ④ is a bit device, the object for the word device will be the number of bits designated by the bit device digit designation.

If K1Y30 has been designated by ④, the lower four bits of the word device designated by ⑤ will become the object.



- (4) If bit device has been designated for ⑤ and ④, then ⑤ and ④ should always have the same number of digits.
- (5) When using a link direct device and an intelligent function module device for ⑤ and ④, only either of ⑤ or ④ can be used.
- (6) Selection whether to check a device range

Whether to check a device range during execution of the BMOV instruction can be selected with the device range check inhibit flag (SM237) (only when the conditions for subset processing are established).

While SM237 is ON, whether ⑤ to ⑤ + (n) - 1 and ④ to ④ + (n) - 1 are within the device range or not are not checked.

For details of SM237, refer to Appendix 3 SPECIAL RELAY LIST.

**POINT**

SM237 can be used only for the Universal model QCPU whose first 5 digits of serial number is 10012 or later.

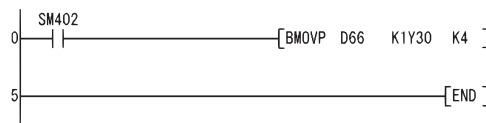
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device range of n points from ③ or ④ exceeds the corresponding device range.  
(Error code: 4101)

## Program Example

- (1) The following program outputs the lower 4 bits of data at D66 to D69 to Y30 to Y3F in 4-point units.

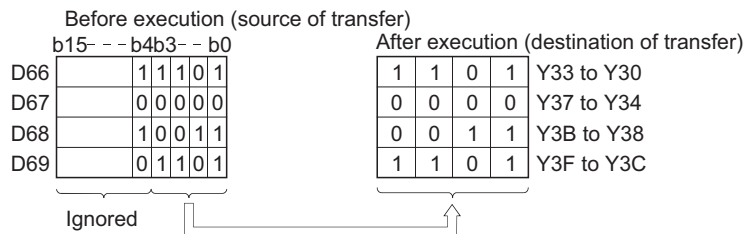
[Ladder Mode]



[List Mode]

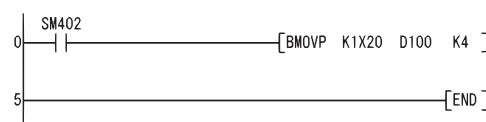
Step	Instruction	Device
0	LD	SM402
1	BMOV	D66 K1Y30 K4
5	END	

[Operation]



- (2) The following program outputs the data at X20 to X2F to D100 to D103 in 4-point units.

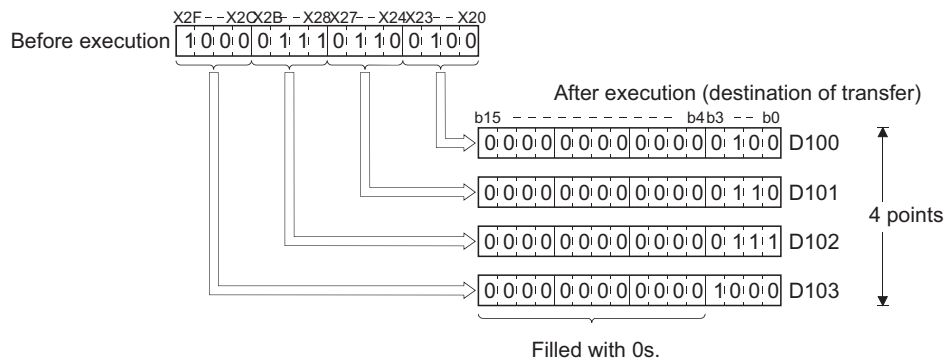
[Ladder Mode]



[List Mode]

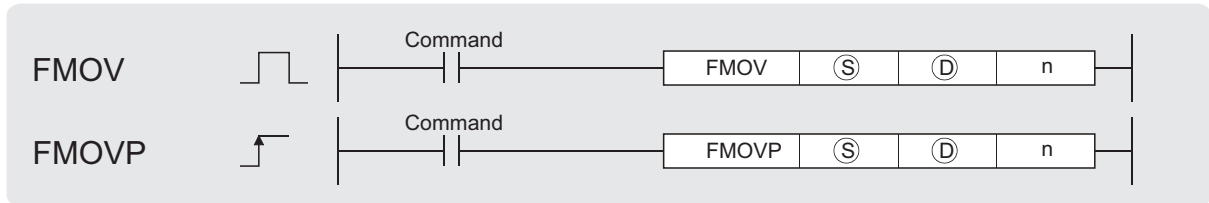
Step	Instruction	Device
0	LD	SM402
1	BMOV	K1X20 D100 K4
5	END	

[Operation]



## 6.4.7 Identical 16-bit data block transfers (FMOV(P))

Basic High performance Process Redundant Universal



Ⓢ : Data to be transferred or the head number of the devices where the data to be transferred is stored (BIN 16 bits)

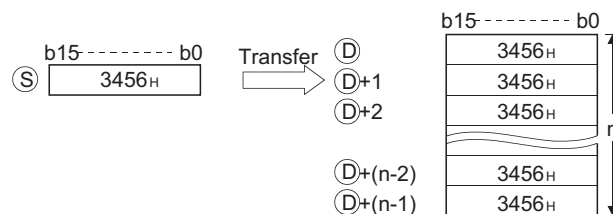
Ⓣ : Head number of the devices of transfer destination (BIN 16 bits)

n : Number of transfers (BIN 16 bits)

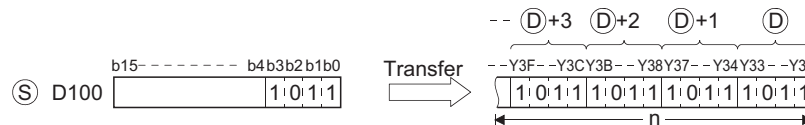
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ					○			○	—
Ⓣ					○			—	—
n					○			○	—

### ★ Function

- Transfers 16-bit data at the device designated by Ⓢ to n points of devices starting from the one designated by Ⓣ.



- In cases where Ⓢ designates a word device and Ⓣ a bit device, the number of bits designated by digit designation for the bit device will be the object bits for the word device. Ⓢ If K1Y30 has been designated by Ⓣ, the lower 4 bits of the word device designated by Ⓢ will become the object.



- If bit device has been designated for Ⓢ and Ⓣ, then Ⓢ and Ⓣ should always have the same number of digits.



- (4) Selection whether to check a device range  
 Whether to check a device range during execution of the FMOV instruction can be selected with the device range check inhibit flag (SM237) (only when the conditions for subset processing are established).

While SM237 is ON, whether  $\text{D}$  to  $\text{D} + (n) - 1$  is within the device range or not is not checked.

For details of SM237, refer to Appendix 3 SPECIAL RELAY LIST.

**POINT**

SM237 can be used only for the Universal model QCPU whose first 5 digits of serial number is 10012 or later.

**Operation Error**

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device range of n points from  $\text{D}$  or exceeds the corresponding device range. (Error code: 4101)

**Program Example**

- (1) The following program outputs the lower 4 bits of D0 when XA goes ON to Y10 to Y23 in 4-bit units.

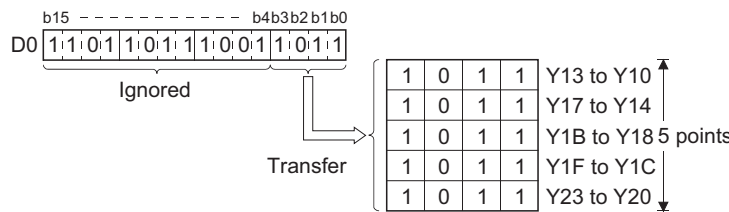
[Ladder Mode]



[List Mode]

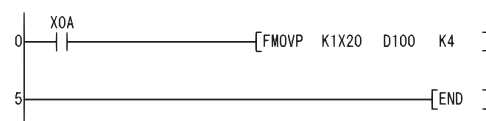
Step	Instruction	Device
0	LD	XOA
1	FMOV P	D0 K1Y10 K5
5	END	

[Operation]



- (2) The following program outputs the data at X20 through X23 to D100 through D103 when XA goes ON.

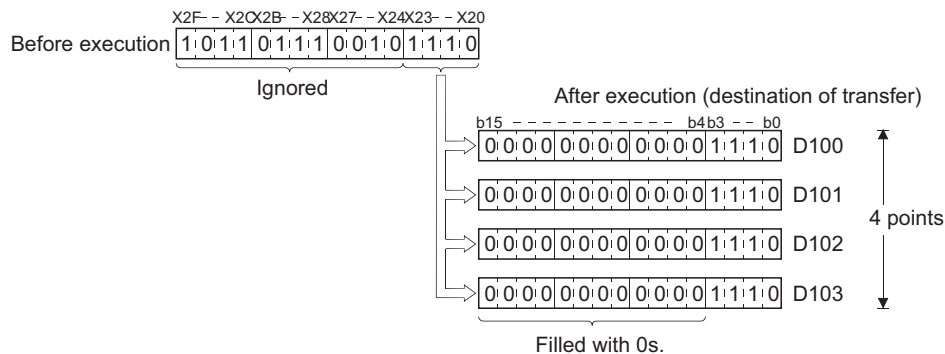
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	XOA
1	FMOV P	K1X20 D100 K4
5	END	

[Operation]

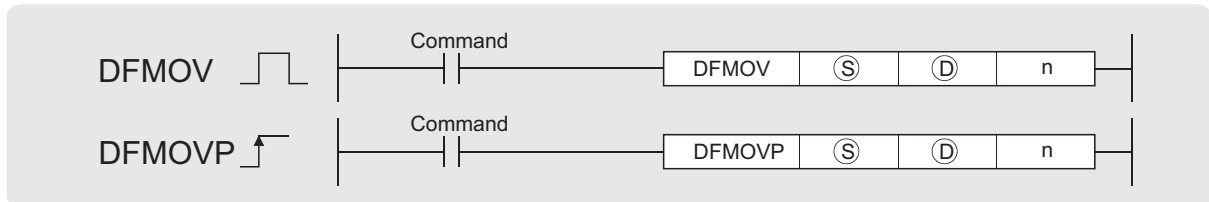


## 6.4.8 Identical 32-bit data block transfers (DFMOV(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ : Data to be transferred or head number of the devices where the data to be transferred are stored (BIN 32 bits)

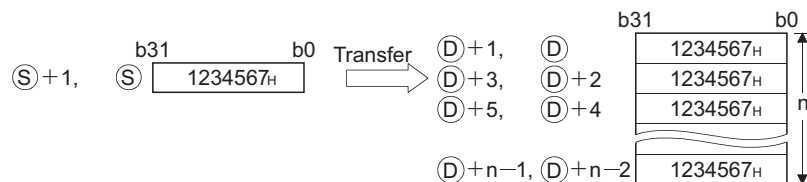
Ⓣ : Head number of the devices of transfer destination (BIN 32 bits)

n : Number of transfers (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	Jm		UAGo	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ									—
Ⓣ									—
n									—

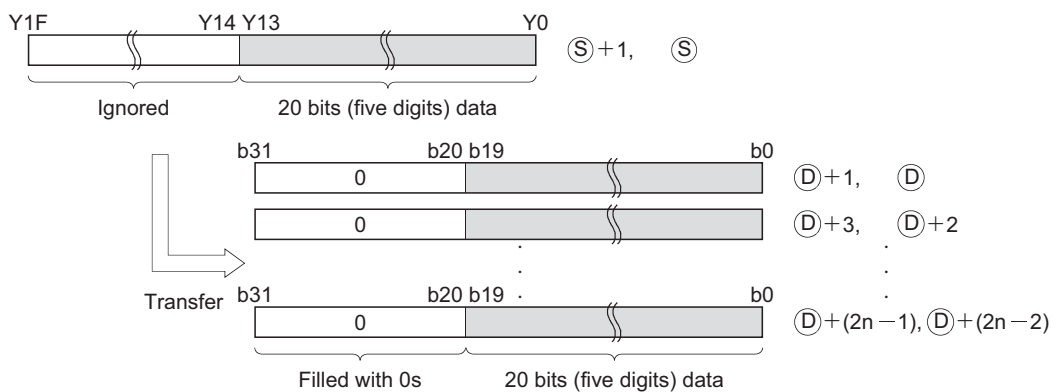
### ★ Function

- (1) This instruction transfers 32-bit data of the device specified by Ⓢ to the n-point devices starting from the device specified by Ⓣ.

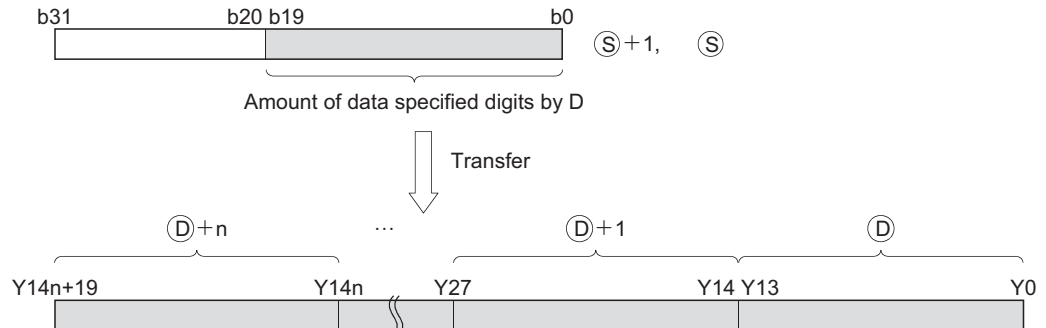


- (2) If Ⓢ specifies data of a device with digit specification, the amount of data to be transferred will be the amount of the data specified digit.

If K5Y0 is specified by Ⓢ, the lower 20 bits (five digits) of the word device specified by Ⓢ will be the object.



- (3) If  $\textcircled{D}$  specifies data of a device with digit specification, the amount of data stored in the device specified by  $\textcircled{D}$  will be transferred.
- If K5Y0 is specified by  $\textcircled{D}$ , the lower 20 bits of the word device specified by  $\textcircled{S}$  will be the object.
- If both  $\textcircled{S}$  and  $\textcircled{D}$  specify data of a device with digit specification, the amount of data specified by  $\textcircled{D}$  will be transferred regardless of the number of digits.



- (4) If the value specified by n is 0, the instruction will be not processed.
- (5) Whether to check a device range during the execution of the FMOV instruction can be selected with the device range check inhibit flag (SM237). (Only when the conditions of the subset processing are established)

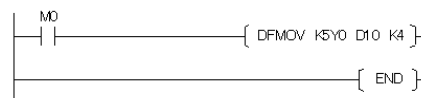
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The value specified by n is negative. (Error code: 4100)
  - The range of n-point devices, to be transferred, exceeds the range of devices specified by  $\textcircled{D}$ . (Error code: 4101)

## Program Example

- (1) The following program stores the value data stored at Y0 to Y13(20 bits) into D10 to D17, when M0 is turned on,

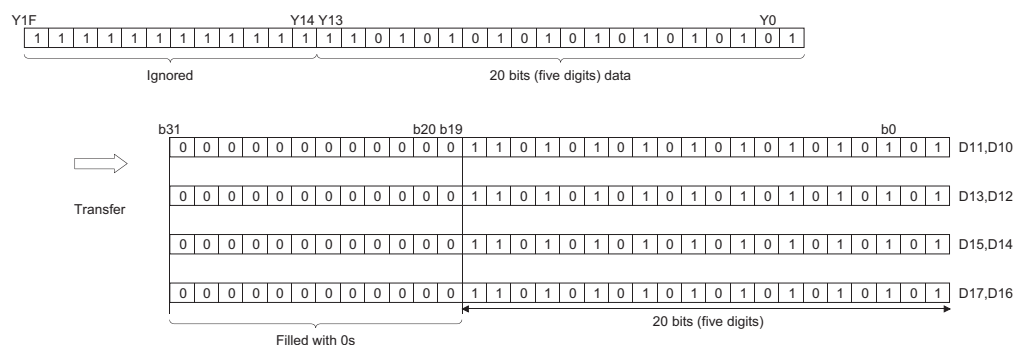
[Ladder Mode]



[List Mode]

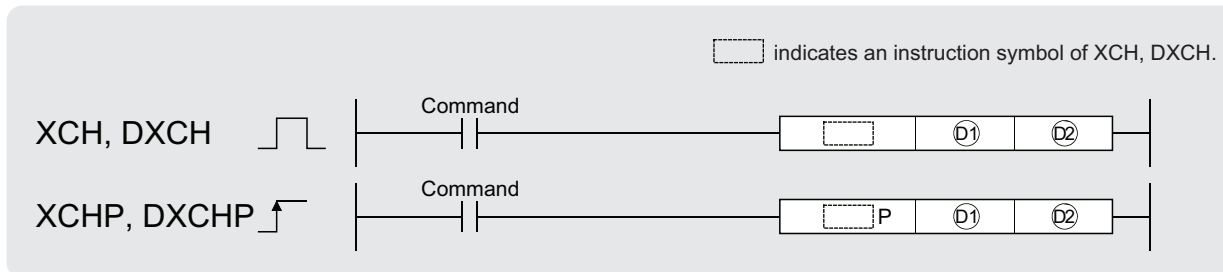
Step	Instruction	Device
0	LD	M0
1	DFMOV	K5Y0 D10 K4
5	END	

[Operation]



## 6.4.9 16-bit and 32-bit data exchanges (XCH(P),DXCH(P))

Basic High performance Process Redundant Universal



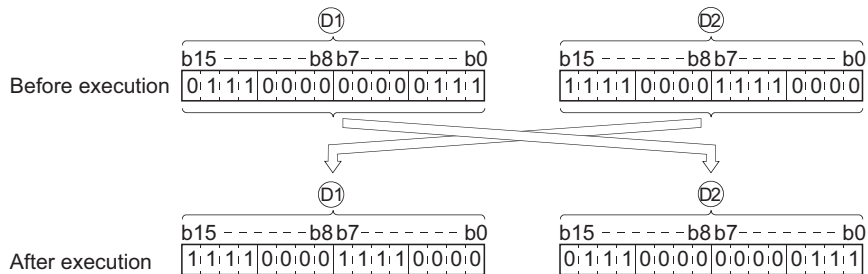
Ⓛ<sub>1</sub>, Ⓛ<sub>2</sub> : Head number of the devices where the data to be exchanged is stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓛ <sub>1</sub>					○				—
Ⓛ <sub>2</sub>					○				—

### ★ Function

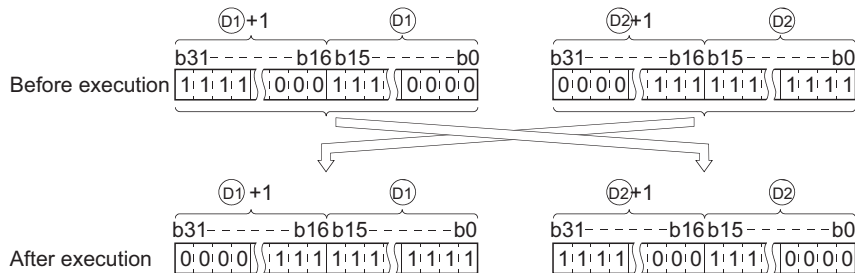
#### XCH

(1) Conducts 16-bit data exchange between Ⓛ<sub>1</sub> and Ⓛ<sub>2</sub>.



#### DXCH

(1) Conducts 32-bit data exchange between Ⓛ<sub>1</sub>+1, Ⓛ<sub>1</sub> and Ⓛ<sub>2</sub>+1, Ⓛ<sub>2</sub>.



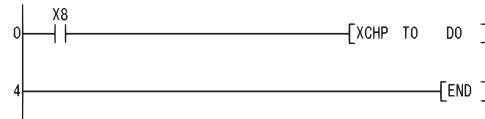
## ! Operation Error

- (1) There are no errors associated with the XCH (P) and DXCH (P) instruction.

## Program Example

- (1) The following program exchanges the present value of T0 with the contents of D0 when X8 goes ON.

[Ladder Mode]

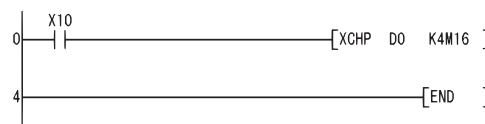


[List Mode]

Step	Instruction	Device
0	LD	X8
1	XCHP	T0 D0
4	END	

- (2) The following program exchanges the contents of D0 with the data from M16 to M31 when X10 goes ON.

[Ladder Mode]

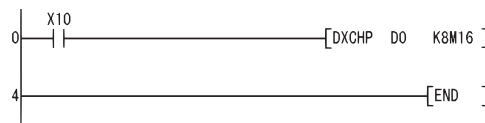


[List Mode]

Step	Instruction	Device
0	LD	X10
1	XCHP	D0 K4M16
4	END	

- (3) The following program exchanges the contents of D0 and D1 with the data at M16 to M47 when X10 goes ON.

[Ladder Mode]

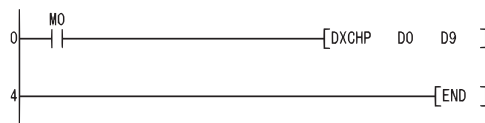


[List Mode]

Step	Instruction	Device
0	LD	X10
1	DXCHP	D0 K8M16
4	END	

- (4) The following program exchanges the contents of D0 and D1 with those of D9 and D10 when M0 goes ON.

[Ladder Mode]

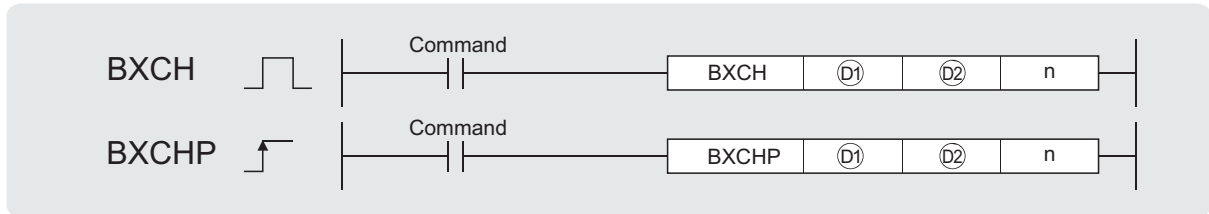


[List Mode]

Step	Instruction	Device
0	LD	M0
1	DXCHP	D0 D9
4	END	

# 6.4.10 Block 16-bit data exchanges (BXCH(P))

Basic High performance Process Redundant Universal

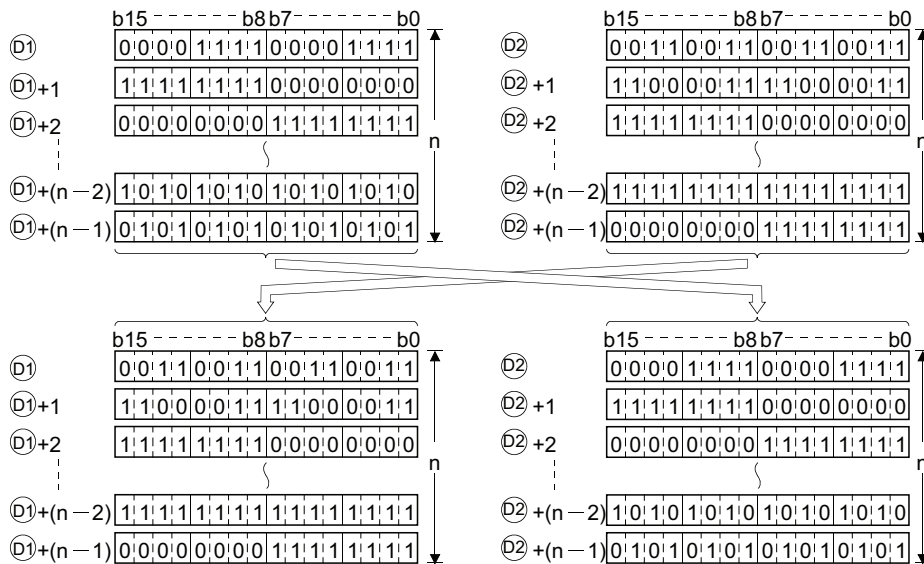


Ⓛ1, Ⓛ2 : Head number of the devices where the data to be exchanged is stored (BIN 16 bits)  
 n : Number of exchanges (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓛ1	—	○				—			—
Ⓛ2	—	○				—			—
n	○	○				○			—

## ★ Function

- (1) Exchanges 16-bit data of n points from device designated by Ⓛ1 and 16-bit data of n points from device designated by Ⓛ2.



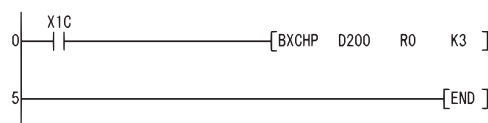
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range of the device of n points from a device designated by (D1), (D2) or exceeds the relevant device. (Error code: 4101)
  - The (D1) and (D2) devices overlap. (Error code: 4101)

## Program Example

- (1) The following program exchanges 16-bit data for 3 points from D200 for 16-bit data for 3 points from R0 when X1C goes ON.

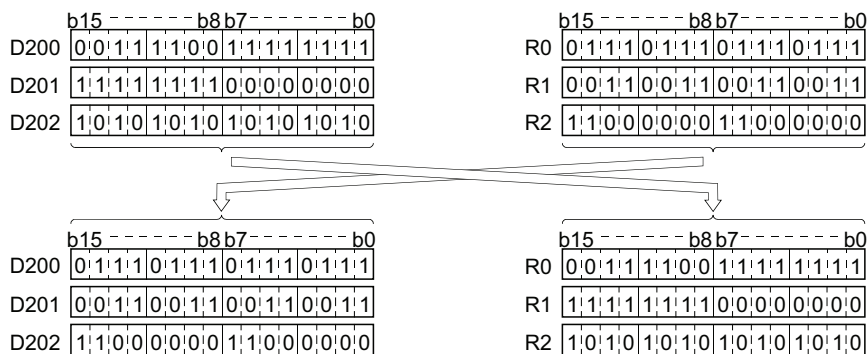
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	BXCHP	D200 R0 K3
5	END	

[Operation]



## 6.4.11 Upper and lower byte exchanges (SWAP(P))

Basic High performance Process Redundant Universal

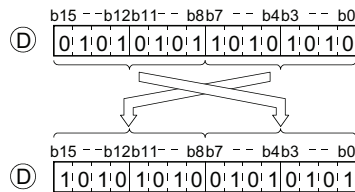


Ⓣ : Head number of the devices where the data is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓣ									—

### ★ Function

- (1) Exchanges the higher and lower 8 bits of the device designated by Ⓣ.



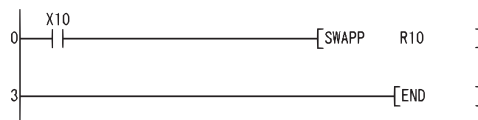
### ! Operation Error

- (1) There are no operation errors associated with the SWAP(P) instruction.

### 📄 Program Example

- (1) The following program exchanges the higher 8 bits and lower 8 bits of R10 when X10 goes ON.

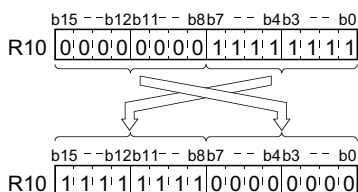
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	SWAPP	R10
3	END	

[Operation]

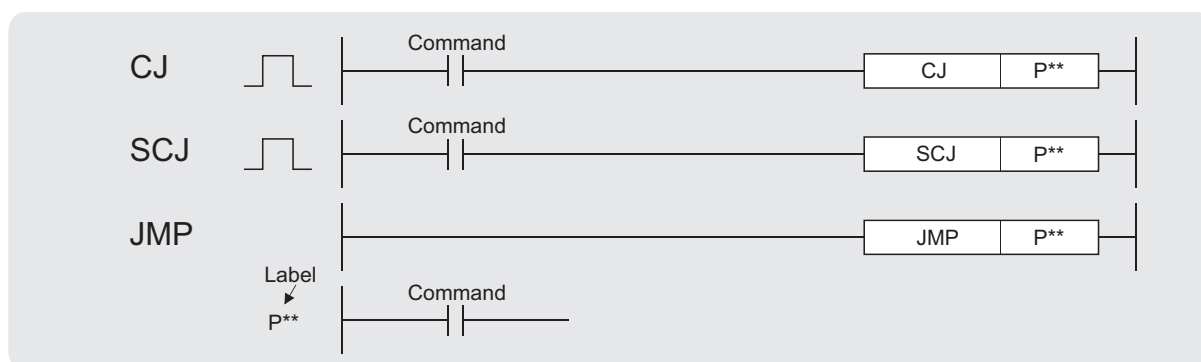




## 6.5 Program Branch Instructions

### 6.5.1 Pointer branch instructions (CJ,SCJ,JMP)

Basic High performance Process Redundant Universal



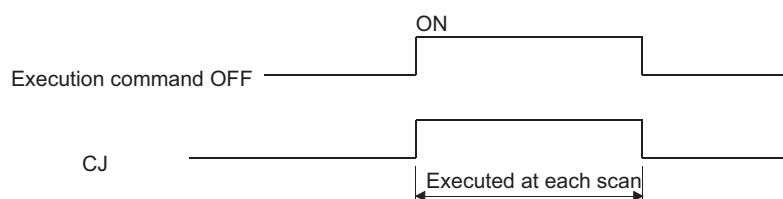
P\*\* : Pointer number of jump destination (Device name)

Setting Data	Internal Devices		R, ZR	JMP		U:G	Zn	Constants	Other P
	Bit	Word		Bit	Word				
P									○

### ★ Function

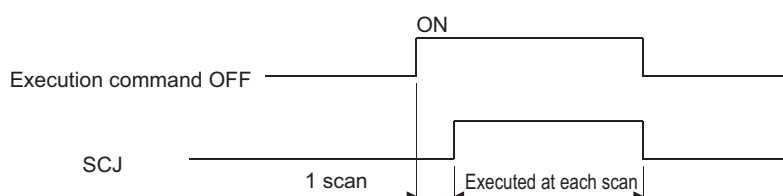
#### CJ

- Executes the program specified by the pointer number within the same program file, when the execution command is ON.
- When the execution command is OFF, the program at the next step is executed.



#### SCJ

- Executes the program specified by the pointer number within the same program file starting with the scan immediately after OFF→ON of the execution command.
- When the execution command is OFF or turned ON→OFF, the program at the next step is executed.



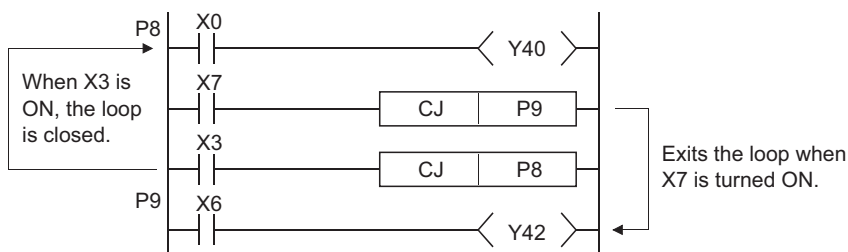
## JMP

- (1) Unconditionally executes program of designated pointer number within the same program file.

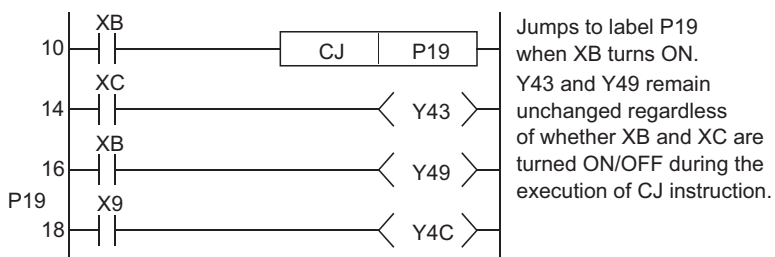
### POINT

Note the following points when using the jump instruction.

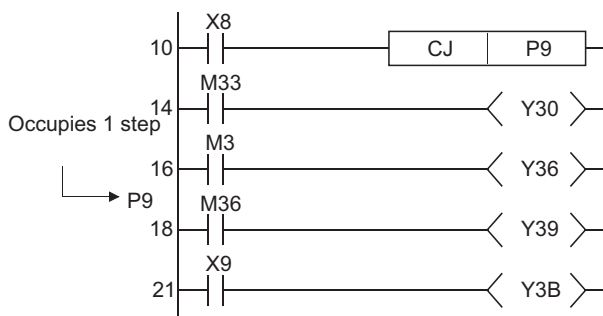
1. After the timer coil has gone ON, accurate measurements cannot be made if there is an attempt to jump the timer of a coil that has been turned ON using the CJ, SCJ or JMP instructions.
2. Scan time is shortened if the CJ, SCJ or JMP instruction is used to force a jump to the OUT instruction.
3. Scan time is shortened if the CJ, SCJ or JMP instruction is used to force a jump to the rear.
4. The CJ, SCJ, and JMP instructions can be used to jump to a step prior to the step currently being executed. However, it is necessary to consider methods to get out of the loop so that the watchdog timer does not time out in the process.



5. The device to which a jump has been made with the CJ, SCJ or JMP does not change.



6. The label (P\*) occupies step 1.



7. The jump instructions can be used only for pointer numbers within the same program file.
8. If a jump is made to a pointer number inside the skip range during a skip operation, program execution will be taken up following the pointer number of the jump destination.

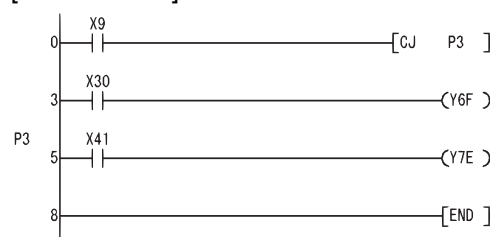
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The pointer number designated does not come prior to the END instruction. (Error code: 4210)
  - A pointer number which is not in use as a label in the same program has been designated. (Error code: 4210)
  - A common pointer has been designated. (Error code: 4210)

## Program Example

- (1) The following program jumps to P3 when X9 goes ON.

[Ladder Mode]

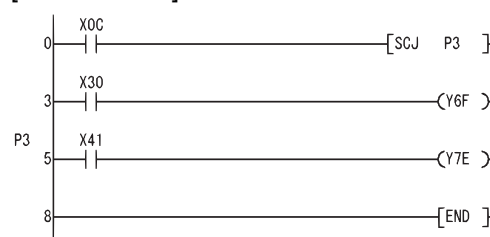


[List Mode]

Step	Instruction	Device
0	LD	X9
1	CJ	P3
3	LD	X30
4	OUT	Y6F
5	P3	
6	LD	X41
7	OUT	Y7E
8	END	

- (2) The following program jumps to P3 from the next scan after XC goes ON.

[Ladder Mode]



[List Mode]

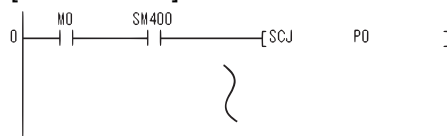
Step	Instruction	Device
0	LD	X0C
1	SCJ	P3
3	LD	X30
4	OUT	Y6F
5	P3	
6	LD	X41
7	OUT	Y7E
8	END	

## ! Caution

- (1) When using the Universal model QCPU with the SCJ instruction, inserting "AND SM400" (or the NOP instruction) in immediately before the SCJ instruction is required.

[Program example1]

[Ladder Mode]

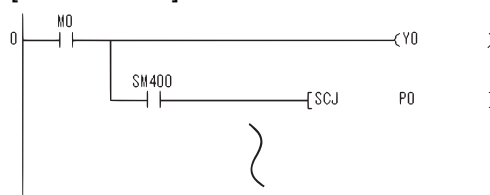


[List Mode]

Step	Instruction	Device
0	LD	M0
1	AND	SM400
2	SCJ	P0

[Program example2]

[Ladder Mode]

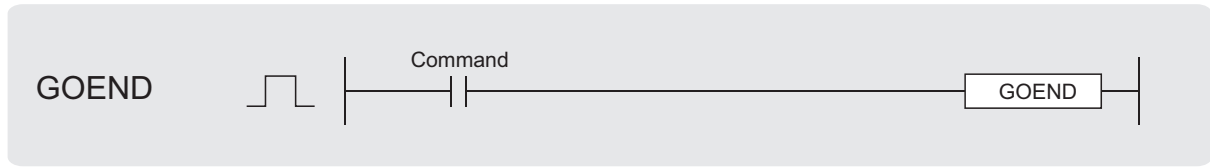


[List Mode]

Step	Instruction	Device
0	LD	M0
1	OUT	Y0
2	AND	SM400
3	SCJ	P0

## 6.5.2 Jump to END (GOEND)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

- (1) Jumps to the FEND or END instruction in the same program file.

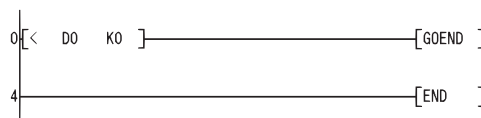
### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The GOEND instruction has been executed after the execution of the CALL, ECALL instruction, and prior to the execution of the RET instruction. (Error code: 4211)
  - The GOEND instruction has been executed after the execution of the FOR instruction, and prior to the execution of the NEXT instruction. (Error code: 4200)
  - The GOEND instruction has been executed during an interrupt program but prior to the execution of the IRET instruction. (Error code: 4221)
  - The GOEND instruction was executed between the CHKCIR and CHKEND instruction block. (Error code: 4230)
  - The GOEND instruction was executed between the IX and IXEND instruction block. (Error code: 4231)

### 📄 Program Example

- (1) The following program jumps to the END instruction if D0 holds a negative number.

[Ladder Mode]



[List Mode]

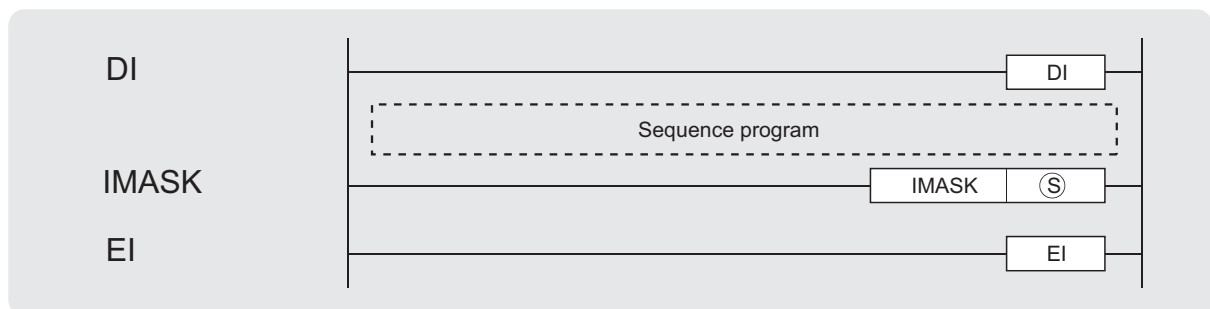
Step	Instruction	Device
0	LD<	D0 KO
3	GOEND	
4	END	

## 6.6 Program Execution Control Instructions

### 6.6.1 Interrupt disable/enable instructions, interrupt program mask (DI,EI,IMASK)

Basic High performance Process Redundant Universal

① When the Basic model QCPU is used



Ⓢ : Interrupt mask data or head number of the devices where the interrupt mask data is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○					—		

#### ★ Function

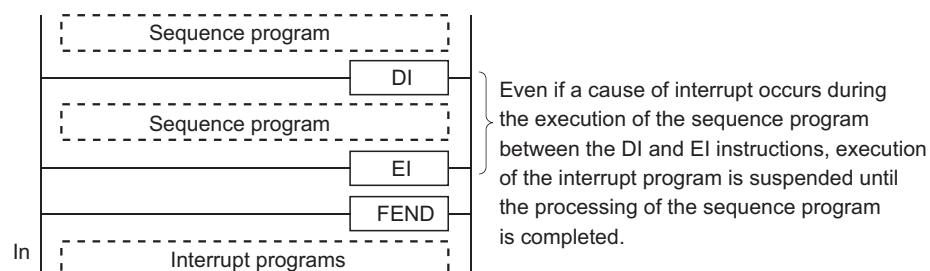
#### DI

- (1) Disables the execution of an interrupt program until the EI instruction has been executed, even if a start cause for the interrupt program occurs.
- (2) A DI state is entered when power is turned ON or when the CPU module is reset.

#### EI

The EI instruction is used to clear the interrupt disable state resulting from the execution of the DI instruction, and to create a state in which the interrupt program designated by the interrupt pointer number certified by the IMASK instruction can be executed.

When the IMASK instruction is not executed, I32 to I47 are disabled.



## IMASK

(1) Enables/disables the execution of the interrupt program marked by the designated interrupt pointer by using the bit pattern of 8 points from the device designated by  $\textcircled{S}$ .

- 1(ON)..... Interrupt program execution enabled
- 0(OFF).... Interrupt program execution disabled

(2) The interrupt pointer numbers corresponding to the individual bits are as shown below:

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
$\textcircled{S}$	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
$\textcircled{S} + 1$	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
$\textcircled{S} + 2$	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
$\textcircled{S} + 3$	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
$\textcircled{S} + 4$	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
$\textcircled{S} + 5$	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
$\textcircled{S} + 6$	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
$\textcircled{S} + 7$	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112

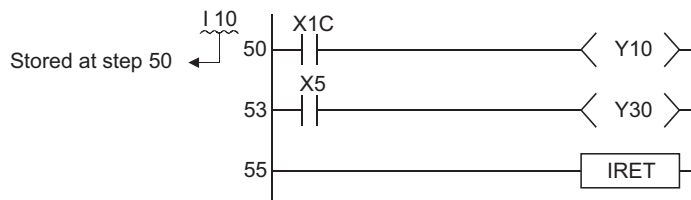
(3) When the power is turned ON or when the CPU module has been reset, the execution of interrupt programs I0 to I31,I48 to I127 is enabled, and the execution of interrupt programs I32 to I47 is disabled.

(4) The statuses of devices  $\textcircled{S}$ ,  $\textcircled{S} + 1$ ,  $\textcircled{S} + 2$ , and  $\textcircled{S} + 3$  to  $\textcircled{S} + 7$  are stored in SD715 to SD717 and SD781 to SD785 (storage area for the IMASK instruction mask pattern).

(5) Although the special registers are separated as SD715 to SD717 and SD781 to SD785, device numbers should be designated as  $\textcircled{S}$  to  $\textcircled{S} + 7$  successively.

## POINT

1. An interrupt pointer occupies 1 step.



2. For the information on interrupt conditions, link direct devices, refer to the QnUCPU User's Manual(Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual(Function Explanation, Program Fundamentals)

3. The DI state (interrupt disabled) is active during the execution of an interrupt program. Do not insert the EI instructions in interrupt programs to attempt the execution of multiple interrupts, with interrupt programs running inside interrupt programs.

4. If there are the EI and DI instructions within a master control, these instructions will be executed regardless of the execution/non-execution status of the MC instruction.

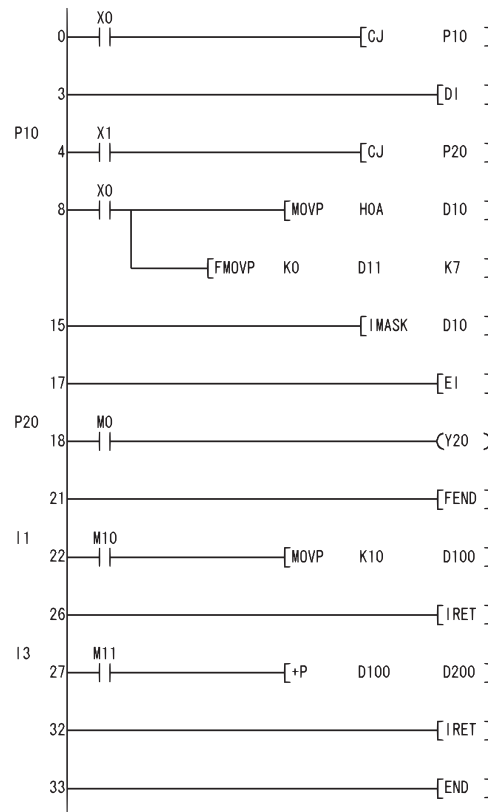
## ! Operation Error

- (1) There are no operation errors associated with the DI, EI or IMASK instruction.

## Program Example

- (1) The following program is designed to enable the execution of only the interrupt programs having the interrupt pointer numbers I1 and I3 while X0 is ON.

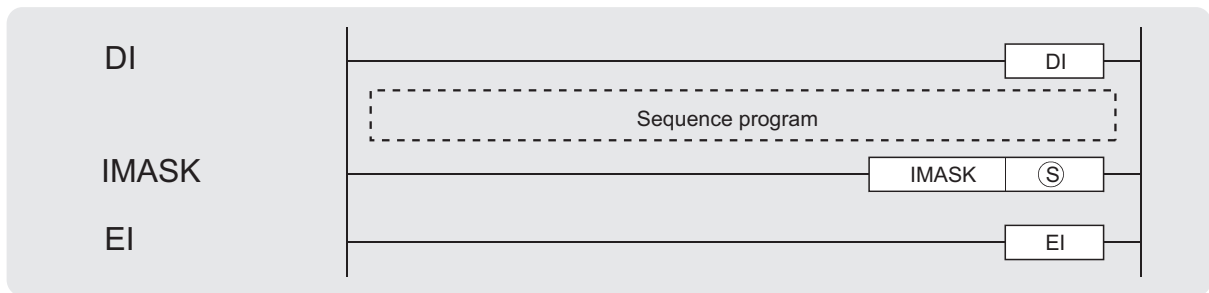
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	CJ	P10
3	DI	
4	P10	
5	LD	X1
6	CJ	P20
8	LD	X0
9	MOV P	HOA D10
11	FMOV P	K0 D11 K7
15	IMASK	D10
17	EI	
18	P20	
19	LD	MO
20	OUT	Y20
21	FEND	
22	I1	
23	LD	M10
24	MOV P	K10 D100
26	IRET	
27	I3	
28	LD	M11
29	+P	D100 D200
32	IRET	
33	END	

2 When the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU is used



Ⓢ : Head number of the devices where the interrupt mask data is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J <sub>00</sub>		U <sub>00</sub> G <sub>0</sub>	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○					—		

## ★ Function

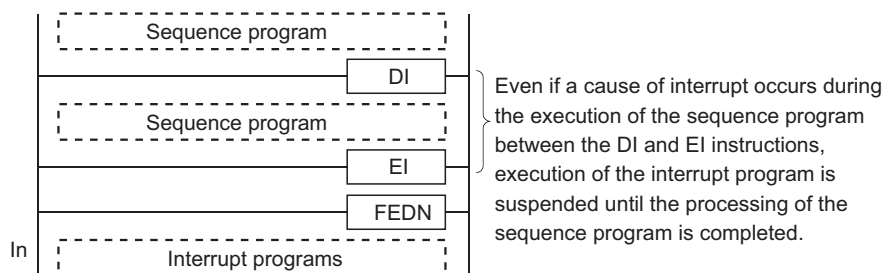
### DI

- (1) Disables the execution of an interrupt program until the EI instruction has been executed, even if a start cause for the interrupt program occurs.
- (2) A DI state is entered when power is turned ON or when the CPU module is reset.

### EI

The EI instruction is used to clear the interrupt disable state resulting from the execution of the DI instruction, and to create a state in which the interrupt program designated by the interrupt pointer number enabled by the IMASK instruction and the fixed cycle execution type program can be executed.

When the IMASK instruction is not executed, I32 to I47 are disabled.





## IMASK

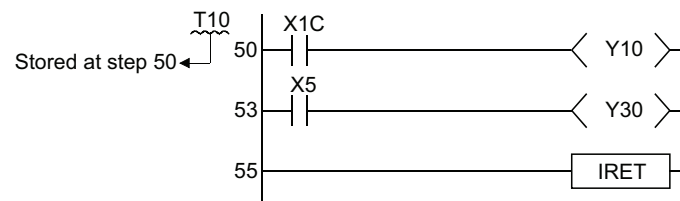
- (1) Enables/disables the execution of the interrupt program marked by the designated interrupt pointer by using the bit pattern of 16 points from the device designated by  $\textcircled{S}$ .
  - 1(ON)..... Interrupt program execution enabled
  - 0(OFF).... Interrupt program execution disabled
- (2) The interrupt pointer numbers corresponding to the individual bits are as shown below:

	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
$\textcircled{S}$	I15	I14	I13	I12	I11	I10	I9	I8	I7	I6	I5	I4	I3	I2	I1	I0
$\textcircled{S} + 1$	I31	I30	I29	I28	I27	I26	I25	I24	I23	I22	I21	I20	I19	I18	I17	I16
$\textcircled{S} + 2$	I47	I46	I45	I44	I43	I42	I41	I40	I39	I38	I37	I36	I35	I34	I33	I32
$\textcircled{S} + 3$	I63	I62	I61	I60	I59	I58	I57	I56	I55	I54	I53	I52	I51	I50	I49	I48
$\textcircled{S} + 4$	I79	I78	I77	I76	I75	I74	I73	I72	I71	I70	I69	I68	I67	I66	I65	I64
$\textcircled{S} + 5$	I95	I94	I93	I92	I91	I90	I89	I88	I87	I86	I85	I84	I83	I82	I81	I80
$\textcircled{S} + 6$	I111	I110	I109	I108	I107	I106	I105	I104	I103	I102	I101	I100	I99	I98	I97	I96
$\textcircled{S} + 7$	I127	I126	I125	I124	I123	I122	I121	I120	I119	I118	I117	I116	I115	I114	I113	I112
$\textcircled{S} + 8$	I143	I142	I141	I140	I139	I138	I137	I136	I135	I134	I133	I132	I131	I130	I129	I128
$\textcircled{S} + 9$	I159	I158	I157	I156	I155	I154	I153	I152	I151	I150	I149	I148	I147	I146	I145	I144
$\textcircled{S} + 10$	I175	I174	I173	I172	I171	I170	I169	I168	I167	I166	I165	I164	I163	I162	I161	I160
$\textcircled{S} + 11$	I191	I190	I189	I188	I187	I186	I185	I184	I183	I182	I181	I180	I179	I178	I177	I176
$\textcircled{S} + 12$	I207	I206	I205	I204	I203	I202	I201	I200	I199	I198	I197	I196	I195	I194	I193	I192
$\textcircled{S} + 13$	I223	I222	I221	I220	I219	I218	I217	I216	I215	I214	I213	I212	I211	I210	I209	I208
$\textcircled{S} + 14$	I239	I238	I237	I236	I235	I234	I233	I232	I231	I230	I229	I228	I227	I226	I225	I224
$\textcircled{S} + 15$	I255	I254	I253	I252	I251	I250	I249	I248	I247	I246	I245	I244	I243	I242	I241	I240

- (3) When the power is turned ON or when the CPU module has been reset, the execution of interrupt programs I0 to I31, I48 to I255 is enabled, and the execution of interrupt programs I32 to I47 is disabled.
- (4) The status of devices  $\textcircled{S}$ ,  $\textcircled{S} + 1$ ,  $\textcircled{S} + 2$ , and  $\textcircled{S} + 3$  to  $\textcircled{S} + 15$  are stored in SD715 to SD717 and SD781 to SD793 (storage area for the IMASK instruction mask pattern).
- (5) Although the special registers are separated as SD715 to SD717 and SD781 to SD793, device numbers should be designated as  $\textcircled{S}$  to  $\textcircled{S} + 15$  successively.

### POINT

1. An interrupt pointer occupies 1 step.



- For the information on interrupt conditions, link direct devices, refer to the QnUCPU User's Manual(Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual(Function Explanation, Program Fundamentals)
2. The DI state (interrupt disabled) is active during the execution of an interrupt program. Do not insert the EI instructions in interrupt programs to attempt the execution of multiple interrupts, with interrupt programs running inside interrupt programs.
  3. If there are the EI and DI instructions within a master control, these instructions will be executed regardless of the execution/non-execution status of the MC instruction.

## Operation Error

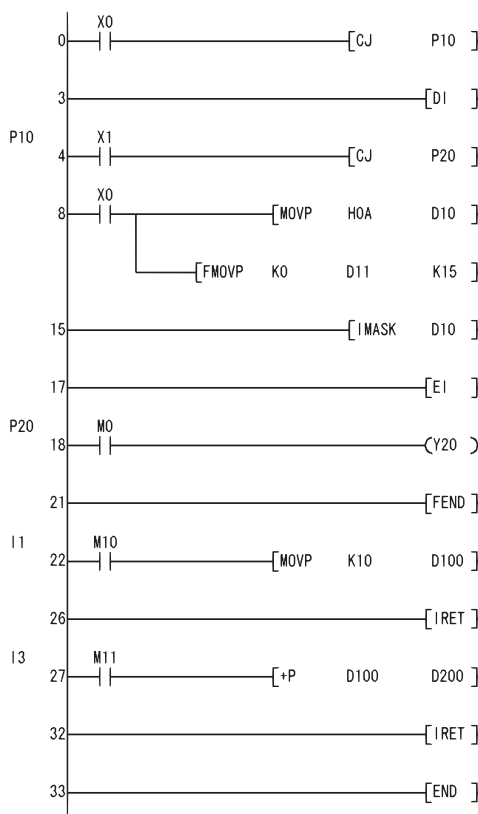
(1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The device specified by ⑤ exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

## Program Example

(1) The following program creates an execution enabled state for the interrupt program marked by the interrupt pointer number when X0 is ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	CJ	P10
3	DI	
4	P10	
5	LD	X1
6	CJ	P20
8	LD	X0
9	MOV P	HOA D10
11	FMOV P	KO D11 K15
15	IMASK	D10
17	EI	
18	P20	
19	LD	MO
20	OUT	Y20
21	FEND	
22	I1	
23	LD	M10
24	MOV P	K10 D100
26	IRET	
27	I3	
28	LD	M11
29	+P	D100 D200
32	IRET	
33	END	

## 6.6.2 Recovery from interrupt programs (IRET)

Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

- (1) Indicates the completion of interrupt program processing.
- (2) Returns to sequence program processing following the execution of the IRET instruction.

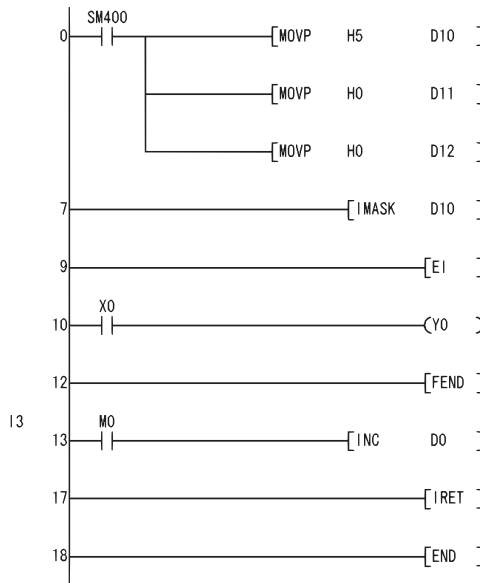
### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - There is no pointer corresponding to the interrupt number. (Error code: 4220)
  - The IRET instruction was executed before the interrupt program is executed. (Error code: 4223)
  - The END, FEND, GOEND, or STOP instruction has been executed after the generation of an interrupt and prior to the execution of the IRET instruction. (Error code: 4221)
  - The IRET instruction was executed during the fixed scan execution type program. (For the Universal model QCPU only) (Error code: 4223)

## Program Example

(1) The following program adds 1 to D0 if M0 is ON when the number 3 interrupt is generated.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOV P	H5 D10
3	MOV P	H0 D11
5	MOV P	H0 D12
7	IMASK	D10
9	EI	
10	LD	X0
11	OUT	Y0
12	FEND	
13	LD	M0
14	INC	D0
15	IRET	
17	IRET	
18	END	

## 6.7 I/O Refresh Instructions

### 6.7.1 I/O refresh (RFS(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the devices to be refreshed (bits)

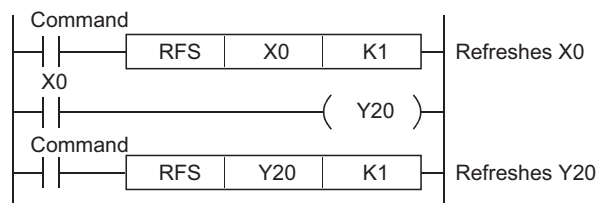
n : Number of refreshes (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○ (Only X, Y)								—
n	○				○				—

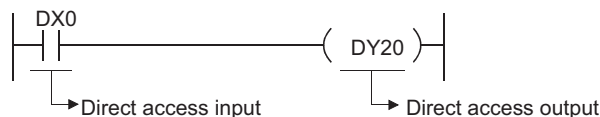
### ★ Function

1. Refreshes only the device being scanned during a scan, and functions to fetch input from external sources or to output data to an output module.
- 2) Fetching of input from or sending output to an external source is conducted in batch only after the execution of the END instruction, so it is not possible to output a pulse signal to an outside source during the execution of a scan.  
When the I/O refresh instruction is executed, the inputs (X) or outputs (Y) of the corresponding device numbers are refreshed forcibly midway through program execution. Therefore, a pulse signal can be output to an external source during a scan.
- 3) Use direct access inputs (DX) or direct access outputs (DY) to refresh inputs (X) or outputs (Y) in 1-point units.

[Program based on the RFS instruction]



[Program based on direct access input and direct access output]





## Operation Error

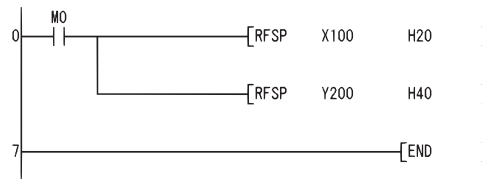
- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range n points from the device designated by (S) exceeds the proximate I/O range.  
(Error code: 4101)



## Program Example

- (1) The following program refreshes X100 to X11F and Y200 to Y23F when M0 goes ON.

[Ladder Mode]

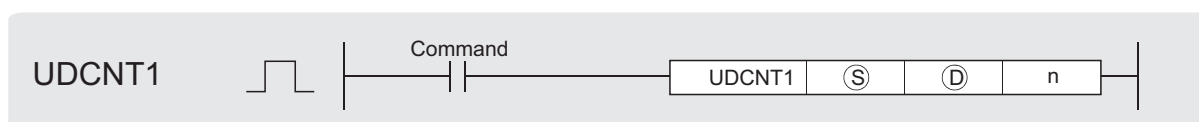


[List Mode]

Step	Instruction	Device
0	LD	M0
1	RFSP	X100 H20
4	RFSP	Y200 H40
7	END	

## 6.8 Other Convenient Instructions

### 6.8.1 Counter 1-phase input up or down (UDCNT1)



Ⓢ : Ⓢ + 0: Input number for count input (bits)

Ⓢ + 1: For setting count up/down (bits)

•OFF: Count up (add numbers when counting)

•ON: Count down (subtract numbers when counting)

Ⓧ : Number of the counter to be enabled to start counting with the UDCNT1 instruction (Device name)

n : Value to set (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○ (Only X)*1	—	—	—	—	—	—	—	—
Ⓧ	—	○ *(Only C)	—	—	—	—	—	—	—
n	△ *2	△ *2	△ *2	—	—	○	—	—	—

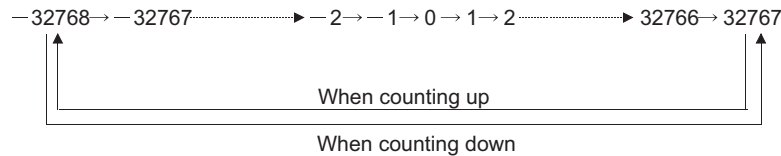
\*1: Only the X device can be used for Ⓢ. However, the X device can be used only in the range of number of I/O points (the number of accessible points to actual I/O modules).

\*2: Local devices and the file registers set for individual programs cannot be used.

#### ★ Function

- (1) When the input designated at Ⓢ goes from OFF to ON, the present value of the counter designated at Ⓧ will be updated.
- (2) The direction of the count is determined by the ON/OFF status of the input designated by Ⓢ+1.
  - OFF : Count up (counts by adding to the present value)
  - ON : Count down (counts by subtracting from the present value)
- (3) Count processing is conducted as described below:
  - When the count is going up, the counter contact designated at Ⓧ goes ON when the present value becomes identical with the setting value designated by n. However, the present value count will continue even when the contact of the counter designated at Ⓧ goes ON. (See Program Example (1))
  - When the count is going down, the counter for the contact designated at Ⓧ goes OFF when the present value reaches the set value - 1. (See Program Example (1))

- The counter designated at ④ is a ring counter. If it is counting up when the present value is 32767, the present value will become -32768. Further, if it is counting down when the present value is -32768, the present value will become 32767. The count processing performed on the present value is as shown below:



- (4) The UDCNT1 instruction triggers counting when the execution command is turned OFF→ON and suspends counting when the execution command is turned ON→OFF. When the execution command is turned OFF→ON again, the counting resumes from the suspended value.
- (5) The RST instruction clears the present value of the counter designated at ④ and turns the contact OFF.

### POINT

1. With the UDCNT1 instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be counted must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of individual modules is shown below:

CPU Module Type Name	Interrupt Interval
High Performance model QCPU, Process CPU, Universal model QCPU	1 ms

2. The set value cannot be changed during counting directed by the UDCNT1 instruction (while the execution command is ON). To change the set value, turn OFF the execution command.
3. Counters which have been designated by the UDCNT1 instruction cannot be used by other instructions. If they are used by other instructions, they will not be capable of returning an accurate count.
4. The UDCNT1 instruction can be used as many as 6 times within all the programs being executed. The seventh and the subsequent UDCNT1 instructions are not processed.



### Operation Error

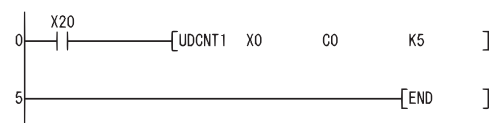
- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by ⑤ exceeds the range of the corresponding device.  
(Error code: 4101)



## Program Example

- (1) This program uses C0 (Up/Down counter) to count the number of times X0 goes from OFF to ON after X20 has gone ON.

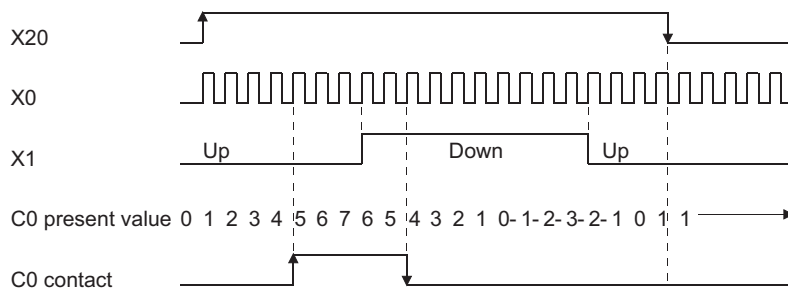
[Ladder Mode]



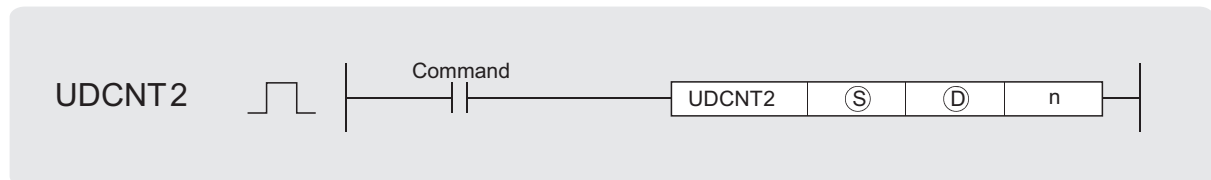
[List Mode]

Step	Instruction	Device
0	LD	X20
1	UDCNT1	X0
5	END	C0 K5

[Operation]



## 6.8.2 Counter 2-phase input up or down (UDCNT2)



Ⓢ : Ⓢ + 0: Input number for count input (A phase pulse) (bits)

Ⓢ + 1: Input number for count input (B phase pulse) (bits)

Ⓧ : Number of the counter to be enabled to start counting with the UDCNT2 instruction (Device name)

n : Value to set (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	Jm		U\GO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○ (Only X)*1	—	—			—			—
Ⓧ	—	○ *(Only C)	—			—			—
n	△ *2	△ *2	△ *2			○			—

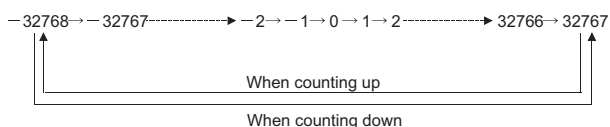
\*1: Only the X device can be used for Ⓢ. However, the X device can be used only in the range of number of I/O points (the number of accessible points to actual I/O modules).

\*2: Local devices and the file registers set for individual programs cannot be used.

### ★ Function

- (1) The present value of the counter designated by Ⓧ is updated depending on the status of the input designated by Ⓢ (A phase pulse) and the status of the input designated by Ⓢ+1 (B phase pulse).
- (2) Direction of the count is determined in the following manner:
  - When Ⓢ is ON, if Ⓢ+1 goes from OFF to ON, count up operation is performed (values are added to the present value of the counter).
  - When Ⓢ is ON, if Ⓢ+1 goes from ON to OFF, count down operation is performed (values are subtracted from the present value of the counter).
  - No count operation is performed if Ⓢ is OFF.
- (3) Count processing is conducted as described below:
  - When the count is going up, the counter contact designated at Ⓧ goes ON when the present value becomes identical with the setting value designated by n. However, the present value count will continue even when the contact of the counter designated at Ⓧ goes ON. (See Program Example (1))
  - When the count is going down, the counter for the contact designated at Ⓧ goes OFF when the present value reaches the set value - 1. (See Program Example (1))

- The counter designated at ④ is a ring counter. If it is counting up when the present value is 32767, the present value will become -32768. Further, if it is counting down when the present value is -32768, the present value will become 32767. The count processing performed on the present value is as shown below:



- (4) Count processing conducted according to the UDCNT2 instruction begins when the count command goes from OFF to ON, and is suspended when it goes from ON to OFF. When the execution command is turned OFF to ON again, the counting resumes from the suspended value.
- (5) The RST instruction clears the present value of the counter designated at ④ and turns the contact OFF.

### ❏ POINT

1. With the UDCNT2 instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be counted must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of individual modules is shown below:

CPU Module Type Name	Interrupt Interval
High Performance model QCPU, Process CPU, Universal model QCPU	1 ms

2. The set value cannot be changed during counting directed by the UDCNT2 instruction (while the execution command is ON). To change the set value, turn OFF the execution command.
3. Counters designated by the UDCNT2 instruction cannot be used by any other instruction. If they are used by other instructions, they will not be capable of returning an accurate count.
4. The UDCNT2 instruction can be used as many as 5 times within all the programs being executed. The sixth and the subsequent UDCNT2 instructions are not processed.

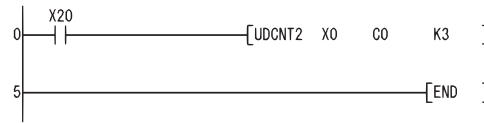
### ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by ⑤ exceeds the range of the corresponding device.  
(Error code: 4101)

## Program Example

- (1) The following program performs a count operation as instructed by C0 (count up or down) on the status of X0 and X1 after X20 has gone ON.

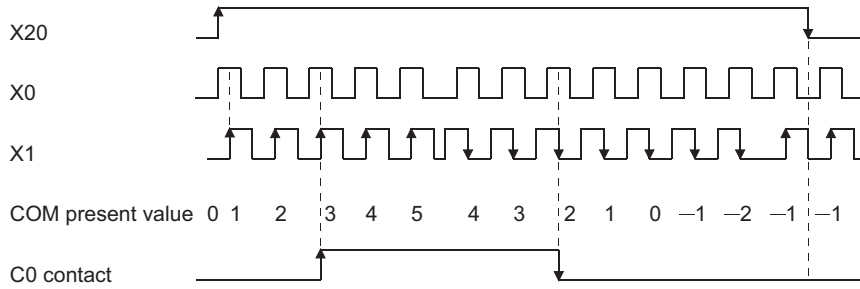
[Ladder Mode]



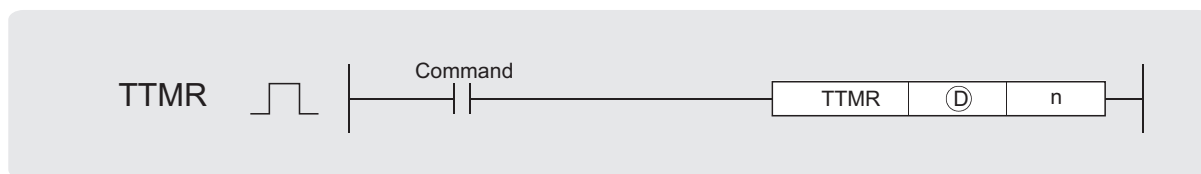
[List Mode]

Step	Instruction	Device
0	LD	X20
1	UDCNT2	X0 C0 K3
5	END	

[Operation]



## 6.8.3 Teaching timer (TTMR)



$\text{D}$  :  $\text{D} + 0$ : The device where measurement value is stored (BIN 16 bit)

$\text{D} + 1$ : For CPU module system use (BIN 16 bit)

$n$  : Measurement value multiplier (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JMP		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
$\text{D}$	—	○				—			—
$n$	—	○				○			—

### ★ Function

- Measures the time while the execution command is ON in units of seconds, and stores the multiplied value of the measured time by the multiplier specified by  $n$  at the device designated by  $\text{D}$ .
- Clears the device designated by  $\text{D}+0$  or  $\text{D}+1$  when the execution command is turned OFF→ON.
- The multipliers that can be designated by  $n$  are as shown below:

$n$	Multiplier
0	1
1	10
2	100

### ☒ POINT

- Time measurements are conducted when the TTMR instruction is executed. Using the JMP or similar instruction to jump the TTMR instruction will make it impossible to get an accurate measurement.
- Do not change the multiplier designated by  $n$  while the TTMR instruction is being executed. Changing this multiplier will result in an inaccurate value being returned.
- The TTMR instruction can also be used in low speed execution type programs.
- The device designated by  $\text{D}+1$  is used by the system of the CPU module, so users should not change its value. If users do change this value, the value stored in the device designated by  $\text{D}$  will no longer be accurate.

- No processing is performed when the value specified by " $n$ " is other than 0 to 2.



## Operation Error

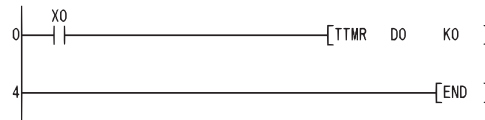
- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified by ④ exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)



## Program Example

- (1) The following program stores the amount of time that X0 is ON at D0.

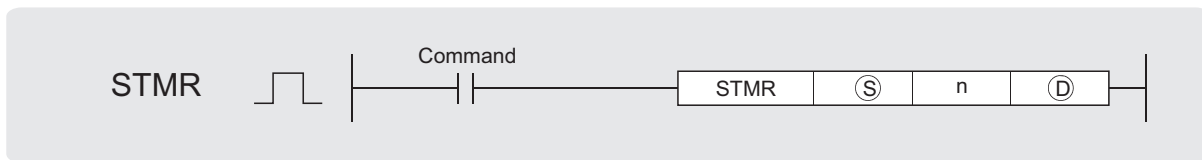
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	TTMR	D0 K0
4	END	

## 6.8.4 Special function timer (STMR)



- Ⓢ : Timer number (word)
- n : Value to set (BIN 16 bits).
- Ⓣ : Ⓣ + 0: Off delay timer output (bits)
  - Ⓣ + 1: One shot timer output after OFF (bits)
  - Ⓣ + 2: One shot timer output after ON (bits)
  - Ⓣ + 3: ON delay and Off delay timer output (bits)

Setting Data	Internal Devices		R, ZR	JOG		U/G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	△ *1	—			—			—
n	○	○	○			○			—
Ⓣ	○	—	—			—			—

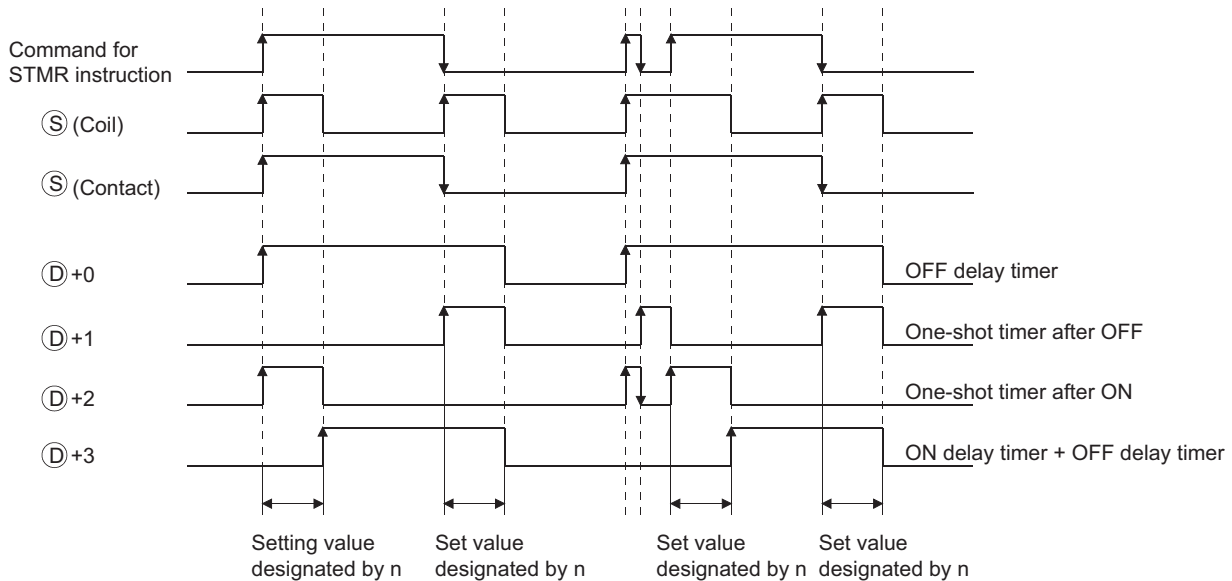
\*1: Can be used only by timer (T) data

### ★ Function

- (1) The STMR instruction uses the 4 points from the device designated by Ⓣ to perform four types of timer output.
  - OFF delay timer output (Ⓣ+0)  
Goes ON at the leading edge of the command for the STMR instruction, and after the trailing edge of the command, goes OFF when the amount of time designated by n has passed.
  - One shot timer output after OFF (Ⓣ+1)  
Goes ON at the trailing edge of the command for the STMR instruction, and goes OFF when the amount of time designated by n has passed.
  - One shot timer output after ON (Ⓣ+2)  
Goes ON at the leading edge of the command for the STMR instruction, and goes OFF either when the amount of time designated by n has passed, or when the command for the STMR instruction goes OFF.
  - ON delay timer output (Ⓣ+3)  
Goes ON at the trailing edge of the timer coil, and after the trailing edge of the command for the STMR instruction, goes OFF when the amount of time designated by n has passed.
- (2) The timer coil designated by Ⓢ turns ON at the leading edge and trailing edge of the command for the STMR instruction, and starts measurement of the present value.
  - The timer coil measures to the point where the value reaches the set value designated by n, then enters a time up state and goes OFF.
  - If the command for the STMR instruction goes OFF before the timer coil reaches the time up state, it will remain ON. Timer measurement is continued at this time. When the STRM instruction command goes ON once again, the present value will be cleared to 0 and measurement will begin once again.

- (3) The timer contact goes ON at the leading edge of the command for the STMR instruction, and after the trailing edge is reached, the timer coil goes OFF at the trailing edge of the STMR instruction command.

The timer contact is used by the CPU module system, and cannot be used by the user.



- (4) Measurement of the present value of the timer specified by the STMR instruction is executed regardless of the command ON/OFF status of the STMR instruction. If the STMR instruction is jumped with the JMP or similar instruction, it will not be possible to get accurate measurement.
- (5) Measurement unit for the timer designated by ④ is identical to the low speed timer.
- (6) A value between 0 to 32767 can be set for n.  
No operation if n is other than 0 to 32767.
- (7) The timer designated by ⑤ cannot be used by the OUT instruction. If the STMR instruction and the OUT instruction use the same timer number, accurate operation will not be conducted.

## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified by ④ exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

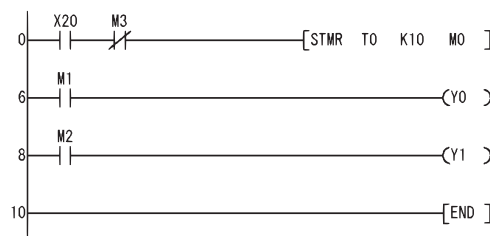


## Program Example

- (1) The following program turns Y0 and Y1 ON and OFF once each second (flicker) when X20 is ON.

(Uses 100 ms timer)

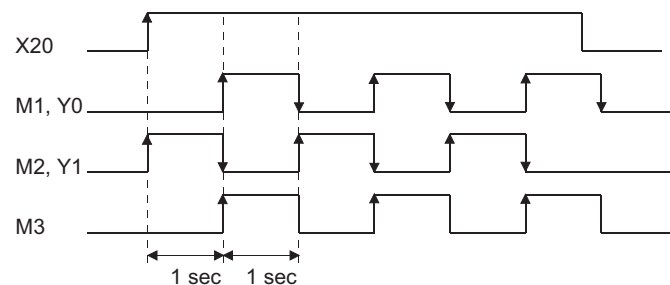
[Ladder Mode]



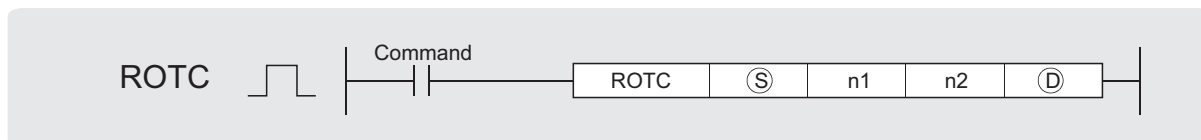
[List Mode]

Step	Instruction	Device
0	LD	X20
1	ANI	M3
2	STMR	T0 K10 M0
6	LD	M1
7	OUT	Y0
8	LD	M2
9	OUT	Y1
10	END	

[Timing Chart]



## 6.8.5 Rotary table shortest direction control (ROTC)



- Ⓢ : Ⓢ + 0 : Measures the number of table rotations (for system use) (BIN 16 bits)
- Ⓢ + 1 : Call station number (BIN 16 bits)
- Ⓢ + 2 : Call item number (BIN 16 bits)
- n1 : Number of divisions of table (2 to 32767) (BIN 16 bits)
- n2 : Number of low-speed sections (value from 0 to less than n1) (BIN 16 bits)
- Ⓧ : Ⓧ + 0 : A phase input signal (bits)
- Ⓧ + 1 : B phase input signal (bits)
- Ⓧ + 2 : 0 point detection input signal (bits)
- Ⓧ + 3 : High speed forward rotation output signal (for system use) (bits)
- Ⓧ + 4 : Low speed forward rotation output signal (for system use) (bits)
- Ⓧ + 5 : Stop output signal (for system use) (bits)
- Ⓧ + 6 : Low speed reverse rotation output signal (for system use) (bits)
- Ⓧ + 7 : High speed reverse rotation output signal (for system use) (bits)

Setting Data	Internal Devices		R, ZR	JOG		UJOG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
n1	○	○				○			—
n2	○	○				○			—
Ⓧ	○	—				—			—

### ★ Function

- (1) This control functions to enable shortest direction control of the rotary table to the position of the station number designated by Ⓢ+1 in order to remove or deposit an item whose number has been designated by Ⓢ+2 on a rotary table with equal divisions of the value designated by n1.
- (2) The item number and station number are controlled as items allocated by counterclockwise rotation.
- (3) The system uses Ⓢ+0 as a counter to instruct it as to what item is at which number counting from station number 0. Do not rewrite the sequence program data.  
Accurate controls will not be possible in cases where users have rewritten the data.
- (4) The value of n2 should be less than the number of table divisions specified by n1.
- (5) Ⓧ+0 and Ⓧ+1 are A and B phase input signals that are used to detect whether the direction of the rotary table rotation is forward or reverse.  
The direction of rotation is judged by whether the B phase pulse is at its leading or trailing edge when the A phase pulse is ON:
  - When the B phase is at the leading edge: Forward rotation (clockwise rotation)
  - When the B phase is at the trailing edge: Reverse rotation (counterclockwise rotation)

- (6)  $\textcircled{D}+2$  is the 0 point detection output signal that goes ON when item number 0 has arrived at the No. 0 station.  
When the device designated by  $\textcircled{D}+2$  goes ON while the ROTC instruction is being executed,  $\textcircled{S}+0$  is cleared.  
It is best to perform this clear operation first, then to begin shortest direction control with the ROTC instruction.
- (7) The data from  $\textcircled{D}+3$  to  $\textcircled{D}+7$  consists of output signals needed to control the table's operation.  
The output signal of one of the devices from  $\textcircled{D}+3$  to  $\textcircled{D}+7$  will go ON in response to the execution results of the ROTC instruction.
- (8) If the command for the ROTC instruction is OFF, clears all  $\textcircled{D}+3$  to  $\textcircled{D}+7$  without performing shortest direction control.
- (9) The ROTC instruction can be used only one time in all programs where it is executed.  
Attempts to use it more than one time will result in inaccurate operations.
- (10) No processing is performed when the value of  $\textcircled{S}+0$  to  $\textcircled{S}+2$ , or the value of n2 is greater than n1.

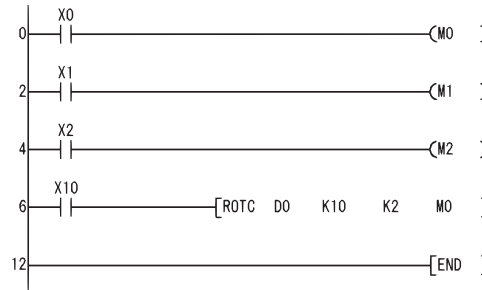
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified by  $\textcircled{S}$  or  $\textcircled{D}$  exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

## Program Example

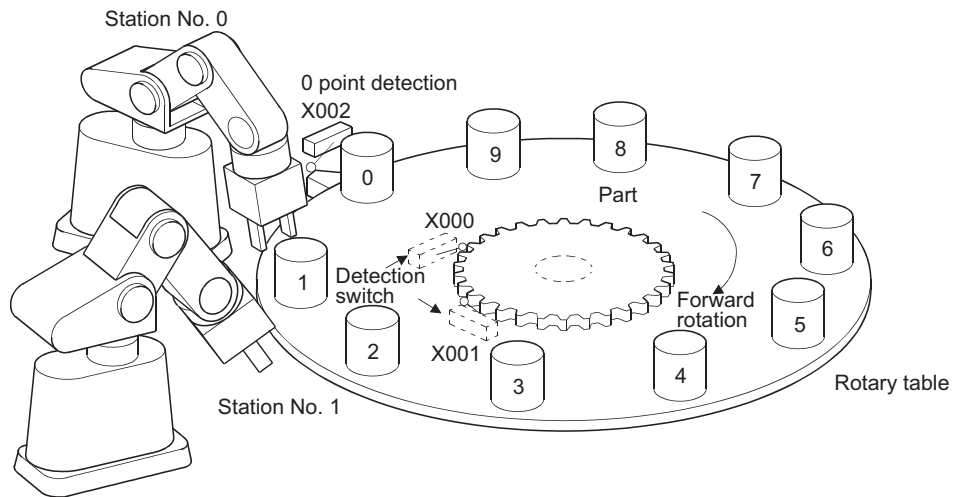
- (1) The following program deposits the item at section D2 on a 10-division rotary table at the station at section D1, and the two sections ahead and behind this determine the rotation direction and control speed of the motor when the table is being rotated at low speed.

[Ladder Mode]



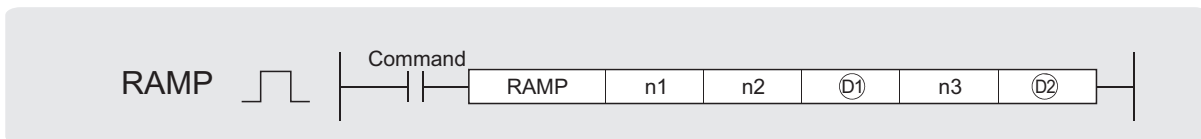
[List Mode]

Step	Instruction	Device
0	LD	X0
1	OUT	M0
2	LD	X1
3	OUT	M1
4	LD	X2
5	OUT	M2
6	LD	X10
7	ROTC	D0 K10 K2 M0
12	END	



# 6.8.6 Ramp signal (RAMP)

Basic
High performance
Process
Redundant
Universal



- n1 : Initial value (BIN 16 bits)
- n2 : Final value (BIN 16 bits)
- (D1) : (D1) + 0 : Present value (BIN 16 bits)
- (D1) + 1 : Number of executions (BIN 16 bits)
- n3 : Number of shifts (BIN 16 bits)
- (D2) : (D2) + 0 : Completion device (bits)
- (D2) + 1 : Bit for selecting data retaining at completion (bit)

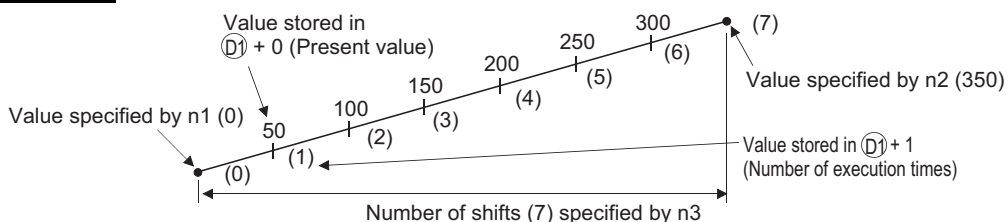
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n1	○				○			○	—
n2	○				○			○	—
(D1)	○				○			—	—
n3	○				○			○	—
(D2)	○				—			—	—

## ★ Function

- (1) When the execution command is ON, the following processing is executed.
  - Shifts from the value specified by n1 to the value specified by n2 in the number of times specified by n3.
  - For n3, designate the number of scans (number of shifts) required for shift from n1 to n2. No operation if other than 0 < n3 < 32768.
  - The system uses (D1)+1 to store the number of times the instruction has been executed.
  - The value of one variation (one scan) is obtained by the expression below:

$$\text{Value of one variation (one scan)} = \frac{(\text{Value specified by } n2) - (\text{Value specified by } n1)}{(\text{Value specified by } n3)}$$

**Example** 0 is varied to 350 in seven scans as shown below.



When the calculated one variation is indivisible, compensation is made to achieve the value specified in n2 by the number of shifts specified in n3. Hence, a linear ramp may not be made.

- (2) If the scan is performed for the number of moves specified by n3, the complete device specified by  $\text{D}2+0$  is turned ON.  
The ON/OFF status of the completion device and the contents of  $\text{D}1+0$  are determined by the ON/OFF status of the device designated by  $\text{D}2+1$ .
- When  $\text{D}2+1$  is OFF, +0 will go OFF at the next scan, and the RAMP instruction will begin a new move operation from the value currently at  $\text{D}2+0$ .
  - When  $\text{D}2+1$  is ON,  $\text{D}2+0$  will remain ON, and the contents of  $\text{D}1+0$  will not change.
- (3) When the command is turned OFF during the execution of this instruction, the contents of  $\text{D}1+0$  will not change following this.  
When the command goes ON again, the RAMP instruction will begin a new move from the present value at +0.
- (4) Do not change the specified values in n1 and n2 before the completion device specified in  $\text{D}2+0$  turns ON.  
Since the same expression is used every scan to calculate the value stored in  $\text{D}1+1$ , changing n1/n2 may cause a sudden variation.



## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified by  $\text{D}1$  or  $\text{D}2$  exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)



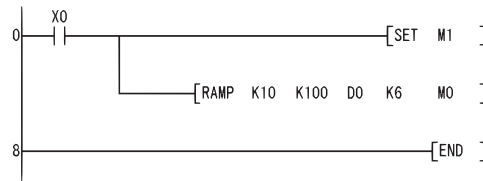
## Caution

- (1) When the digit specification of bit device is made to  $\text{D}1$ , the digit specification of bit device can only be used when the following condition is met.
- Specification of digits: K8

## Program Example

- (1) The following program changes the contents of D0 from 10 to 100 in a total of 6 scans, and saves the contents of D0 when the move has been completed.

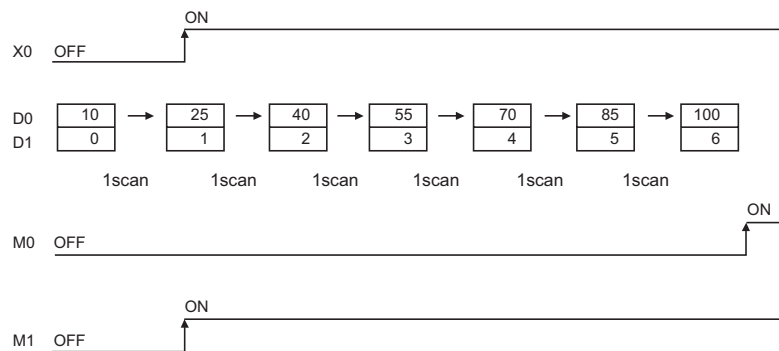
[Ladder Mode]



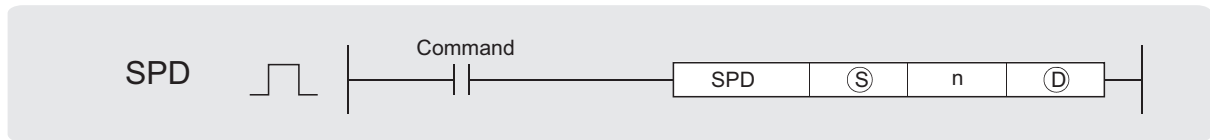
[List Mode]

Step	Instruction	Device
0	LD	X0
1	SET	M1
2	RAMP	K10 K100 D0 K6 MO
8	END	

[Timing Chart]



## 6.8.7 Pulse density measurement (SPD)



Ⓢ : Pulse input (bits)

n : Measurement time (unit: ms) (BIN 16 bits)

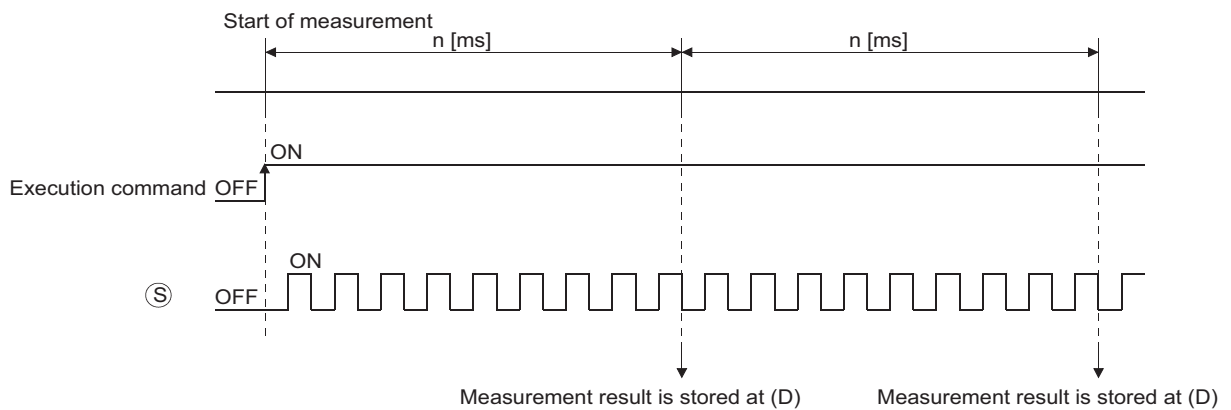
Ⓣ : Head number of the devices where the measurement result will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○ (Only X)	—				—			—
n	△ *1	△ *1				○			—
Ⓣ	—	△ *1				—			—

\*1: Local devices and the file registers set for individual programs cannot be used.

### ★ Function

- (1) The number of turning OFF→ON input of the device specified by Ⓢ is counted for just the amount of time specified by n, and the count results are stored in the device specified by Ⓣ.



- (2) When measurement directed by the SPD instruction has been completed, measurement is done again from 0.  
Turn OFF the execution command to stop the measurement directed by the SPD instruction.



## POINT

1. With the SPD instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be counted must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of individual modules is shown below:

CPU Module Type Name	Interrupt Interval
High Performance model QCPU, Process CPU, Universal model QCPU	1 ms

2.
  - When the High Performance model QCPU or Process CPU is used:  
The instruction is not processed when n = 0.
3. The SPD instruction can be used as many as 6 times within all the programs being executed. The seventh and the subsequent SPD instructions are not processed.
4. While the measurement is in execution (while the command input is ON) by the SPD instruction, the setting value cannot be changed. Turn OFF the command input before changing the setting value.

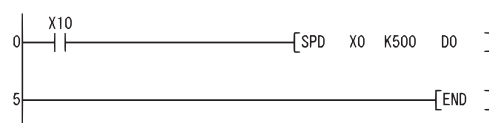
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by (S) exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program measures the pulses input to X0 for a period of 500 ms when X10 goes ON, and stores the result at D0.

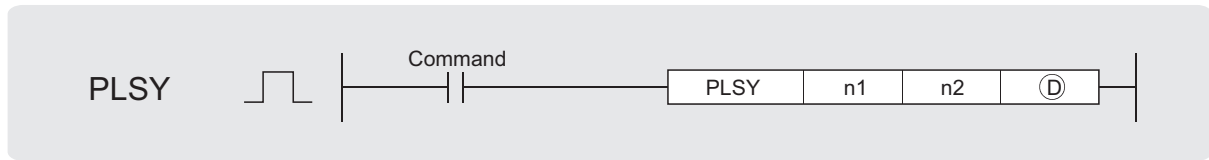
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	SPD	X0 K500 D0
5	END	

## 6.8.8 Fixed cycle pulse output (PLSY)



n1 : Frequency or the number of the device where frequency is stored (BIN 16 bits)

n2 : Outputs count or the number of the device where the outputs count is stored (BIN 16 bits)

Ⓣ : Number of the device to which pulses are output (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n1	○								—
n2	○								—
Ⓣ	△ *1								—

\*1: Only output (Y) can be used.

### ★ Function

- (1) Outputs a pulse at a frequency designated by n1 the number of times designated by n2, to the output module with the output signal (Y) designated by Ⓣ.
- (2) Frequencies between 1 to 100 Hz can be designated by n1.  
If n1 is other than 1 to 100 Hz, the PLSY instruction will not be executed.
- (3) The number of outputs that can be designated by n2 is between 1 to 65535 (0000<sub>H</sub> to FFFF<sub>H</sub>).  
If n2 is set to "0", pulses are continuously output.
- (4) Only an output number corresponding to the output module can be designated for pulse output at Ⓣ.
- (5) Pulse output commences with the command leading edge of the PLSY instruction.  
Pulse output is suspended when the PLSY instruction command goes OFF.

**POINT**

1. With the PLSY instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) For this reason, the pulses that can be output must have longer ON and OFF times than the interrupt interval of the CPU module. The interrupt interval of individual modules is shown below:

CPU Module Type Name	Interrupt Interval
High Performance model QCPU, Process CPU, Universal model QCPU	1 ms

2. Do not change the argument for the PLSY instruction during pulse output directed by the PLSY instruction (while the execution command is ON). To change the argument, turn OFF the execution command.
3. The PLSY instruction can be used only once in all programs executed by the CPU module. The second and the subsequent PLSY instructions are not processed.

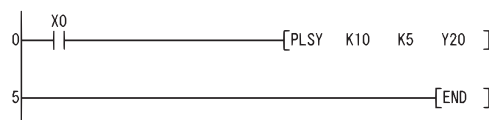
**! Operation Error**

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by  $\text{\textcircled{D}}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

**Program Example**

- (1) The following program outputs a 10 Hz pulse 5 times to Y20 when X0 is ON.

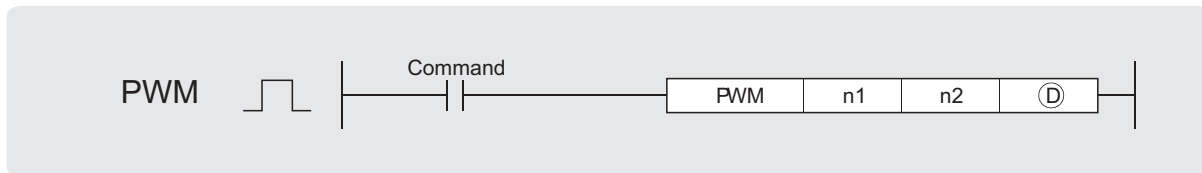
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	PLSY	K10 K5 Y20
5	END	

## 6.8.9 Pulse width modulation (PWM)



n1 : ON time or the number of the device where the ON time is stored (BIN 16 bits)

n2 : Frequency or the number of the device where the frequency is stored (BIN 16 bits)

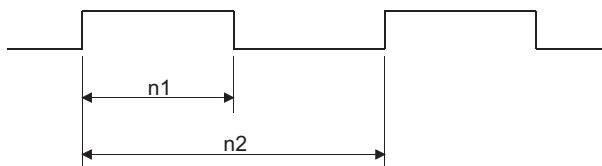
Ⓣ : Number of the device to which pulses are output (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n1	<input type="radio"/>				<input type="radio"/>				—
n2	<input type="radio"/>				<input type="radio"/>				—
Ⓣ	<input type="checkbox"/>	*1			—				—

\*1: Only output (Y) can be used.

### ★ Function

- Outputs the pulse of the cycle set by n2, for the amount of time ON designated by n1, to the output module designated by Ⓣ.



- The setting ranges for n1 and n2 are shown below:

CPU Module Type Name	Setting Range for n1 and n2 [ms] *2
High Performance model QCPU, Process CPU, Universal model QCPU	1 to 65535 (0001 <sub>H</sub> to FFFF <sub>H</sub> )

\*2: The value specified by n1 should be less than the value specified by n2.

**POINT**

1. With the PWM instruction, the argument device data is registered in the work area of the CPU module and counting operation is processed as a system interrupt. (The device data registered in the work area is cleared by turning the execution command OFF, or turning the STOP/RUN switch STOP→RUN.) The interrupt interval of individual modules is shown below:

CPU Module Type Name	Interrupt Interval of n1, n2
High Performance model QCPU, Process CPU, Universal model QCPU	1 ms

For this reason, the PWM instruction can be used only once within all the programs being executed by the CPU module.

2. The instruction is not processed in the following cases:
  - When both n1 and n2 are 0
  - When  $n1 \geq n2$
  - When the PWM instruction is executed twice or more.
3. Do not change the argument for the PWM instruction during pulse output directed by the PWM instruction (while the execution command is ON). To change the argument, turn OFF the execution command.

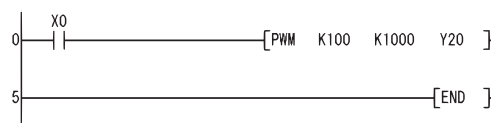
**Operation Error**

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by  $\text{D}$  exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)

**Program Example**

- (1) The following program outputs a 100 ms pulse once each second to Y20 when X0 is ON.

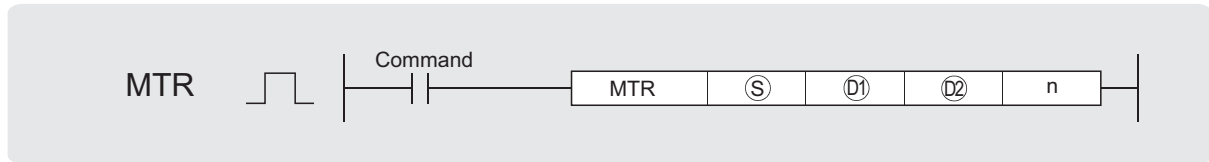
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	PWM	K100 K1000 Y20
5	END	

## 6.8.10 Matrix input (MTR)



Ⓢ : Head input device (bits)

Ⓛ1 : Head output device (bits)

Ⓛ2 : Head number of the devices where matrix input data will be stored (bits)

n : Number of input rows (BIN 16 bit)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	<input type="radio"/> (Only X)								—
Ⓛ1	<input type="radio"/> (Only Y)								—
Ⓛ2	<input type="radio"/>								—
n	<input type="radio"/>				<input type="radio"/>				—

### ★ Function

- (1) It reads the input from 16 points × n-rows starting from the input number designated by Ⓢ, then stores fetched input data from the device designated by Ⓛ2 onward.
- (2) One row (16 points) can be fetched in 1 scan.
- (3) Fetching from the first to the n th row is repeated.
- (4) The first through the 16th points store the first row of data and the next 16 points store the second row of data at the devices following the device designated by Ⓛ2.  
For this reason, the space of 16 × n points from the device designated by Ⓛ2 are occupied by the MTR instruction.
- (5) Ⓛ1 is the output needed to select the row which will be fetched, and the system automatically turns it ON and OFF.  
It uses the n points from the device designated by Ⓛ1.
- (6) Only device numbers divisible by 16 can be designated for Ⓢ, Ⓛ1 and Ⓛ2.
- (7) For n, a value in the range from 2 to 8 can be assigned.
- (8) No processing is performed in the following cases.
  - The device number designated by Ⓢ, Ⓛ1, or Ⓛ2 is not divisible by 16.
  - The device designated by Ⓢ is outside the actual input range.
  - The device designated by Ⓛ1 is outside the actual output range.
  - The space 16 × n points following the device designated by Ⓛ2 exceeds the relevant device range.
  - The value for n is not between 2 and 8.

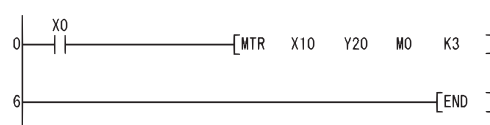
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device other than the input (X) was specified at (S). (Error code: 4101)
  - The device other than the output (Y) was specified at (D). (Error code: 4101)

## Program Example

- (1) The following program fetches, when X0 is turned ON, the 16 points × 3 matrix starting from X10, and stores the matrix into the area starting from M0.

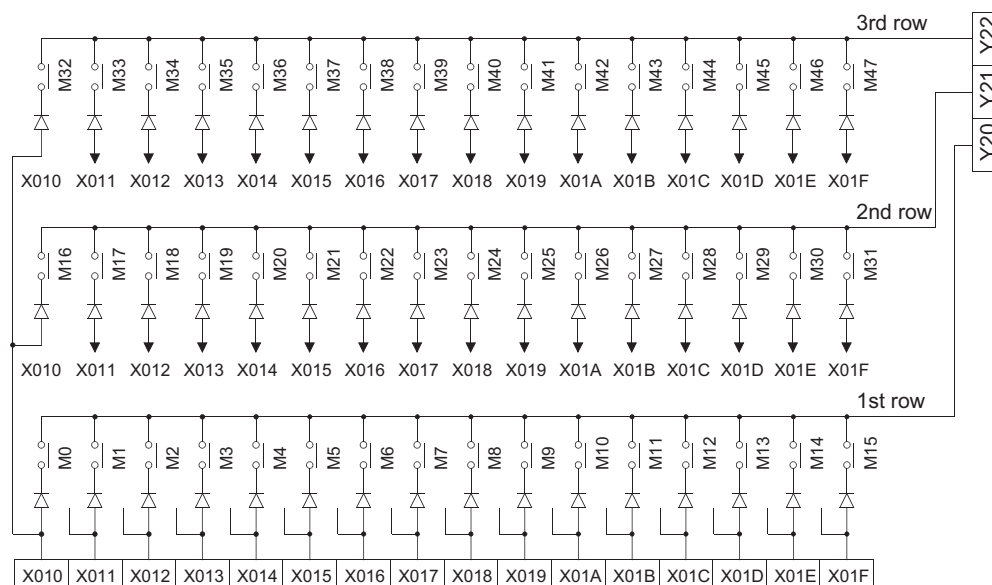
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	MTR	X10 Y20 M0 K3
6	END	

[Operation]



## Caution

- (1) Note that the MTR instruction directly operates on actual input and output.
- The output (D) that had been turned ON by the MTR instruction does not turn OFF when the MTR command turns OFF.
- Turn OFF the specified output (D) in the sequence program.
- (2) The MTR instruction execution interval must be longer than the total of response time of input and output modules.
- If the set interval is shorter than the value indicated above, an input cannot be read correctly. If the scan time in a sequence program is short, select the constant scan and set the scan time longer than the total of response time.





# 7

# APPLICATION INSTRUCTIONS

Category	Processing Details	Reference section
Logical operation instructions	Logical operations such as logical sum, logical product, etc.	Section 7.1
Rotation instruction	Rotation of designated data	Section 7.2
Shift instruction	Shift of designated data	Section 7.3
Bit processing instructions	Sets and resets bit data; bit extraction	Section 7.4
Data processing instructions	Data processing including data searching, sorting, decoding and encoding	Section 7.5
Structure creation instructions	Repeated operation, subroutine program calls, index modification in ladder units	Section 7.6
Data Table Operation Instructions	Data table read/write	Section 7.7
Buffer memory access instruction	Read/write from/to the buffer memory of intelligent function modules	Section 7.8
Display instructions	Character code outputs to external devices and displays on indicators	Section 7.9
Debugging and failure diagnosis instructions	Check, status latch, sampling trace, program trace	Section 7.10
Character string processing instructions	Character string (ASCII code data) processing	Section 7.11
Special function instructions	BCD real number and floating point real number processing	Section 7.12
Data control instructions	Output value controls based on input data range checks	Section 7.13
File register switching instructions	Sets file registers; switches block numbers	Section 7.14
Clock instructions	Reading, writing, addition, subtraction, and conversion of clock values; comparison between the clock values; and comparison between the date values	Section 7.15
Expansion clock instruction	Reading, addition, and subtraction of clock values up to millisecond	Section 7.16
Program control instructions	Instructions to switch program execution conditions	Section 7.17
Other instructions	Instructions that do not fit in the above categories, such as watchdog timer reset instructions and timing clock instructions	Section 7.18

## 7.1 Logical operation instructions

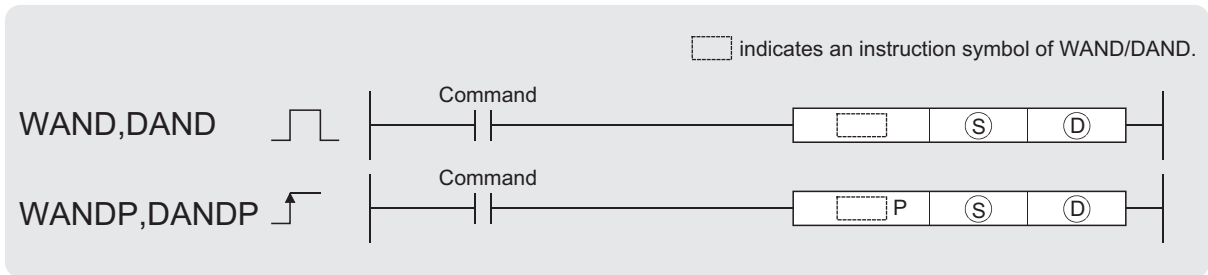
- (1) The logical operation instructions perform logical sum, logical product or other logical operations in 1-bit units.

Category	Processing Details	Formula for Operation	Example		
			A	B	Y
Logical product (AND)	Becomes 1 only when both input A and input B are 1; otherwise, is 0	$Y = A \cdot B$	0	0	0
			0	1	0
			1	0	0
			1	1	1
Logical sum (OR)	Becomes 0 only when both input A and input B are 0; otherwise, is 1	$Y = A + B$	0	0	0
			0	1	1
			1	0	1
			1	1	1
Exclusive OR (XOR)	Becomes 0 if input A and input B are equal; otherwise, is 1	$Y = \bar{A} \cdot B + A \cdot \bar{B}$	0	0	0
			0	1	1
			1	0	1
			1	1	0
NON exclusive logical sum (XNR)	Becomes 1 if input A and input B are equal; otherwise, is 0	$Y = (\bar{A} + B)(A + \bar{B})$	0	0	1
			0	1	0
			1	0	0
			1	1	1

# 7.1.1 Logical products with 16-bit and 32-bit data (WAND(P),DAND(P))

Basic High performance Process Redundant Universal

1 When two data are set ( $\textcircled{D} \wedge \textcircled{S} \rightarrow \textcircled{D}$ ,  $(\textcircled{D} + 1, \textcircled{D}) \wedge (\textcircled{S} + 1, \textcircled{S}) \rightarrow (\textcircled{D} + 1, \textcircled{D})$ )



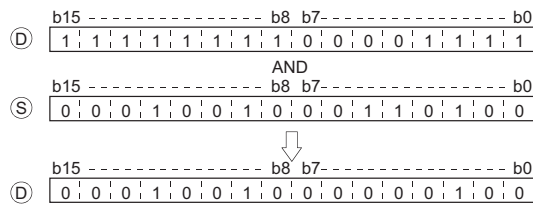
Ⓢ: Data for a logical product operation or the head number of the devices where the data is stored (BIN 16/32 bits)  
 Ⓣ: Head number of the devices where the logical product operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:AND		U:AND	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

## ★ Function

### WAND

- (1) A logical product operation is conducted for each bit of the 16-bit data of the device designated at Ⓣ and the 16-bit data of the device designated at Ⓢ, and the results are stored in the device designated at Ⓣ.



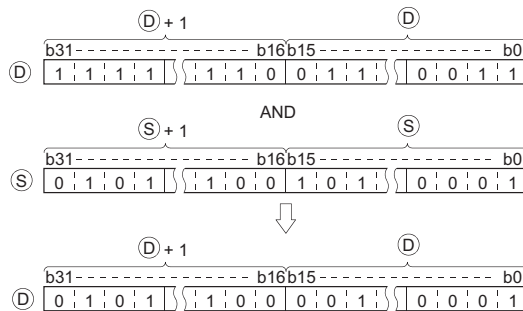
- (2) When bit devices are designated, the bit devices after the points designated as digits are regarded as "0" in the operation. (See Program Example (2))

7

7.1 Logical operation instructions  
 7.1.1 Logical products with 16-bit and 32-bit data (WAND(P),DAND(P))

## DAND

- (1) Conducts a logical product operation on each bit of the 32-bit data for the device designated by  $\text{S}_1$  and the 32-bit data for the device designated by  $\text{S}_2$ , and stores the results at the device designated by  $\text{D}$ .



- (2) When bit devices are designated, the bit devices below the points designated as digits are regarded as "0" in the operation. (See Program Example (2))



## Operation Error

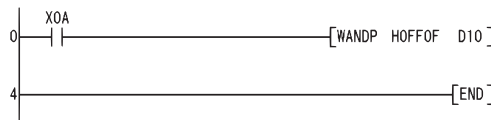
- (1) There are no operation errors associated with the WAND(P) or DAND(P) instruction.



## Program Example

- (1) The following program masks the digit in the 10s place of the 4-digit BCD value at D10 (second digit from the end) to 0 when XA is turned ON.

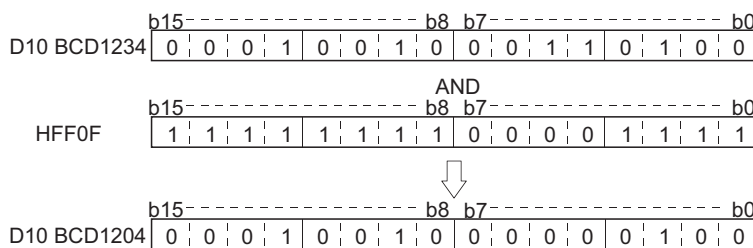
[Ladder Mode]



[List Mode]

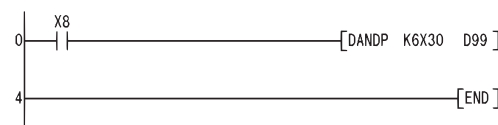
Step	Instruction	Device
0	LD	X0A
1	WANDP	HOFF0F D10
4	END	

[Operation]



- (2) The following program performs a logical product operation on the data at D99 and D100, and the 24-bit data between X30 and X47 when X8 is ON, and stores the results at D99 and D100.

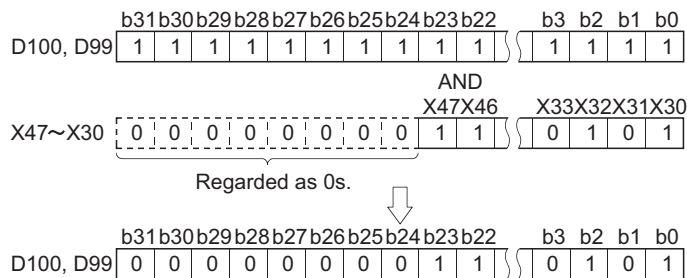
[Ladder Mode]



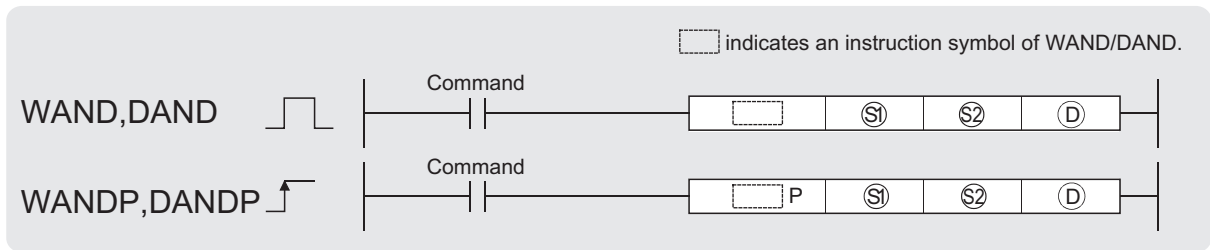
[List Mode]

Step	Instruction	Device
0	LD	X8
1	DANDP	K6X30 D99
4	END	

[Operation]



2 When three data are set ( $S_1 \wedge S_2 \rightarrow D$ ,  $(S_1+1, S_1) \wedge (S_2+1, S_2) \rightarrow (D+1, D)$ )



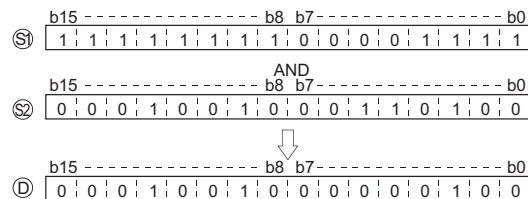
$S_1, S_2$  : Data for a logical product operation or the head number of the devices where the data is stored (BIN 16/32 bits)  
 $D$  : Head number of the devices where the logical product operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
$S_1$									
$S_2$									
$D$									

## ★ Function

### WAND

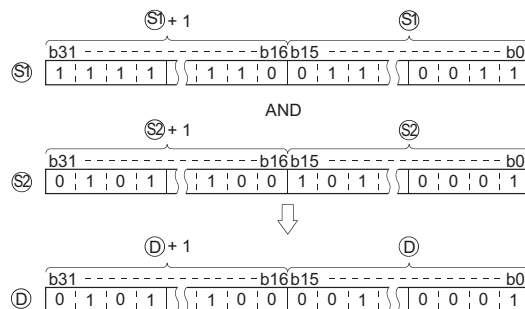
- (1) A logical product operation is conducted for each bit of the 16-bit data of the device designated at  $S_1$  and the 16-bit data of the device designated at  $S_2$ , and the results are stored in the device designated at  $D$ .



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation. (See Program Examples (1) and (2))

### DAND

- (1) Conducts a logical product operation on each bit of the 32-bit data for the device designated by  $S_1$  and the 32-bit data for the device designated by  $S_2$ , and stores the results at the device designated by  $D$ .



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded

as "0" in the operation. (See Program Example (3))

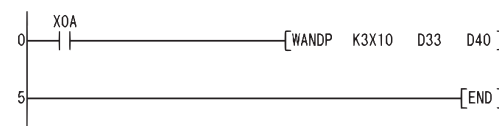
## ! Operation Error

- (1) There are no operation errors associated with the WAND(P) or DAND(P) instruction.

## Program Example

- (1) The following program performs a logical product operation on the data from X10 to X1B and the data at D33 when XA is ON, and stores the results at D40.

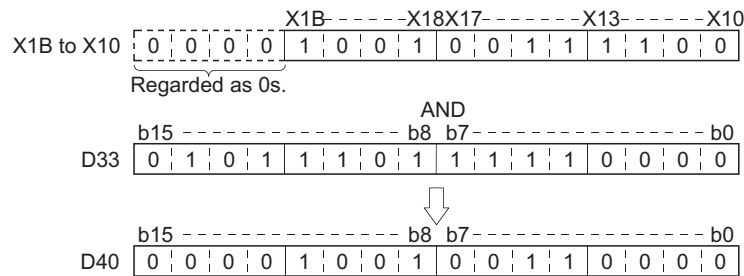
[Ladder Mode]



[List Mode]

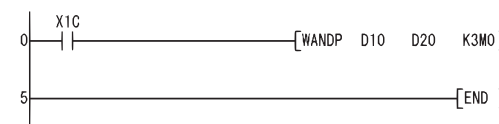
Step	Instruction	Device
0	LD	X0A
1	WANDP	K3X10 D33 D40
5	END	

[Operation]



- (2) The following program performs a logical product operation on the data at D10 and at D20 when X1C is ON, and stores the results from M0 to M11.

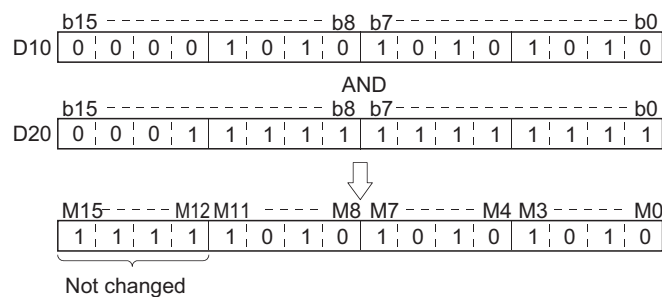
[Ladder Mode]



[List Mode]

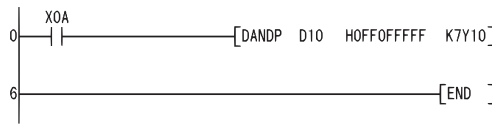
Step	Instruction	Device
0	LD	X1C
1	WANDP	D10 D20 K3M0
5	END	

[Operation]



- (3) The following program masks the digit in the hundred-thousands place of the 8-digit BCD value at D10 and D11 (sixth digit from the end) to 0 when XA is ON, and outputs the results to from Y10 to Y2B.

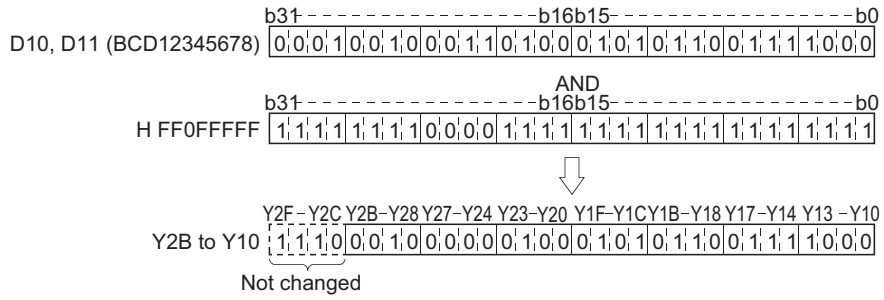
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0A
1	DANDP	D10 H0FF0FFFFF K7Y10
6	END	

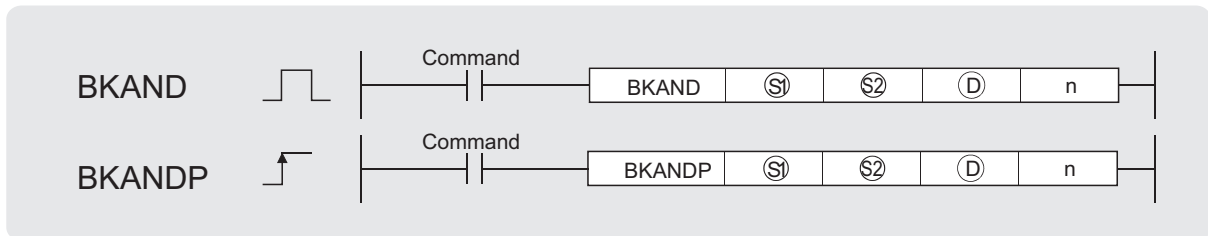
[Operation]





## 7.1.2 Block logical products (BKAND(P))

Basic High performance Process Redundant Universal



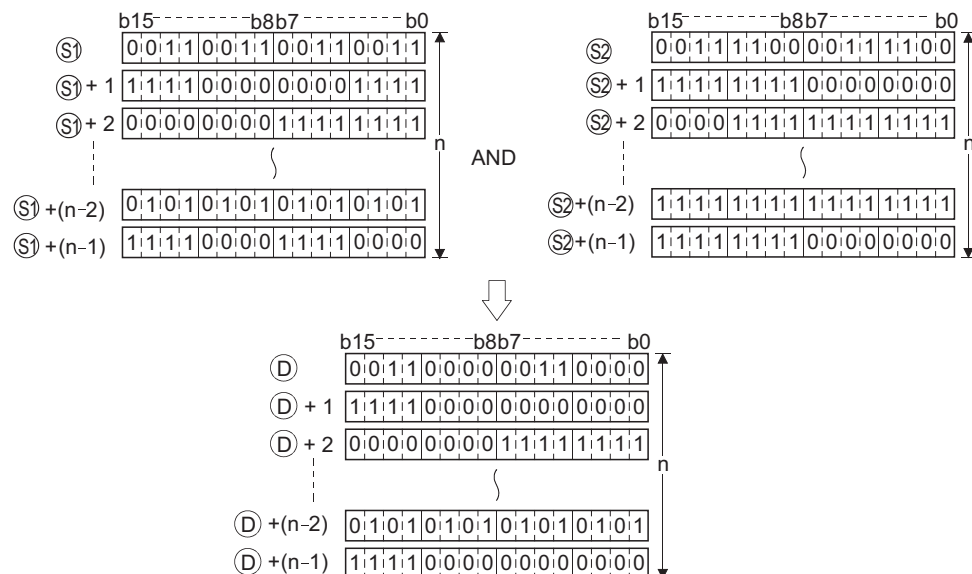
- Ⓢ<sub>1</sub>\*1 : Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
- Ⓢ<sub>2</sub>\*1 : Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
- ⓓ\*1 : Head number of the devices where the operation result will be stored (BIN 16 bits)
- n : Number of operation data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub> *1	—	○				—		—	—
Ⓢ <sub>2</sub> *1	—	○				—		○	—
ⓓ*1	—	○				—		—	—
n	○	○				○		○	—

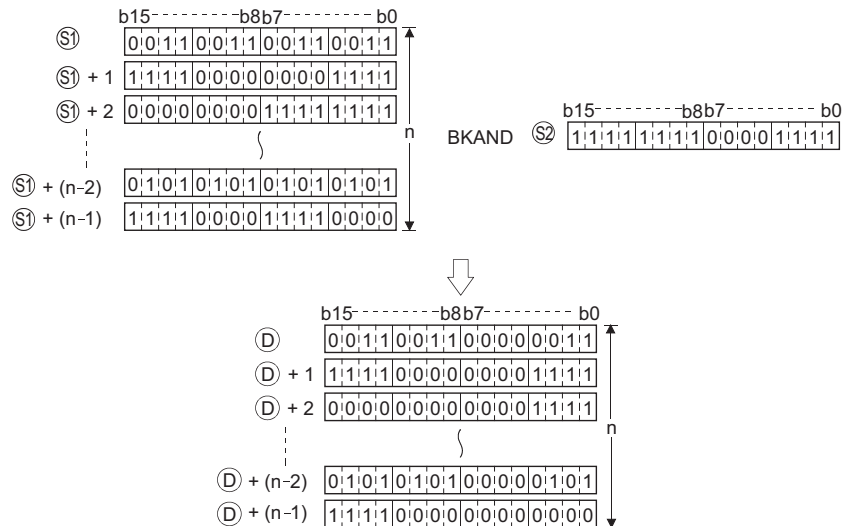
\*1: The same device number can be specified for Ⓢ<sub>1</sub> and ⓓ or Ⓢ<sub>2</sub> and ⓓ.

### ★ Function

- (1) Performs a logical product operation on the data located in the n points from the device designated by Ⓢ<sub>1</sub>, and the data located in the n points from the device designated by Ⓢ<sub>2</sub>, and stores the results into the area starting from the device designated by ⓓ.



(2) The constant designated by ② can be between -32768 and 32767 (BIN 16-bit data).



### Operation Error

(1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

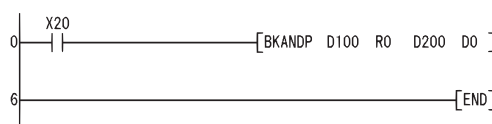
- The n-bit range from the ①, ②, or ③ device exceeds the range of that device. (Error code: 4101)
- The device range for n points starting from the device designated by ① overlaps with the device range for n points starting from the device designated by ③. (except when the same device is specified for ① and ③) (Error code: 4101)
- The device range for n points starting from the device designated by ② overlaps with the device range for n points starting from the device designated by ③. (except when the same device is specified for ② and ③) (Error code: 4101)



### Program Example

(1) The following program performs a logical product operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

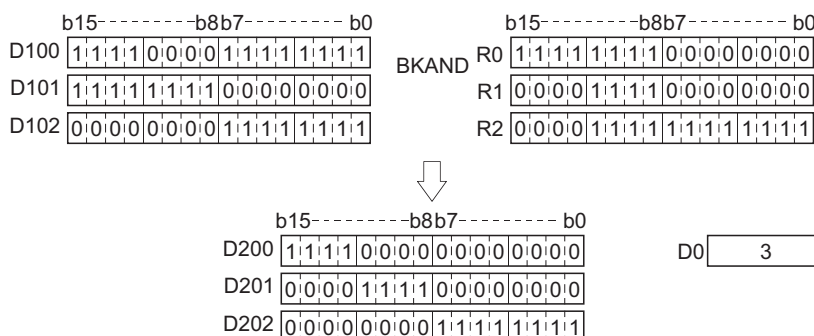
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKANDP	D100 R0 D200 D0
6	END	

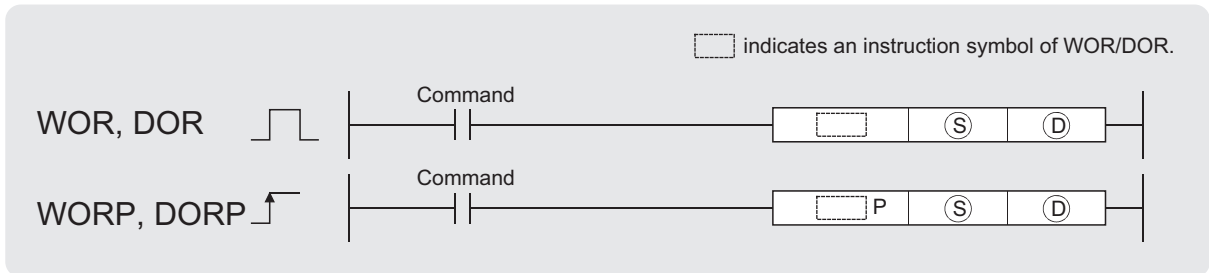
[Operation]



### 7.1.3 Logical sums of 16-bit and 32-bit data (WOR(P),DOR(P))

Basic High performance Process Redundant Universal

① When two data are set  $(\text{D} \vee \text{S}) \rightarrow \text{D}, (\text{D}+1, \text{D}) \vee (\text{S}+1, \text{S}) \rightarrow (\text{D}+1, \text{D})$



Ⓢ : Data for a logical sum operation or head number of the devices where the data is stored (BIN 16/32 bits)

ⓓ : Head number of the devices where the logical sum operation result will be stored (BIN 16/32 bits)

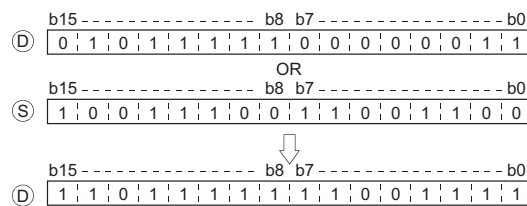
Setting Data	Internal Devices		R, ZR	JES		UGG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
ⓓ								—	—

7

## ★ Function

### WOR

- Conducts a logical sum operation on each bit of the 16-bit data of the device designated by ⓓ and the 16-bit data of the device designated by Ⓢ, and stores the results at the device designated by ⓓ.

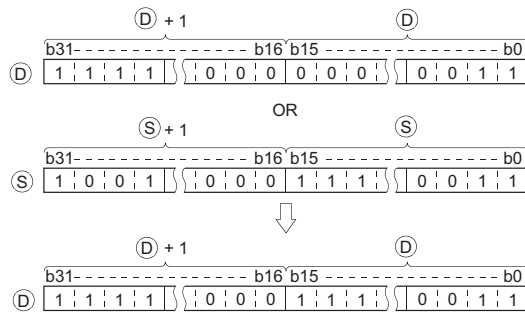


- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

7.1 Logical operation instructions  
7.1.3 Logical sums of 16-bit and 32-bit data (WOR(P),DOR(P))

## DOR

- (1) Conducts a logical sum operation on each bit of the 32-bit data of the device designated by (D) and the 32-bit data of the device designated by (S), and stores the results at the device designated by (D).



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.



## Operation Error

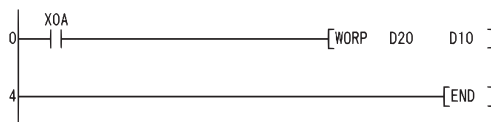
- (1) There are no operation errors associated with the WOR(P) or DOR(P) instructions.



## Program Example

- (1) The following program performs a logical sum operation on the data at D10 and D20 when XA is turned ON, and stores the results at D10.

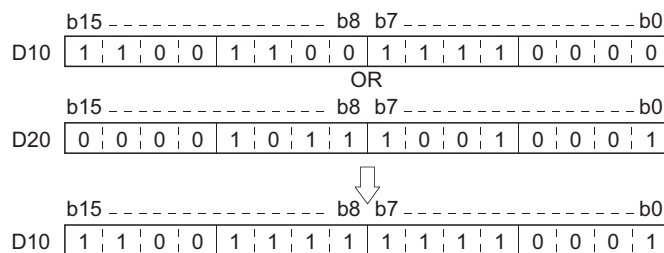
[Ladder Mode]



[List Mode]

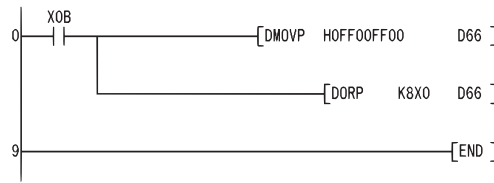
Step	Instruction	Device
0	LD	XOA
1	WORP	D20 D10
4	END	

[Operation]



- (2) The following program performs a logical sum operation on the 32-bit data from X0 to X1F, and on the hexadecimal value FF00FF00<sub>H</sub> when XB is turned ON, and stores the results at D66 and D67.

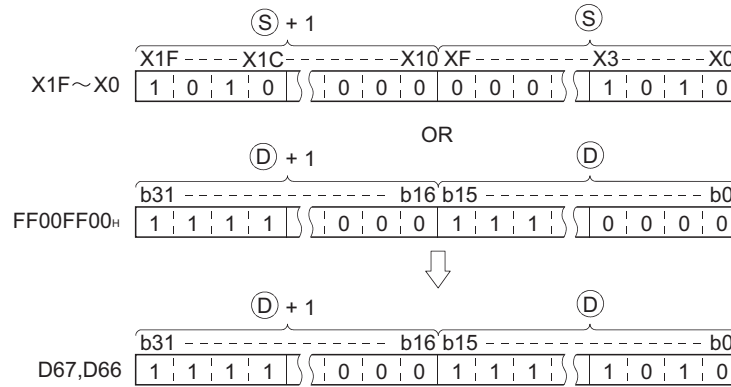
[Ladder Mode]



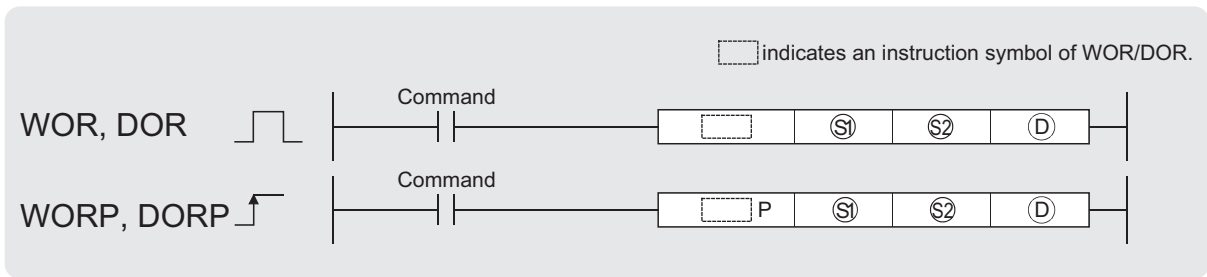
[List Mode]

Step	Instruction	Device
0	LD	X0B
1	DMOVP	H0FF00FF00 D66
4	DORP	K8X0 D66
9	END	

[Operation]



2 When three data are set ( $S_1 \vee S_2 \rightarrow D, (S_1+1, S_1) \vee (S_2+1, S_2) \rightarrow (D+1, D)$ )



$S_1, S_2$  : Data for a logical sum operation or head number of the devices where the data is stored (BIN 16/32 bits)

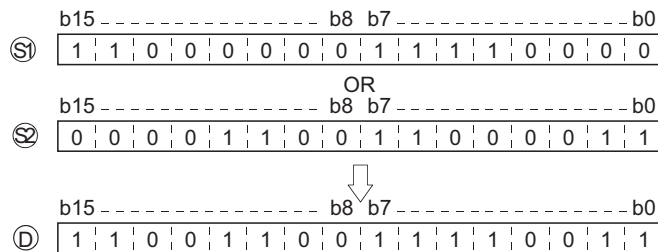
$D$  : Head number of the devices where the logical sum operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
$S_1$					○			○	—
$S_2$					○			○	—
$D$					○			—	—

## ★ Function

### WOR

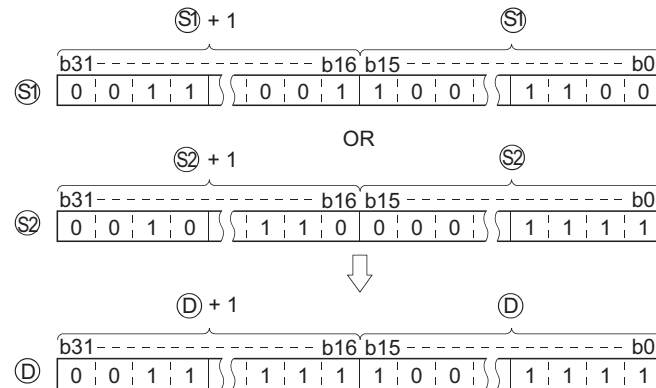
- Conducts a logical sum operation on each bit of the 16-bit data of the device designated by  $S_1$  and the 16-bit data of the device designated by  $S_2$ , and stores the results at the device designated by  $D$ .



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation. (See Program Example (1))

## DOR

- (1) Conducts a logical sum operation on each bit of the 32-bit data of the device designated by  $\textcircled{S1}$  and the 32-bit data of the device designated by  $\textcircled{S2}$ , and stores the results at the device designated by  $\textcircled{D}$ .



- (2) When bit devices are designated, the bit devices below the points designated as digits are regarded as "0" in the operation. (See Program Example (2))

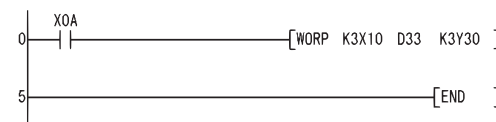
## ! Operation Error

- (1) There are no operation errors associated with the WOR(P) or DOR(P) instructions.

## Program Example

- (1) The following program performs a logical sum operation on the data from X10 to X1B, and the data at D33, and stores the result at Y30 to Y3B when XA is ON.

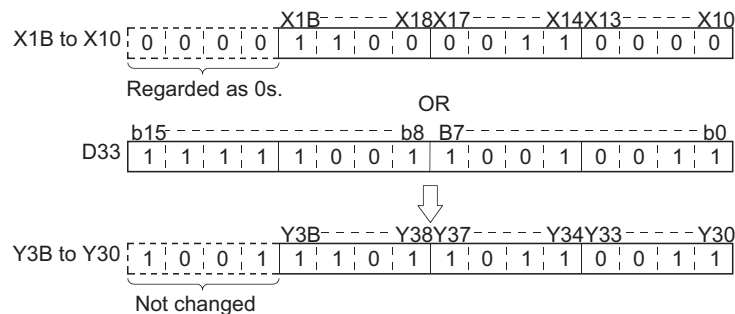
[Ladder Mode]



[List Mode]

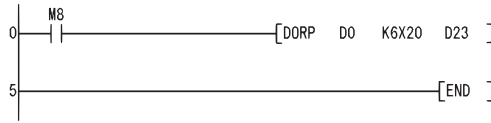
Step	Instruction	Device
0	LD	XOA
1	WORP	K3X10 D33 K3Y30
5	END	

[Operation]



(2) The following program performs a logical sum operation on the 32-bit data at D0 and D1, and the 24-bit data from X20 to X37, and stores the results at D23 and D24 when M8 is ON.

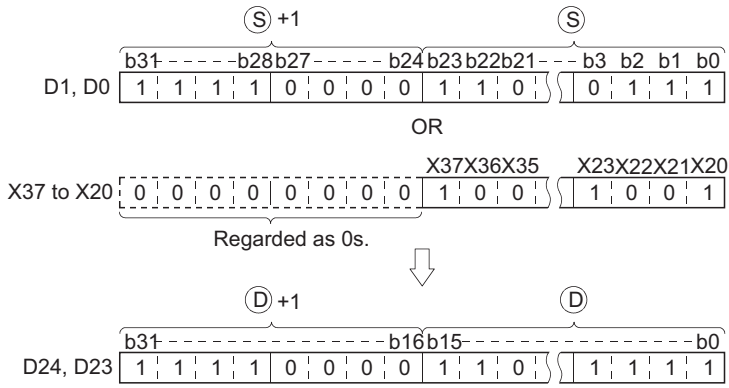
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M8
1	DORP	D0 K6X20 D23
5	END	

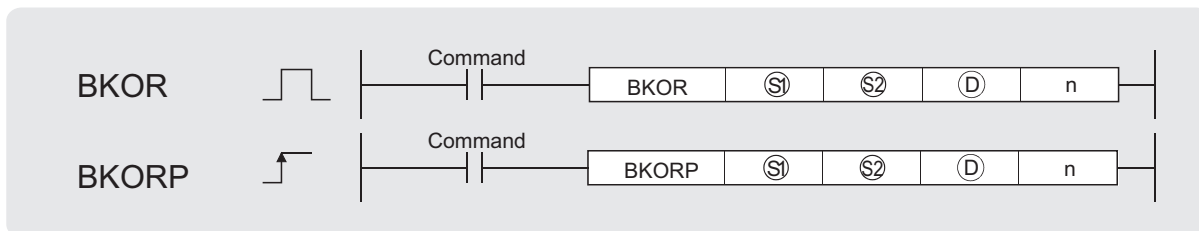
[Operation]





## 7.1.4 Block logical sum operations (BKOR(P))

Basic High performance Process Redundant Universal



Ⓢ<sub>1</sub>\*1 : Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)

Ⓢ<sub>2</sub>\*1 : Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)

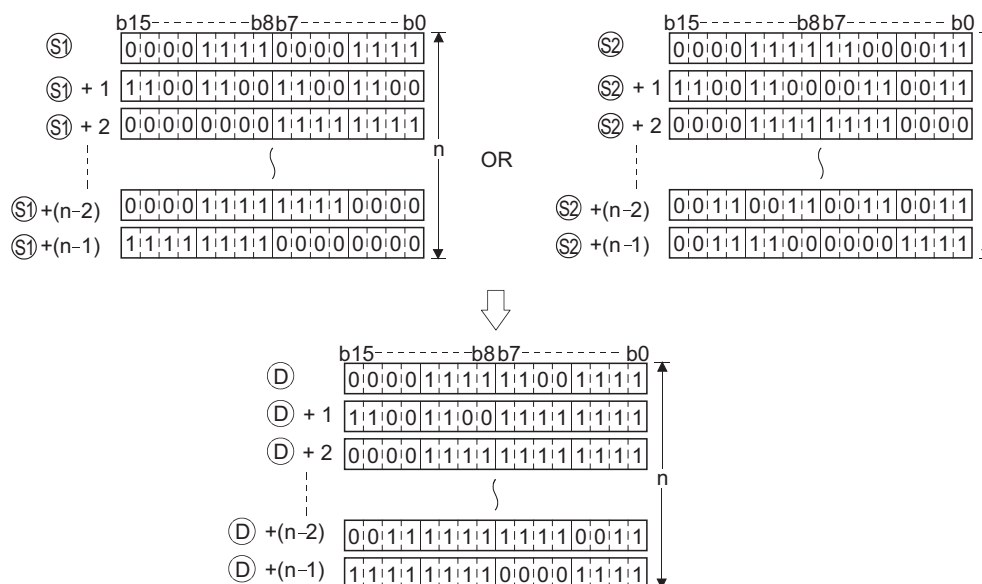
ⓓ\*1 : Head number of the devices where the operation result will be stored (BIN 16 bits)  
n : Number of operation data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub> *1	—	○				—		—	—
Ⓢ <sub>2</sub> *1	—	○				—		○	—
ⓓ*1	—	○				—		—	—
n	○	○				○		○	—

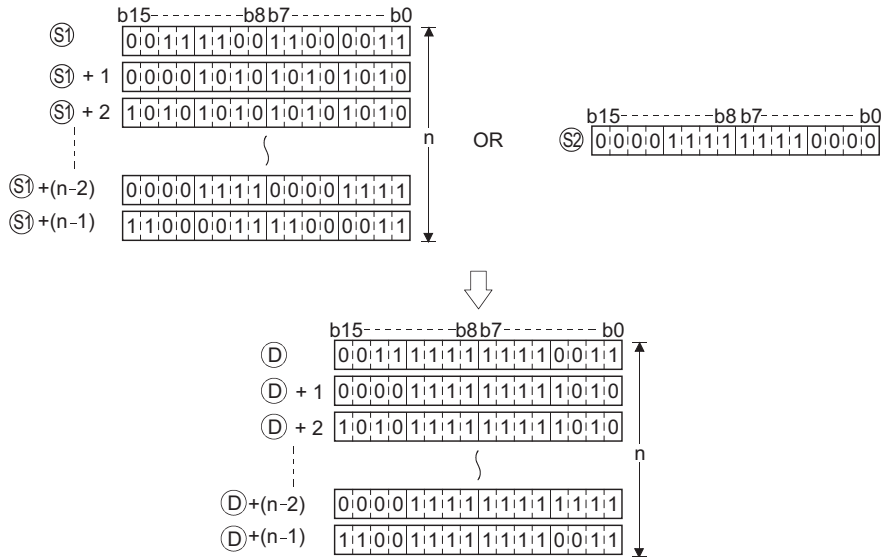
\*1: The same device number can be specified for Ⓢ<sub>1</sub> and ⓓ or Ⓢ<sub>2</sub> and ⓓ.

### ★ Function

- (1) Performs a logical sum operation on the data located in the n points from the device designated by Ⓢ<sub>1</sub>, and the data located in the n points from the device designated by Ⓢ<sub>2</sub>, and stores the results into the area starting from the device designated by ⓓ.



(2) The constant designated by ② can be between -32768 and 32767 (BIN 16-bit data).



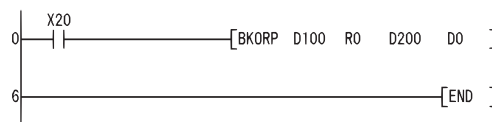
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The n-bit range from the ①, ②, or ③ device exceeds the range of that device. (Error code: 4101)
  - The device range for n points starting from the device designated by ① overlaps with the device range for n points starting from the device designated by ③. (except when the same device is specified for ① and ③) (Error code: 4101)
  - The device range for n points starting from the device designated by ② overlaps with the device range for n points starting from the device designated by ③. (except when the same device is specified for ② and ③) (Error code: 4101)

## Program Example

(1) The following program performs a logical sum operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

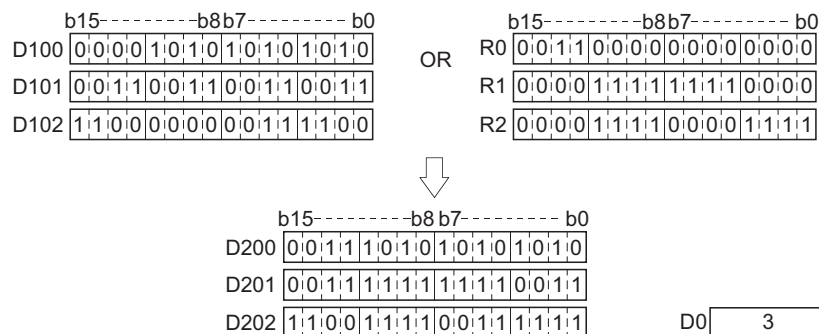
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKORP	D100 R0 D200 D0
6	END	

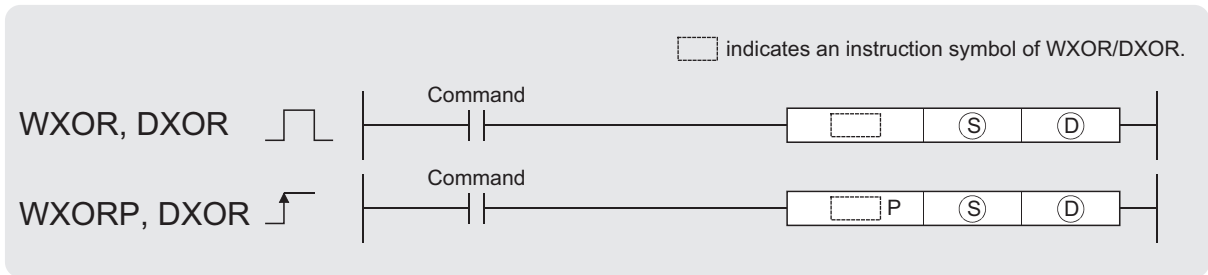
[Operation]



# 7.1.5 16-bit and 32-bit exclusive OR operations (WXOR(P),DXOR(P))

Basic High performance Process Redundant Universal

1 When two data are set (D) ∨ (S) → D, (D+1, D) ∨ (S+1, S) → (D+1, D))



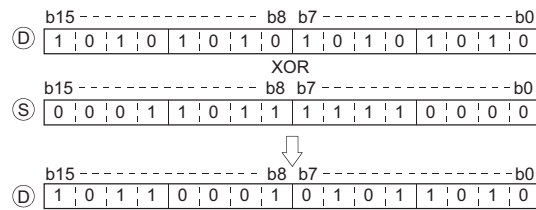
(S): Data for an exclusive OR operation or head number of the devices where the data is stored (BIN 16/32 bits)  
 (D): Head number of the devices where the exclusive OR operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
(S)								○	—
(D)								—	—

## ★ Function

### WXOR

- Conducts an exclusive OR operation on each bit of the 16-bit data of the device designated by (D) and the 16-bit data of the device designated by (S), and stores the results at the device designated by (D).



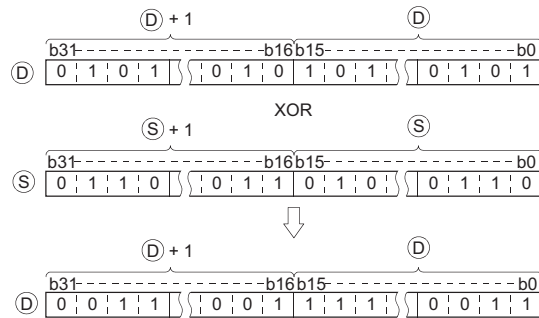
- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

7

7.1 Logical operation instructions  
7.1.5 16-bit and 32-bit exclusive OR operations (WXOR(P),DXOR(P))

## DXOR

- (1) Conducts an exclusive OR operation on each bit of the 32-bit data of the device designated by (D) and the 32-bit data of the device designated by (S), and stores the results at the device designated by (D).



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.



## Operation Error

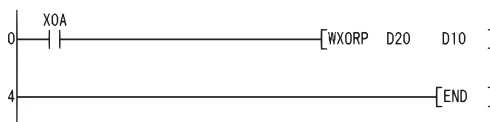
- (1) There are no operation errors associated with the WXOR(P) or DXOR(P) instructions.



## Program Example

- (1) The following program performs an exclusive OR operation on the data at D10 and D20 when XA is ON, and stores the result at D10.

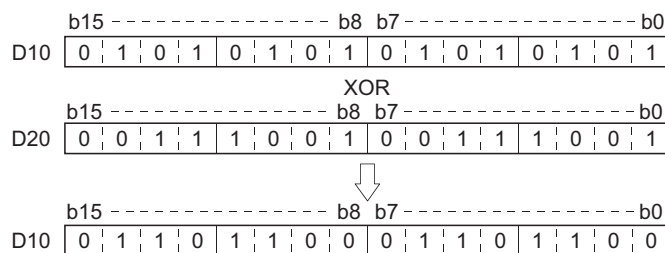
[Ladder Mode]



[List Mode]

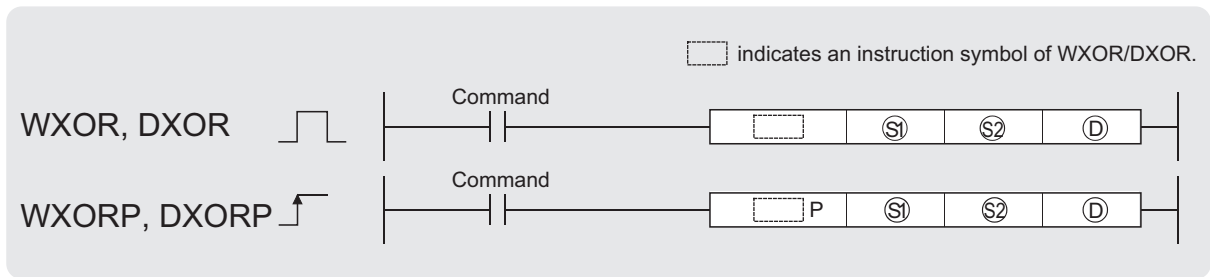
Step	Instruction	Device
0	LD	XA
1	WXORP	D20 D10
4	END	

[Operation]





2 When three data are set (S1) ∨ (S2) → (D) (S1+1, S1) ∨ (S2+1, S2) → (D+1, D))



S1, S2 : Data for an exclusive OR operation or head number of the devices where the data is stored (BIN 16/32 bits)

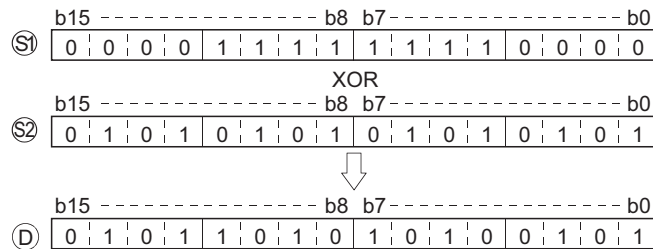
D : Head number of the devices where the exclusive OR operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	JMO		UAGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
S1					○			○	—
S2					○			○	—
D					○			—	—

## ★ Function

### WXOR

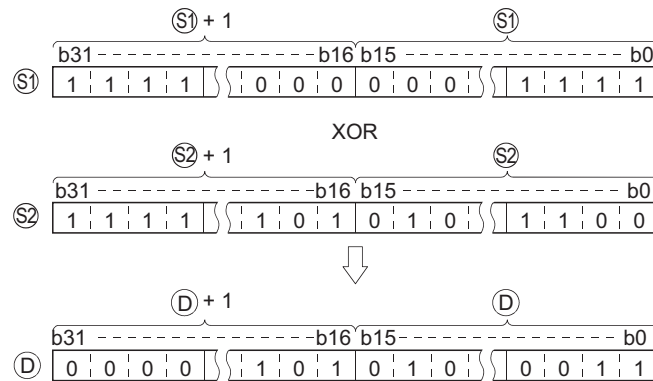
- Conducts an exclusive OR operation on each bit of the 16-bit data of the device designated by S1 and the 16-bit data of the device designated by S2, and stores the results at the device designated by D.



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation. (See Program Example (1))

## DXOR

- (1) Conducts an exclusive OR operation on each bit of the 32-bit data of the device designated by  $\text{S1}$  and the 32-bit data of the device designated by  $\text{S2}$ , and stores the results at the device designated by  $\text{D}$ .



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

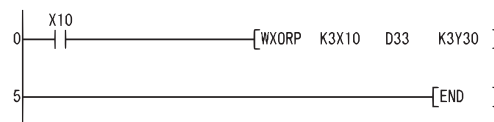
## ! Operation Error

- (1) There are no operation errors associated with the WXOR(P) or DXOR(P) instructions.

## Program Example

- (1) The following program conducts an exclusive OR operation on the data from X10 to X1B and the data at D33 when X10 is ON, and outputs the result to Y30 to Y3B.

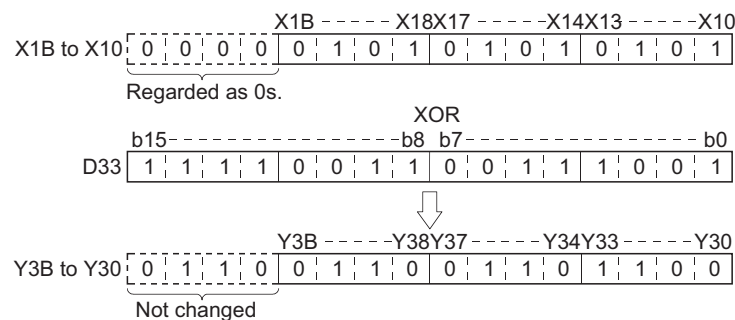
[Ladder Mode]



[List Mode]

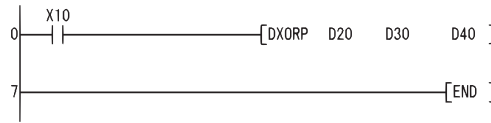
Step	Instruction	Device
0	LD	X10
1	WXORP	K3X10 D33 K3Y30
5	END	

[Operation]



(2) The following program conducts an exclusive OR operation on the data at D20 and D21, and the data at D30 and D31 when X10 is turned ON, and stores the results at D40 and D41.

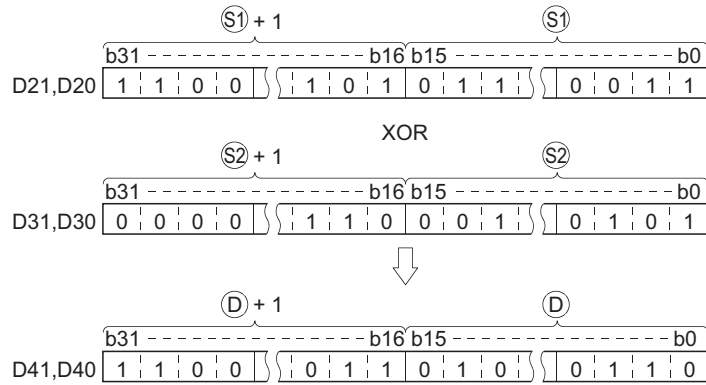
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	DXORP	D20 D30 D40
7	END	

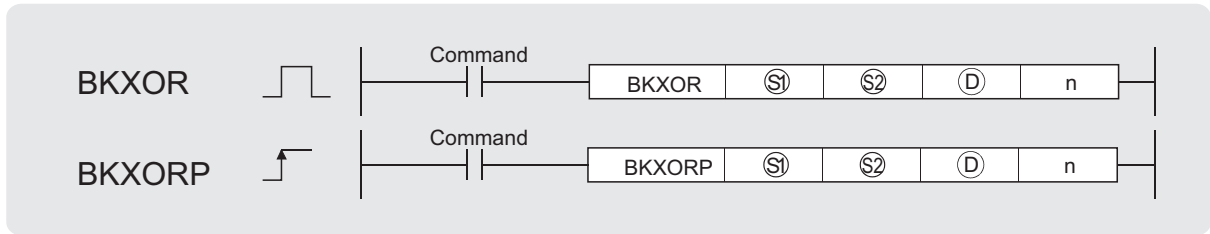
[Operation]





# 7.1.6 Block exclusive OR operations (BKXOR(P))

Basic High performance Process Redundant Universal



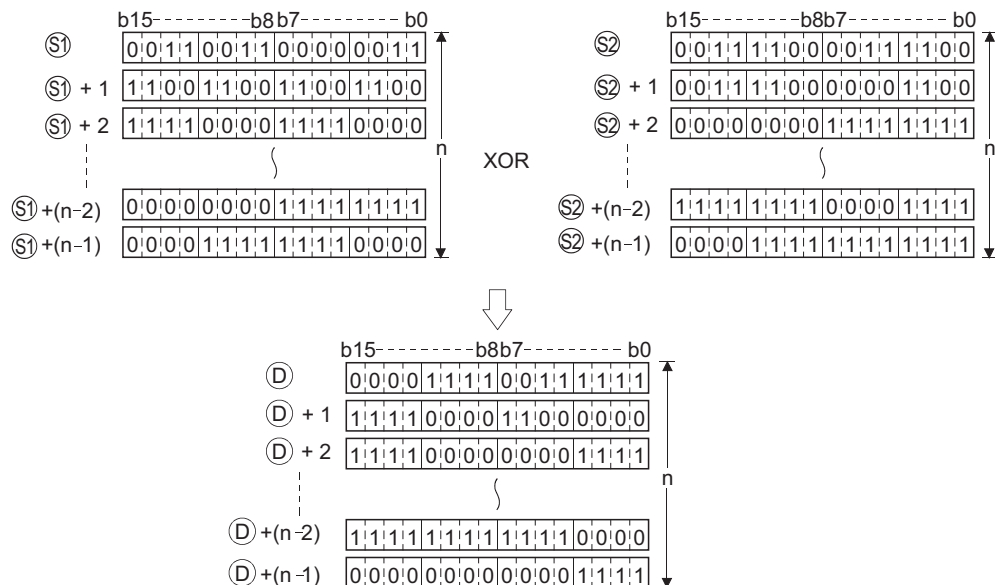
- Ⓢ<sub>1</sub>\*1 : Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
- Ⓢ<sub>2</sub>\*1 : Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
- ⓓ\*1 : Head number of the devices where the operation result will be stored (BIN 16 bits)
- n : Number of operation data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub> *1	—	○				—		—	—
Ⓢ <sub>2</sub> *1	—	○				—		○	—
ⓓ*1	—	○				—		—	—
n	○	○				○		○	—

\*1: The same device number can be specified for Ⓢ<sub>1</sub> and ⓓ or Ⓢ<sub>2</sub> and ⓓ.

## ★ Function

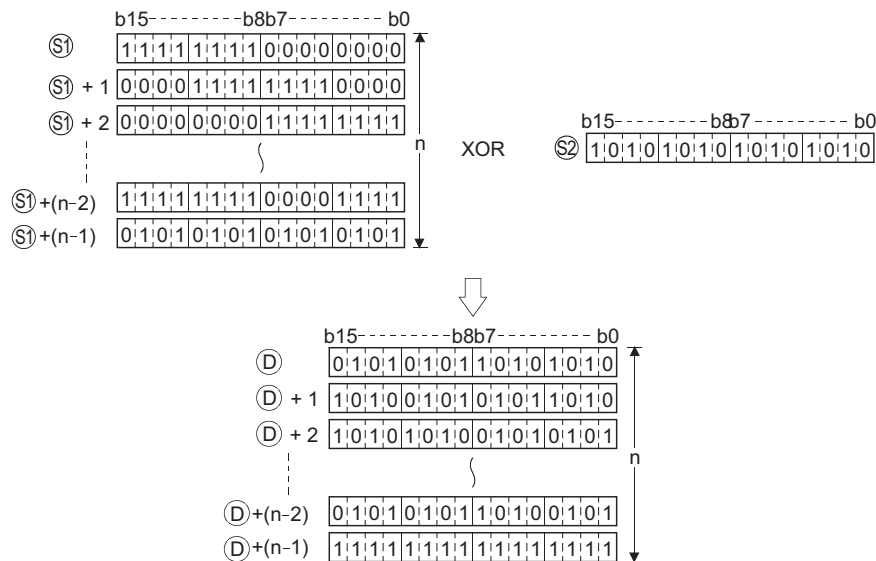
- (1) Performs an exclusive OR operation on the data located in the n points from the device designated by Ⓢ<sub>1</sub>, and the data located in the n points from the device designated by Ⓢ<sub>2</sub>, and stores the results into the area starting from the device designated by ⓓ.



7.1 Logical operation instructions  
7.1.6 Block exclusive OR operations (BKXOR(P))

7

(2) The constant designated by  $\textcircled{S2}$  can be between  $-32768$  and  $32767$  (BIN 16-bit data).



## Operation Error

(1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

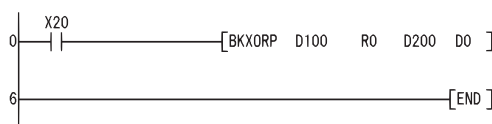
- The n-bit range from the  $\textcircled{S1}$ ,  $\textcircled{S2}$ , or  $\textcircled{D}$  device exceeds the range of that device. (Error code: 4101)
- The device range for n points starting from the device designated by  $\textcircled{S1}$  overlaps with the device range for n points starting from the device designated by  $\textcircled{D}$ . (except when the same device is specified for  $\textcircled{S1}$  and  $\textcircled{D}$ ) (Error code: 4101)
- The device range for n points starting from the device designated by  $\textcircled{S2}$  overlaps with the device range for n points starting from the device designated by  $\textcircled{D}$ . (except when the same device is specified for  $\textcircled{S2}$  and  $\textcircled{D}$ ) (Error code: 4101)



## Program Example

(1) The following program performs an exclusive OR operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

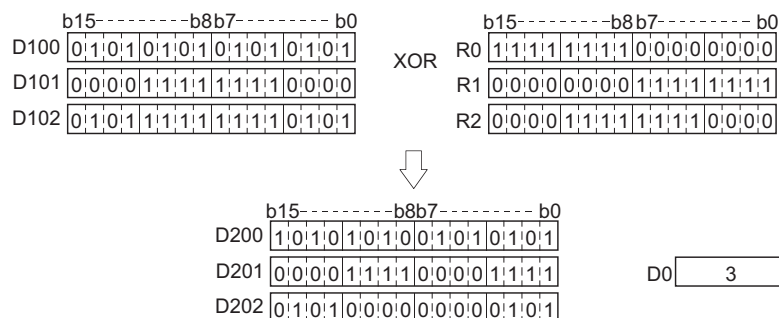
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKXORP	D100 R0 D200 D0
6	END	

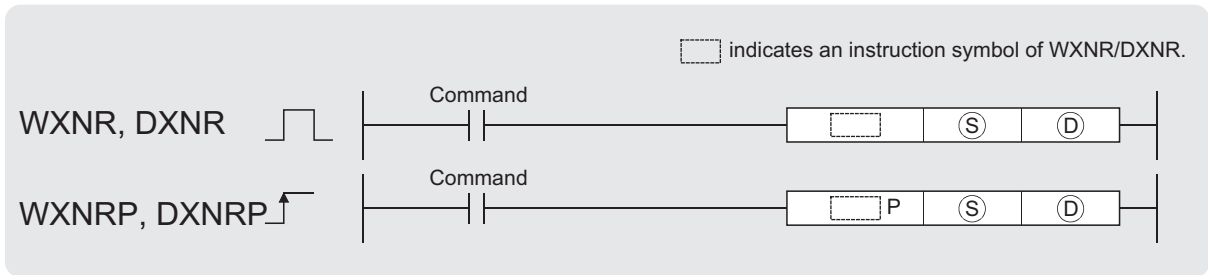
[Operation]



# 7.1.7 16-bit and 32-bit data exclusive NOR operations (WXNR(P),DXNR(P))

Basic High performance Process Redundant Universal

1 When two data are set  $(\textcircled{D} \vee \textcircled{S}) \rightarrow \textcircled{D}, (\textcircled{D}+1, \textcircled{D}) \vee (\textcircled{S}+1, \textcircled{S}) \rightarrow (\textcircled{D}+1, \textcircled{D})$



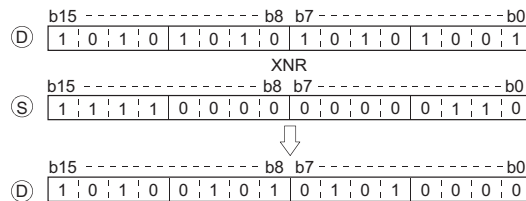
Ⓢ: Data for an exclusive NOR operation or head number of the devices where the data is stored (BIN 16/32 bits)  
 Ⓣ: Head number of the devices where the exclusive NOR operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

## ★ Function

### WXNR

- Conducts an exclusive NOR operation on the 16-bit data of the device designated by Ⓣ and the 16-bit data of the device designated by Ⓢ, and stores the results at the device designated by Ⓣ.



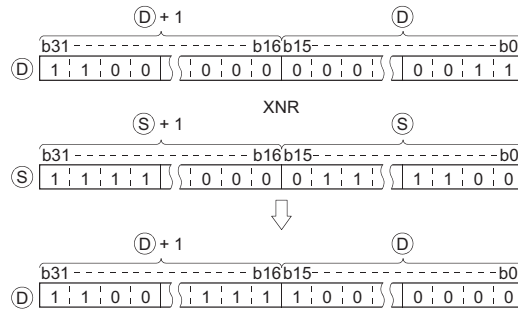
- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

7

7.1 Logical operation instructions  
 7.1.7 16-bit and 32-bit data exclusive NOR operations (WXNR(P),DXNR(P))

## DXNR

- (1) Conducts an exclusive NOR operation on the 32-bit data of the device designated by  $\text{D} + 1$  and the 32-bit data of the device designated by  $\text{D}$ , and stores the results at the device designated by  $\text{D}$ .



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.



## Operation Error

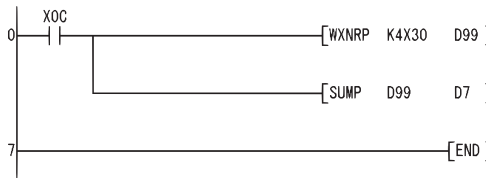
- (1) There are no operation errors associated with the WXNR(P) or DXNR(P) instruction.



## Program Example

- (1) The following program compares the bit patterns of the 16-bit data located from X30 to X3F with the bit patterns of the 16-bit data at D99 when XC is ON, and stores the number of identical bit patterns at D7.

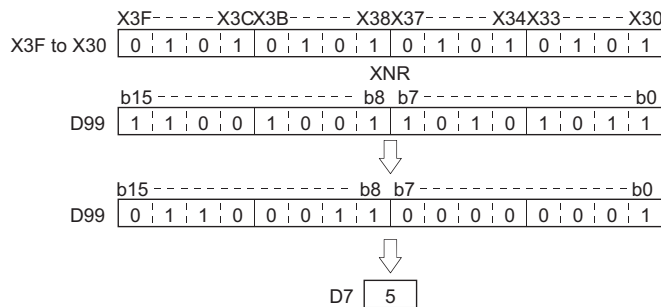
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0C
1	WXNRP	K4X30 D99
4	SUMP	D99 D7
7	END	

[Operation]



- (2) The following program compares the bit patterns of the 32-bit data located from X20 to X3F with the bit patterns of the data at D16 and D17 when X6 is ON, and stores the number of identical bit patterns at D18.

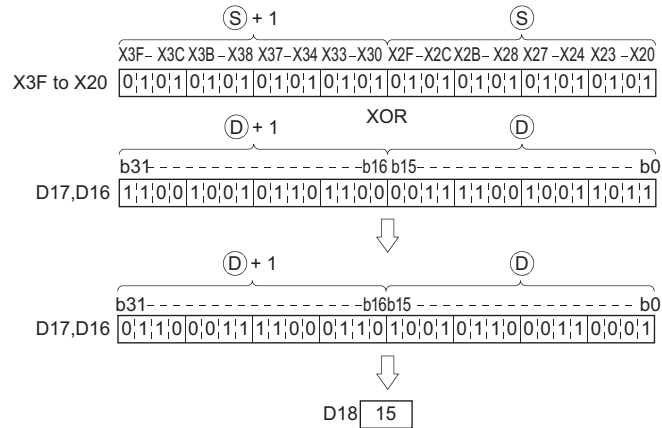
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X6
1	DXNRP	K8X20 D16
6	DSUMP	D16 D18
9	END	

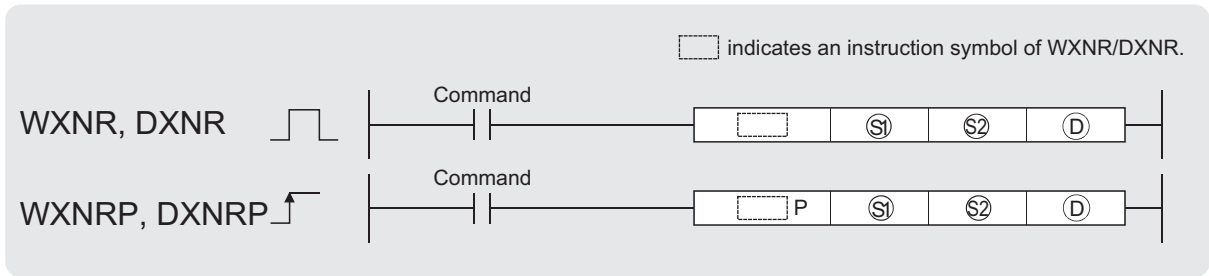
[Operation]



**Remark**

See 7.5.2 for more information on the SUMP/DSUMP instructions.

2 When three data are set ( $\text{S1} \vee \text{S2} \rightarrow \text{D}$ ,  $(\text{S1}+1, \text{S1}) \vee (\text{S2}+1, \text{S2}) \rightarrow (\text{D}+1, \text{D})$ )



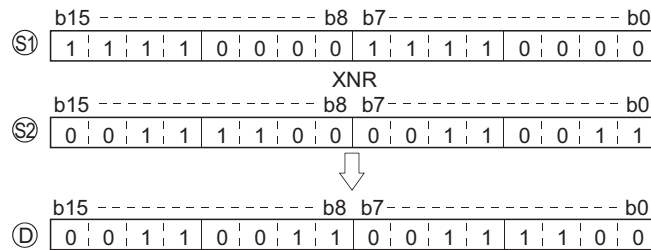
S1, S2 : Data for an exclusive NOR operation or head number of the devices where the data is stored (BIN 16/32 bits)  
 D : Head number of the devices where the exclusive NOR operation result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
S1								○	—
S2								○	—
D								—	—

## ★ Function

### WXNR

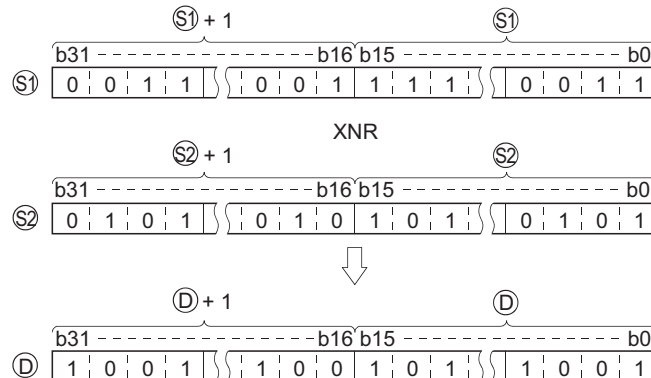
- Conducts an exclusive NOR operation on the 16-bit data of the device designated by S1 and the 16-bit data of the device designated by S2, and stores the results at the device designated by D.



- For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

## DXNR

- (1) Conducts an exclusive NOR operation on the 32-bit data of the device designated by  $\textcircled{S1}$  and the 32-bit data of the device designated by  $\textcircled{S2}$ , and stores the results at the device designated by  $\textcircled{D}$ .



- (2) For bit devices, the bit devices after the points designated by digit specification are regarded as "0" in the operation.

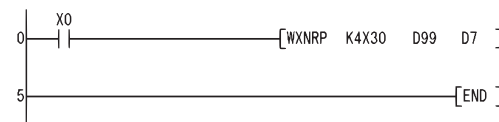
## ! Operation Error

- (1) There are no operation errors associated with the WXNR(P) or DXNR(P) instructions.

## Program Example

- (1) The following program performs an exclusive NOR operation on the 16-bit data from X30 to X3F and the data at D99 when X0 is turned ON, and stores the results to D7.

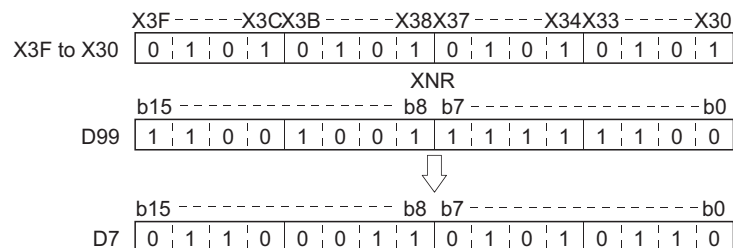
[Ladder Mode]



[List Mode]

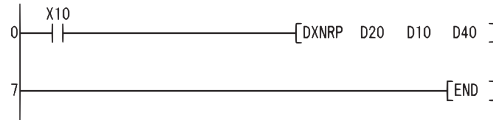
Step	Instruction	Device
0	LD	X0
1	WXNRP	K4X30 D99 D7
5	END	

[Operation]



- (2) The following program performs an exclusive NOR operation on the 32-bit data at D20 and D21 and the data at D10 and D11 when X10 is turned ON, and stores the result to D40 and D41.

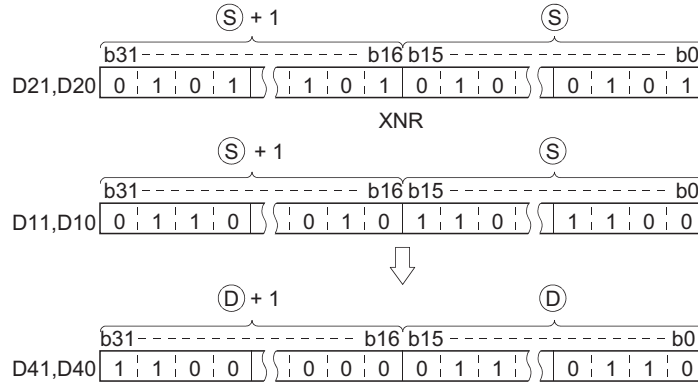
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	DXNRP	D20 D10 D40
7	END	

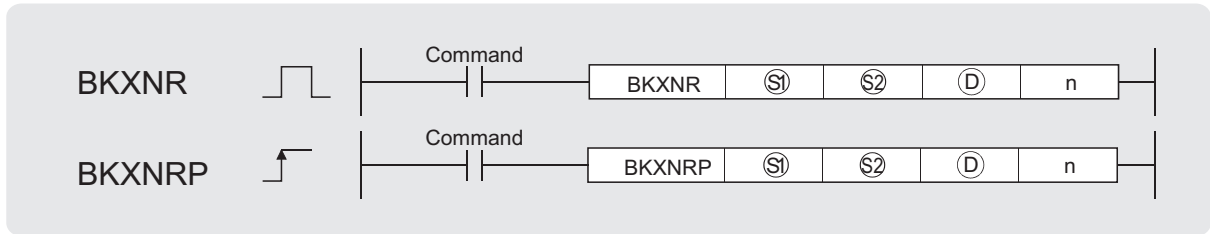
[Operation]





# 7.1.8 Block exclusive NOR operations (BKXNR(P))

Basic High performance Process Redundant Universal



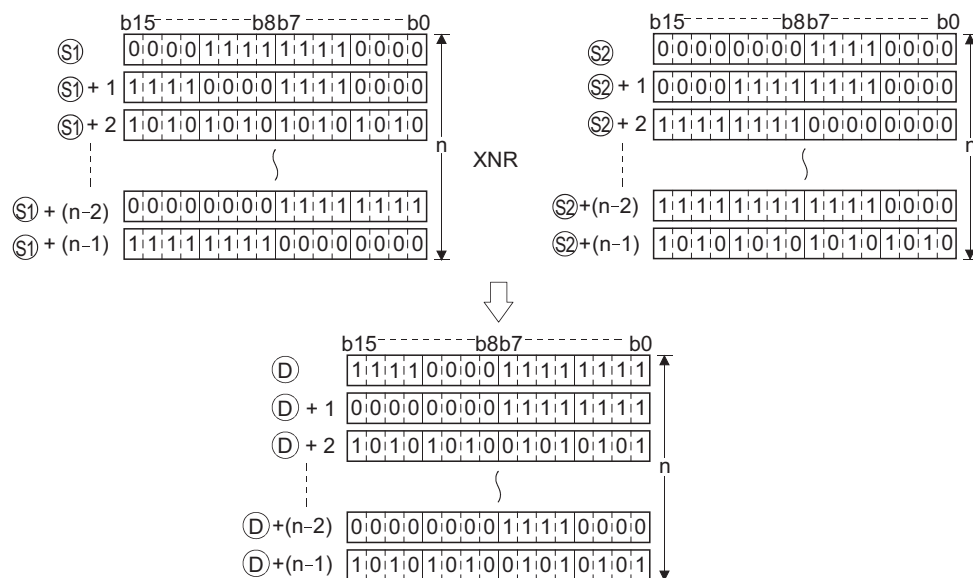
- Ⓢ1 \*1 : Head number of the devices where data on which a logical operation will be conducted is stored (BIN 16 bits)
- Ⓢ2 \*1 : Data for a logical operation or head number of the devices where the data for the logical operation is stored (BIN 16 bits)
- Ⓧ \*1 : Head number of the devices where the operation result will be stored (BIN 16 bits)
- n : Number of operation data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JESD		U <sub>0</sub> G <sub>0</sub>	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1 *1	—	○				—		—	—
Ⓢ2 *1	—	○				—		○	—
Ⓧ *1	—	○				—		—	—
n	○	○				○		○	—

\*1: The same device number can be specified for Ⓢ1 and Ⓧ or Ⓢ2 and Ⓧ.

## ★ Function

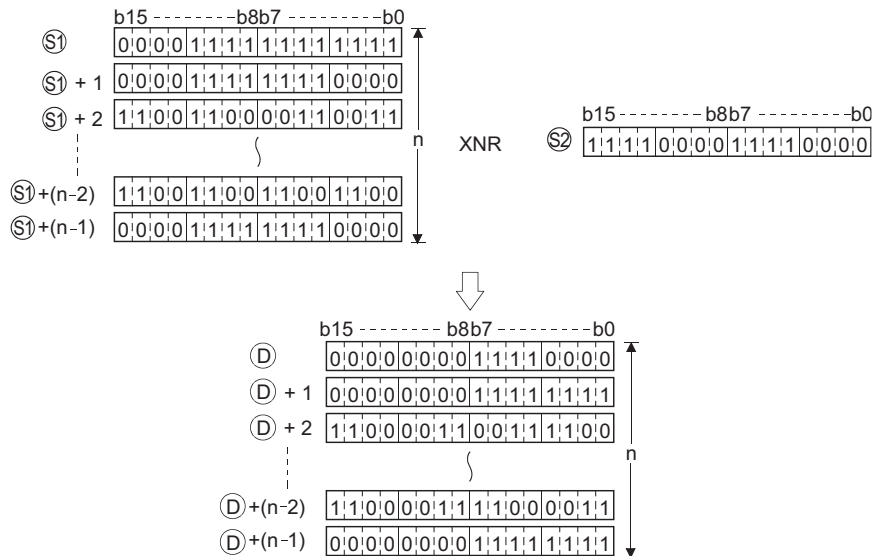
- (1) Performs an exclusive NOR operation on the data located in the n points from the device designated by Ⓢ1, and the data located in the n points from the device designated by Ⓢ2, and stores the results into the area starting from the device designated by Ⓧ.



7

7.1 Logical operation instructions  
7.1.8 Block exclusive NOR operations (BKXNR(P))

(2) The constant designated by ② can be between -32768 and 32767 (BIN 16-bit data).



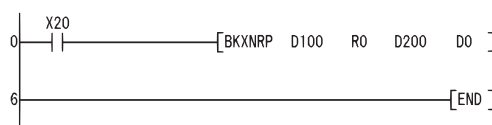
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The n-bit range from the ①, ②, or ③ device exceeds the range of that device. (Error code: 4101)
  - The device range for n points starting from the device designated by ① overlaps with the device range for n points starting from the device designated by ③. (except when the same device is specified for ① and ③) (Error code: 4101)
  - The device range for n points starting from the device designated by ② overlaps with the device range for n points starting from the device designated by ③. (except when the same device is specified for ② and ③) (Error code: 4101)

## Program Example

(1) The following program performs an exclusive NOR operation on the data stored at D100 to D102 and the data stored at R0 to R2 when X20 is turned ON, and stores the operation result into the area starting from D200.

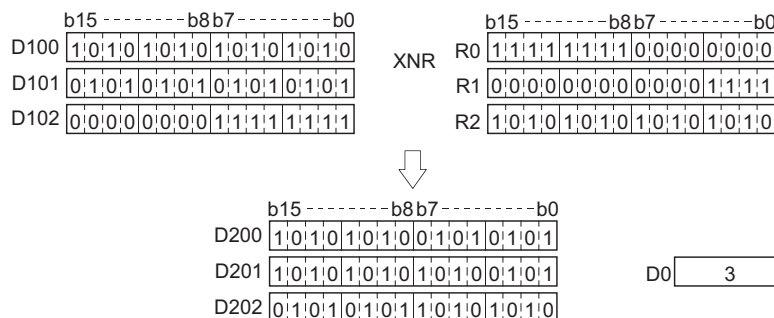
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKXNRP	D100 R0 D200 D0
6	END	

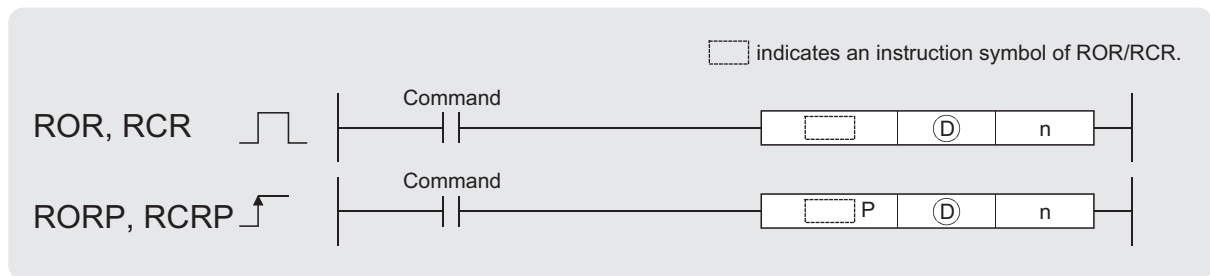
[Operation]



## 7.2 Rotation instruction

### 7.2.1 Right rotation of 16-bit data (ROR(P),RCR(P))

Basic High performance Process Redundant Universal



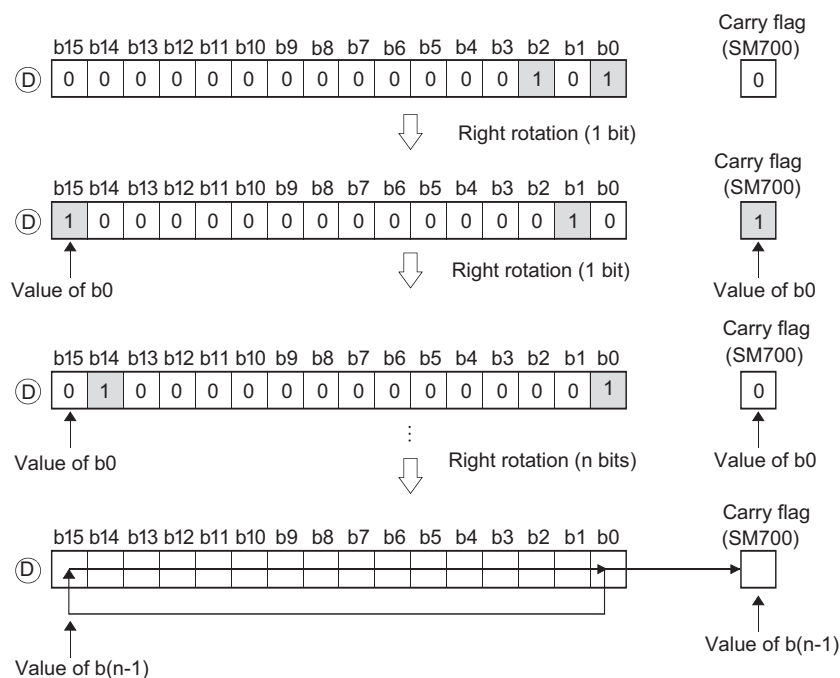
Ⓓ : Head number of the devices to rotate (BIN 16 bits)  
n : Number of rotations (0 to 15) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JESD		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓓ								—	—
n								○	—

## ★ Function

### ROR

- (1) Rotates 16-bit data of the device designated by Ⓓ, not including the carry flag, n-bits to the right. The carry flag is ON or OFF depending on the status prior to the execution of the ROR instruction.



- (2) When a bit device is designated for  $\textcircled{D}$ , a rotation is performed within the device range specified by digit specification.

The number of bits by which a rotation is carried out is the remainder of  $n/(\text{specified number of bits})$ .

For example, when  $n = 15$  and  $(\text{specified number of bits}) = 12$  bits, the remainder of  $15/12 = 1$  is "3", and the data is rotated 3 bits.

- (3) Specify any of 0 to 15 as  $n$ .

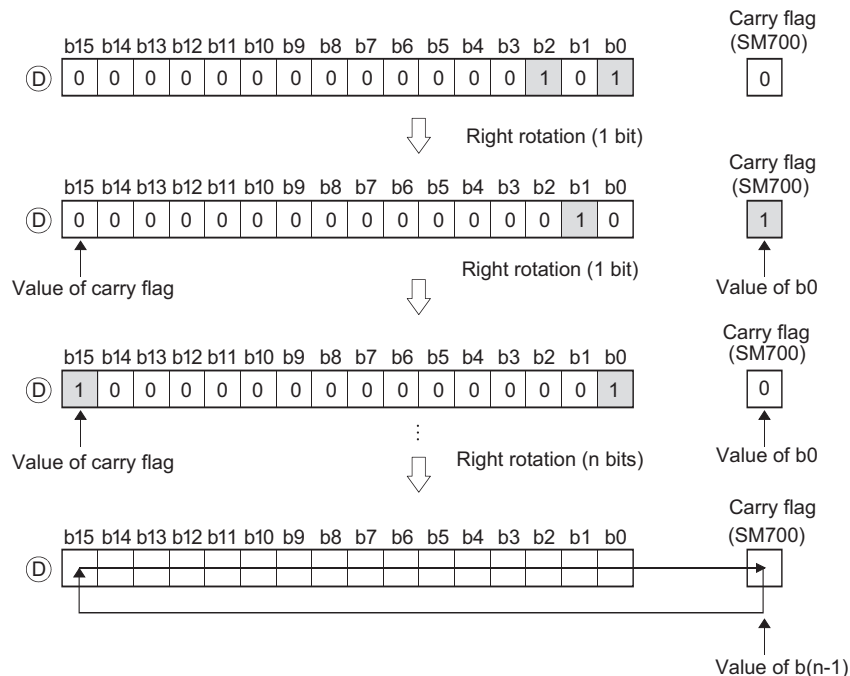
If the value specified as  $n$  is 16 or greater, the remainder of  $n / 16$  is used for rotation.

For example, when  $n = 18$ , the contents are rotated two bits to the right since the remainder of  $18 / 16 = 1$  is "2".

## RCR

- (1) Rotates 16-bit data of the device designated by  $\textcircled{D}$ , including the carry flag,  $n$ -bits to the right.

The carry flag is ON or OFF depending on the status prior to the execution of the ROR instruction.



- (2) When a bit device is designated for  $\textcircled{D}$ , a rotation is performed within the device range specified by digit specification.

The number of bits by which a rotation is executed is the remainder of  $n/(\text{specified number of bits})$ .

For example, when  $n = 15$  and  $(\text{specified number of bits}) = 12$  bits, the remainder of  $15/12 = 1$  is "3", and the data is rotated 3 bits.

- (3) Specify any of 0 to 15 as  $n$ .

If the value specified as  $n$  is 16 or greater, the remainder of  $n / 16$  is used for rotation.

For example, when  $n = 18$ , the contents are rotated two bits to the right since the remainder of  $18 / 16 = 1$  is "2".

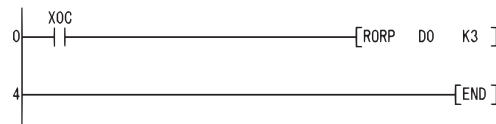
## ! Operation Error

- (1) There are no operation errors associated with the ROR(P) or RCR(P) instructions.

## Program Example

- (1) The following program rotates the contents of D0, not including the carry flag, 3 bits to the right when XC is turned ON.

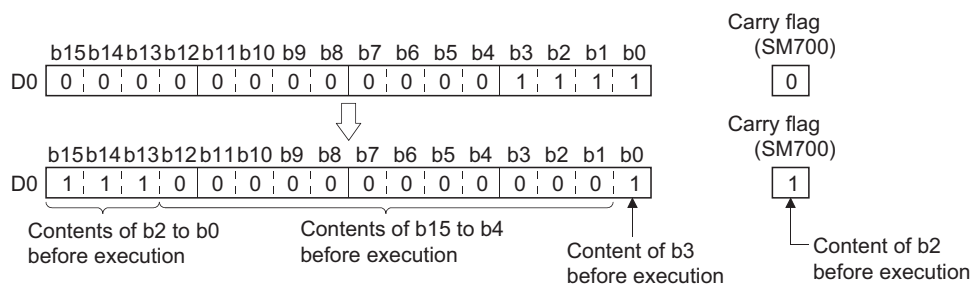
[Ladder Mode]



[List Mode]

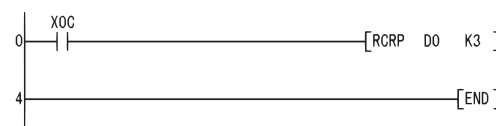
Step	Instruction	Device
0	LD	XOC
1	RORP	D0
4	END	K3

[Operation]



- (2) The following program rotates the contents of D0, including the carry flag, 3 bits to the right when XC is turned ON.

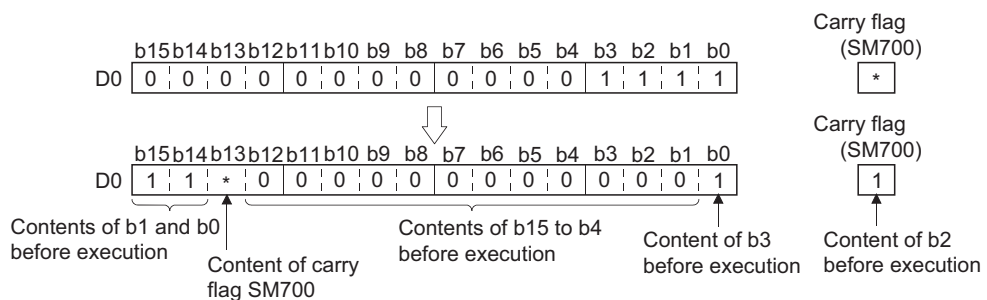
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	XOC
1	RCRP	D0
4	END	K3

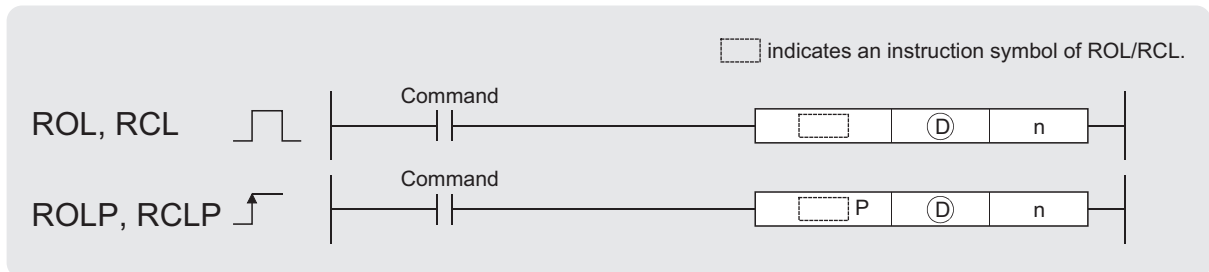
[Operation]



\* ON/OFF status of the carry flag depends on its status before the execution of ROR.

## 7.2.2 Left rotation of 16-bit data (ROL(P),RCL(P))

Basic High performance Process Redundant Universal



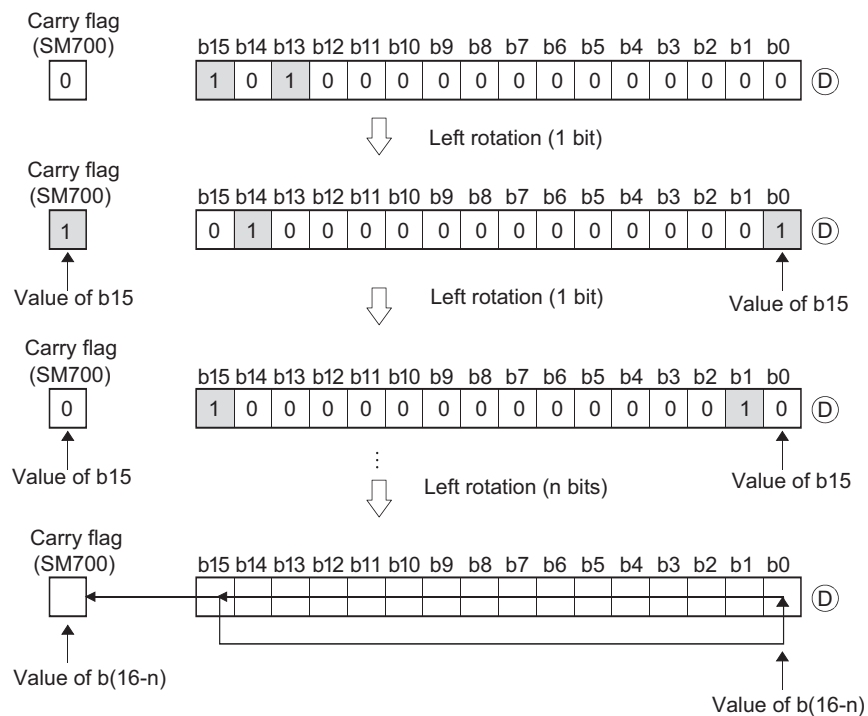
Ⓧ : Head number of the devices to rotate (BIN 16 bits)  
 n : Number of rotations (0 to 15) (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ					○			—	—
n					○			○	—

### ★ Function

#### ROL

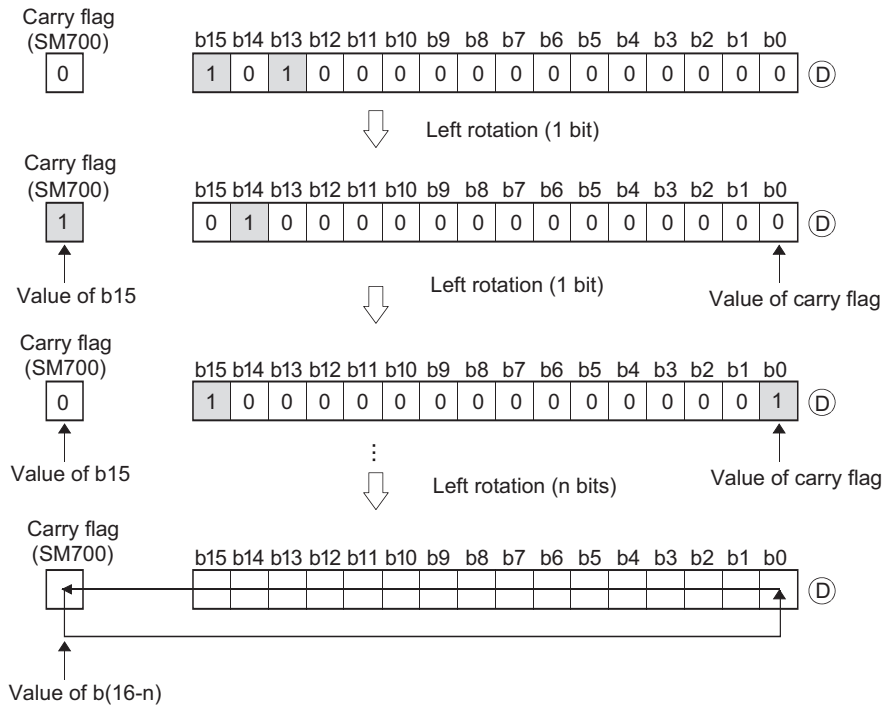
- (1) Rotates the 16-bit data of the device designated at Ⓧ, not including the carry flag, n-bits to the left.  
 The carry flag turns ON or OFF depending on its status prior to the execution of ROL instruction.



- (2) When a bit device is designated for  $\textcircled{D}$ , a rotation is performed within the device range specified by digit specification.  
The number of bits by which a rotation is executed is the remainder of  $n/(\text{specified number of bits})$ .  
For example, when  $n = 15$  and  $(\text{specified number of bits}) = 12$  bits, the remainder of  $15/12 = 1$  is "3", and the data is rotated 3 bits.
- (3) Specify any of 0 to 15 as  $n$ .  
If the value specified as  $n$  is 16 or greater, the remainder of  $n / 16$  is used for rotation.  
For example, when  $n = 18$ , the data is rotated 2 bits to the left since the remainder of  $18/16 = 1$  is "2".

**RCL**

- (1) Rotates the 16-bit data of the device designated by  $\textcircled{D}$ , including the carry flag,  $n$ -bits to the left.  
The carry flag turns ON or OFF depending on its status prior to the execution of RCL instruction.



- (2) When a bit device is designated for  $\textcircled{D}$ , a rotation is performed within the device range specified by digit specification.  
The number of bits by which a rotation is executed is the remainder of  $n/(\text{specified number of bits})$ .  
For example, when  $n = 15$  and  $(\text{specified number of bits}) = 12$  bits, the remainder of  $15/12 = 1$  is "3", and the data is rotated 3 bits.
- (3) Specify any of 0 to 15 as  $n$ .  
If the value specified as  $n$  is 16 or greater, the remainder of  $n / 16$  is used for rotation.  
For example, when  $n = 18$ , the data is rotated 2 bits to the left since the remainder of  $18/16 = 1$  is "2".

7.2 Rotation instruction  
7.2.2 Left rotation of 16-bit data (ROL(P),RCL(P))



## Operation Error

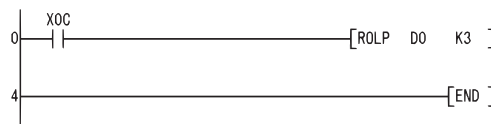
- (1) There are no operation errors associated with the ROL(P) or RCL(P) instructions.



## Program Example

- (1) The following program rotates the contents of D0, not including the carry flag, 3 bits to the left when XC is turned ON.

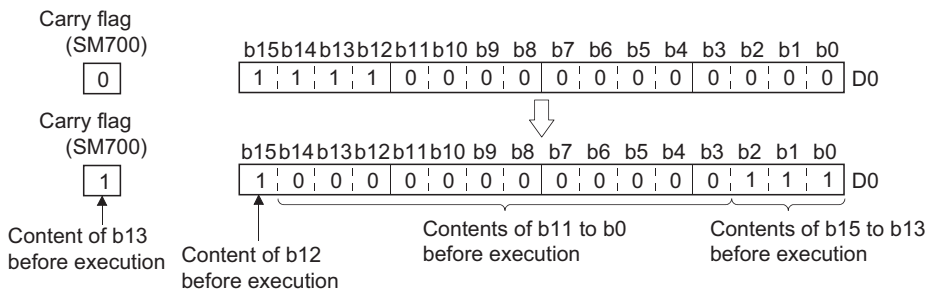
[Ladder Mode]



[List Mode]

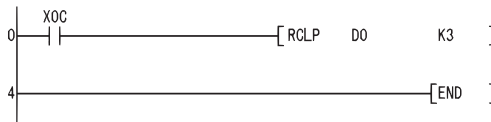
Step	Instruction	Device
0	LD	XOC
1	ROLP	D0
4	END	K3

[Operation]



- (2) The following program rotates the contents of D0, including the carry flag, 3 bits to the left when XC is turned ON.

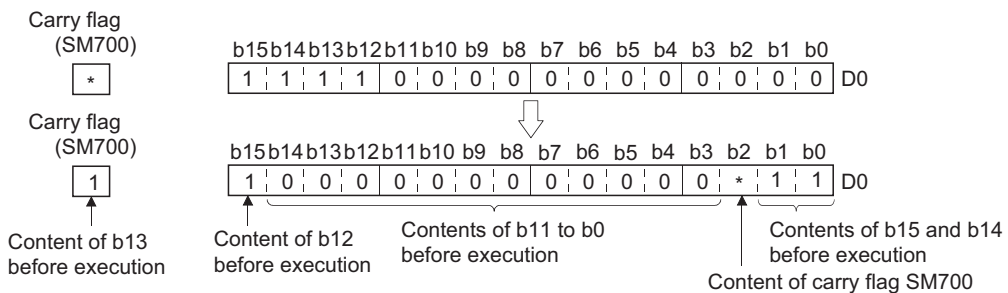
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	XOC
1	RCLP	D0
4	END	K3

[Operation]

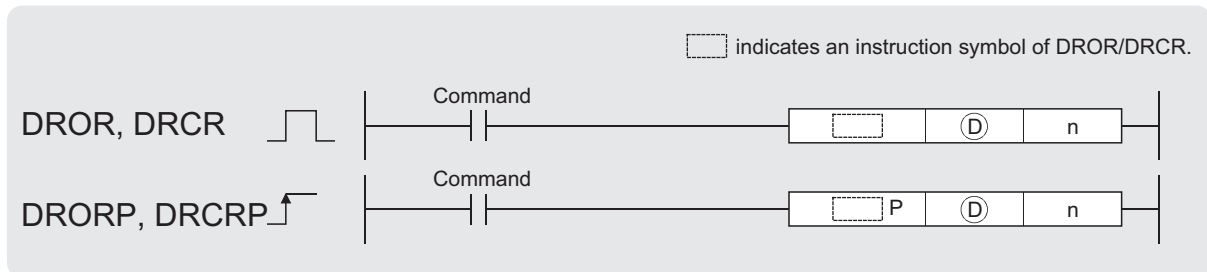


\* ON/OFF status of the carry flag depends on its status before the execution of RCL.



## 7.2.3 Right rotation of 32-bit data (DROR(P),DRCR(P))

Basic High performance Process Redundant Universal



Ⓧ : Head number of the devices to rotate (BIN 32 bits)  
n : Number of rotations (0 to 31) (BIN 16 bits)

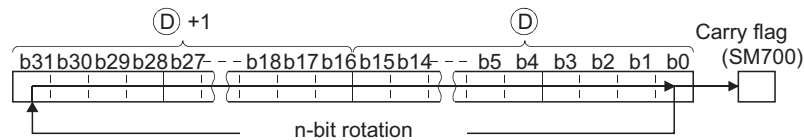
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ					○			—	—
n					○			○	—

### ★ Function

#### DROR

- (1) The 32-bit data of the device designated at Ⓧ, not including the carry flag, is rotated n-bits to the right.

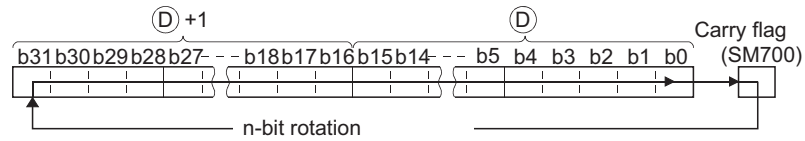
The carry flag turns ON or OFF depending on its status prior to the execution of the DROR instruction.



- (2) When a bit device is designated for Ⓧ, a rotation is performed within the device range specified by digit specification.  
The number of bits by which a rotation is executed is the remainder of  $n / (\text{specified number of bits})$ .  
For example, when  $n = 31$  and  $(\text{specified number of bits}) = 24$  bits, the remainder of  $31 / 24 = 1$  is "7", and the data is rotated 7 bits.
- (3) Specify any of 0 to 31 as n.  
If the value specified as n is 32 or greater, the remainder of  $n / 32$  is used for rotation.  
For example, when  $n = 34$ , the contents are rotated two bits to the right since the remainder of  $34 / 32 = 1$  is "2".

## DRCR

- (1) Rotates 32-bit data, including carry flag, at device designated by  $\textcircled{D}$  n bits to the right. The carry flag goes ON or OFF depending on its status prior to the execution of the DRCR instruction.



- (2) When a bit device is designated for  $\textcircled{D}$ , a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of  $n / (\text{specified number of bits})$ .  
For example, when  $n = 31$  and  $(\text{specified number of bits}) = 24$  bits, the remainder of  $31/24 = 1$  is "7", and the data is rotated 7 bits.
- (3) Specify any of 0 to 31 as n. If the value specified as n is 32 or greater, the remainder of  $n / 32$  is used for rotation. For example, when  $n = 34$ , the contents are rotated two bits to the right since the remainder of  $34 / 32 = 2$ .



## Operation Error

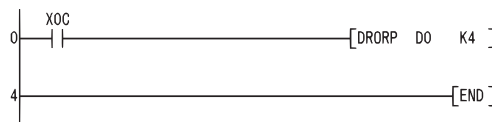
- (1) There are no operation errors associated with the DROR(P) or DRCR(P) instruction.



## Program Example

- (1) The following program rotates the contents of D0 and D1, not including the carry flag, 4 bits to the right when XC is ON.

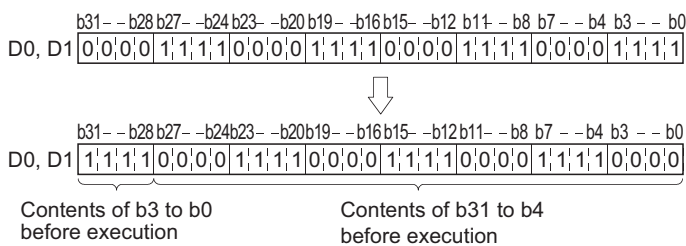
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	XOC
1	DRORP	D0 K4
4	END	

[Operation]



Carry flag (SM700)

0

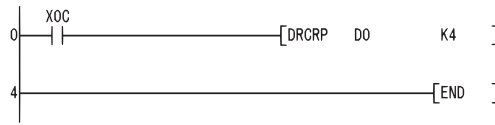
Carry flag (SM700)

1

Content of b3 before execution

- (2) The following program rotates the contents of D0 and D1, including the carry flag, 4 bits to the right when XC is ON.

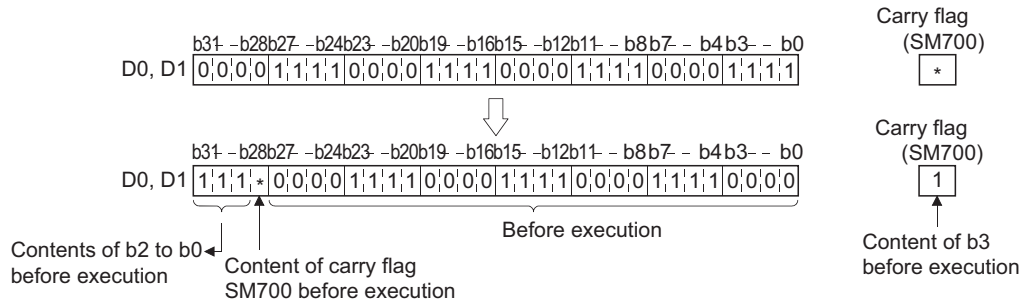
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0C
1	DRCR	D0
4	END	K4

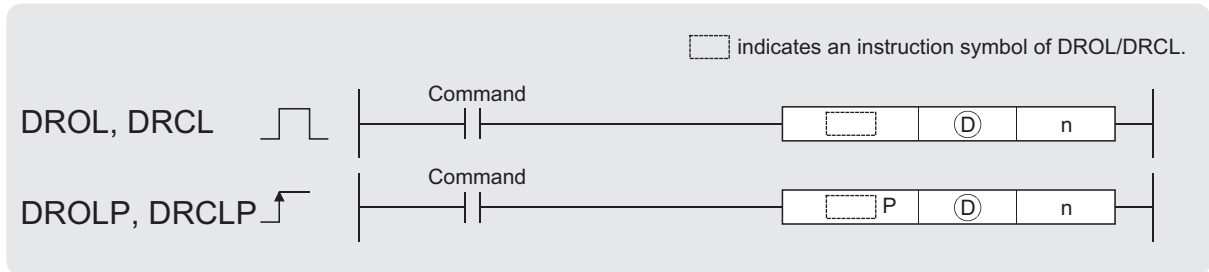
[Operation]



\* : ON/OFF status of the carry flag depends on its status before the execution of DRCR.

## 7.2.4 Left rotation of 32-bit data (DROL(P),DRCL(P))

Basic High performance Process Redundant Universal



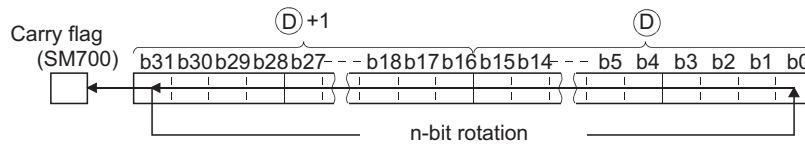
Ⓧ : Head number of the devices to rotate (BIN 32 bits)  
 n : Number of rotations (0 to 31) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ					○			—	—
n					○			○	—

### ★ Function

#### DROL

- The 32-bit data of the device designated at Ⓧ, not including the carry flag, is rotated n-bits to the left. The carry flag turns ON or OFF depending on its status prior to the execution of the DROL instruction.



- When a bit device is designated for Ⓧ, a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of  $n / (\text{specified number of bits})$ .  
 For example, when  $n = 31$  and  $(\text{specified number of bits}) = 24$  bits, the remainder of  $31/24 = 1$  is "7", and the data is rotated 7 bits.
- Specify any of 0 to 31 as n. If the value specified as n is 32 or greater, the remainder of  $n/32$  is used for rotation. For example, when  $n = 34$ , the data is rotated 2 bits to the left since the remainder of  $34/32 = 2$  is "2".

#### DRCL

- Rotates 32-bit data of the device designated by Ⓧ, including the carry flag, n-bits to the left. The carry flag turns ON or OFF depending on its status prior to the execution of the DRCL instruction.



- (2) When a bit device is designated for  $\textcircled{D}$ , a rotation is performed within the device range specified by digit specification. The number of bits by which a rotation is executed is the remainder of  $n / (\text{specified number of bits})$ .  
For example, when  $n = 31$  and (specified number of bits) = 24 bits, the remainder of  $31/24 = 1$  is "7", and the data is rotated 7 bits.
- (3) Specify any of 0 to 31 as  $n$ . If the value specified as  $n$  is 32 or greater, the remainder of  $n/32$  is used for rotation. For example, when  $n = 34$ , the data is rotated 2 bits to the left since the remainder of  $34/32 = 2$  is "2".

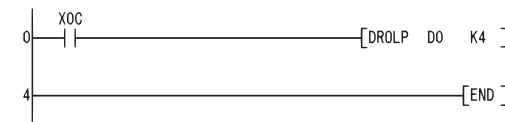
## ! Operation Error

- (1) There are no operation errors associated with the DROL(P) or DRCL(P) instructions.

## Program Example

- (1) The following program rotates the contents of D0 and D1, not including the carry flag, 4 bits to the left when XC is ON.

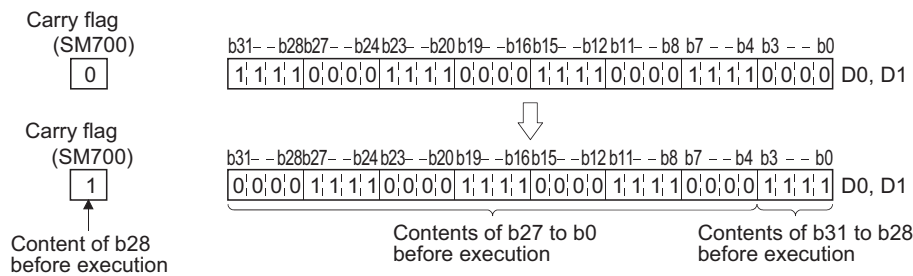
[Ladder Mode]



[List Mode]

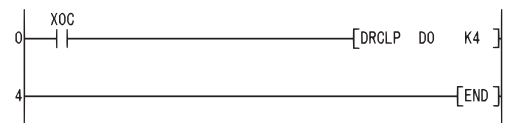
Step	Instruction	Device
0	LD	XOC
1	DROLP	D0 K4
4	END	

[Operation]



- (2) The following program rotates the contents of D0 and D1, including the carry flag, 4 bits to the left when XC is ON.

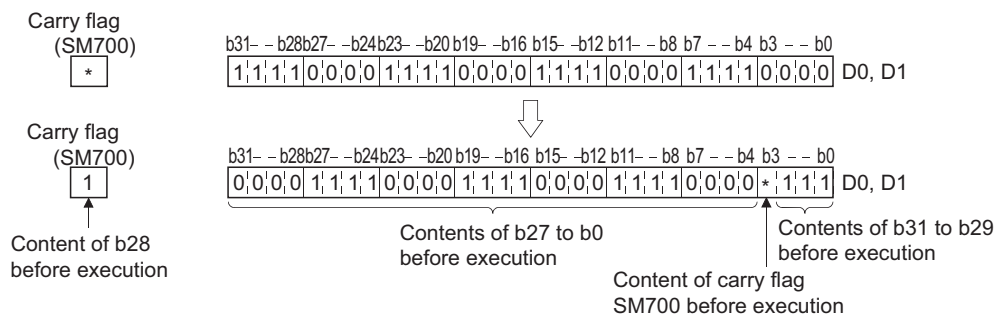
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	XOC
1	DRCLP	D0 K4
4	END	

[Operation]

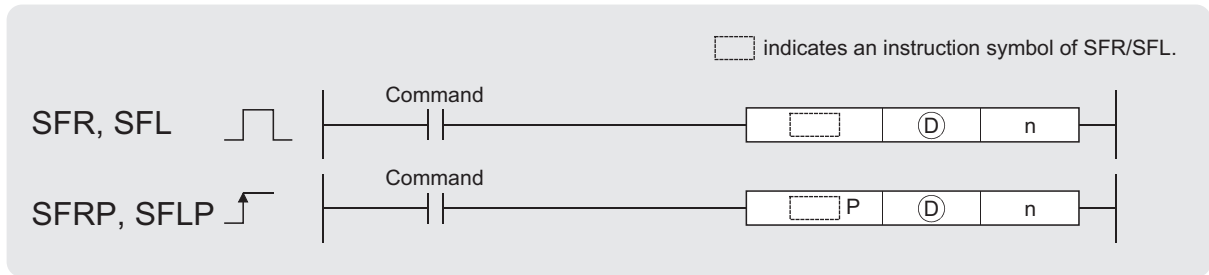


\* : ON/OFF status of the carry flag depends on its status before the execution of DRCL.

# 7.3 Shift instruction

## 7.3.1 n-bit shift to right or left of 16-bit data (SFR(P),SFL(P))

Basic High performance Process Redundant Universal



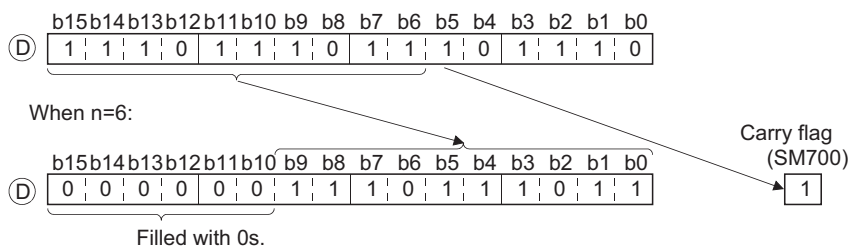
Ⓧ : Head number of the devices where shift data is stored (BIN 16 bits)  
 n : Number of shifts (0 to 15) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ					○			—	—
n					○			○	—

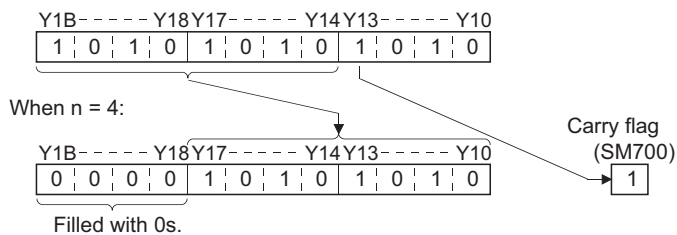
### ★ Function

#### SFR

- (1) Causes a shift to the right by n bits of the 16-bit data from the device designated at Ⓧ. The n bits from the upper bit are filled with 0s.



- (2) When a bit device is designated for Ⓧ, a right shift is executed within the device range specified by digit specification.



The number of bits by which a shift is executed is the remainder of  $n/(\text{specified number of bits})$ .

For example, when  $n = 15$  and (specified number of bits) = 8 bits, the remainder of  $15/8 = 1$  is "7", and the data is shifted 7 bits.

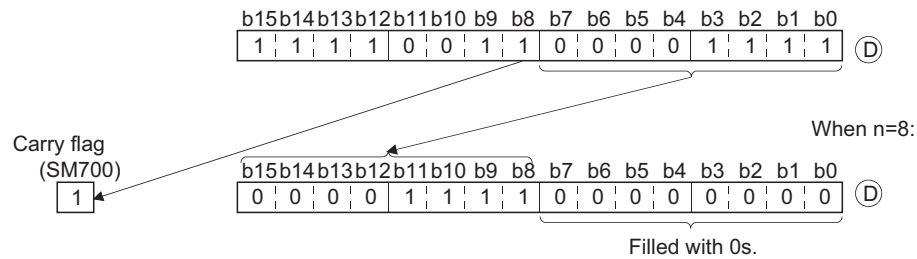
- (3) Specify any of 0 to 15 as  $n$ . If the value specified as  $n$  is 16 or greater, the remainder of  $n/16$  is used for a shift to the right.

For example, when  $n = 18$ , the data is shifted 2 bits to the right since the remainder of  $18/16 = 1$  is 2.

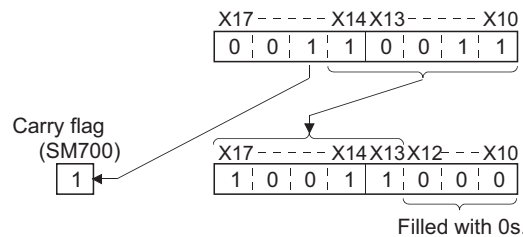
## SFL

- (1) Shifts 16-bit data at device designated by  $\textcircled{D}$   $n$  bits to the left.

Bits starting from the lowest bit to  $n$  bit are filled with 0s.



- (2) When a bit device is designated for  $\textcircled{D}$ , a left shift is executed within the device range specified by digit specification.



The number of bits by which a shift is executed is the remainder of  $n/(\text{specified number of bits})$ . For example, when  $n = 15$  and (specified number of bits) = 8 bits, the remainder of  $15/8 = 1$  is "7", and the data is shifted 7 bits.

- (3) Specify any of 0 to 15 as  $n$ . If the value specified as  $n$  is 16 or greater, the remainder of  $n/16$  is used for a shift to the left.

For example, when  $n = 18$ , the data is shifted 2 bits to the left since the remainder of  $18/16 = 1$  is "2".

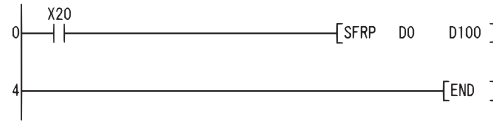
## ! Operation Error

- (1) There are no operation errors associated with the SFR(P) or SFL(P) instructions.

## Program Example

- (1) The following program shifts the data of D0 to the right by the number of bits designated by D100 when X20 is turned ON.

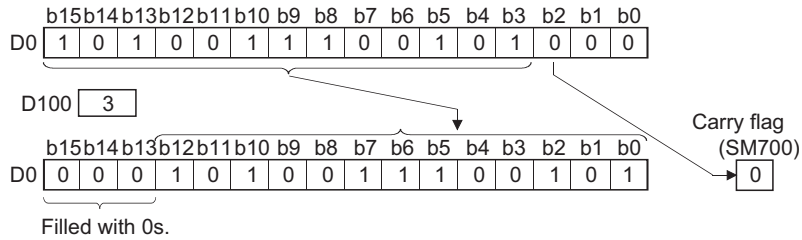
[Ladder Mode]



[List Mode]

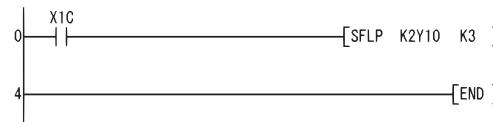
Step	Instruction	Device
0	LD	X20
1	SFRP	D0 D100
4	END	

[Operation]



- (2) The following program shifts the contents of X10 to X17 3 bits to the left when X1C is ON.

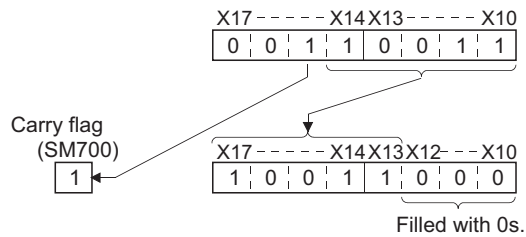
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	SFLP	K2Y10 K3
4	END	

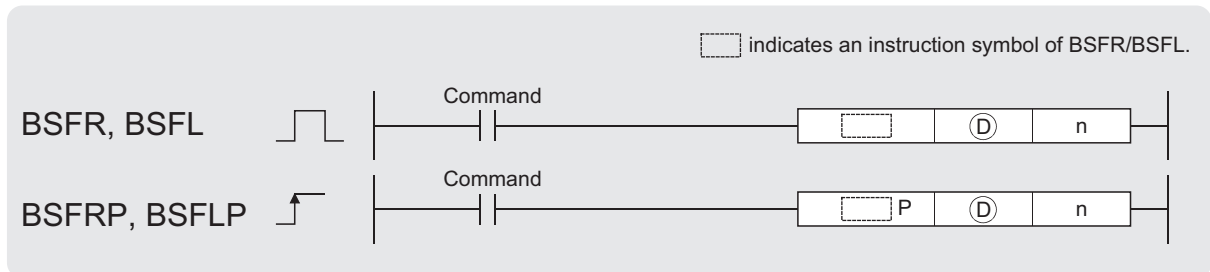
[Operation]





## 7.3.2 1-bit shift to right or left of n-bit data (BSFR(P),BSFL(P))

Basic High performance Process Redundant Universal



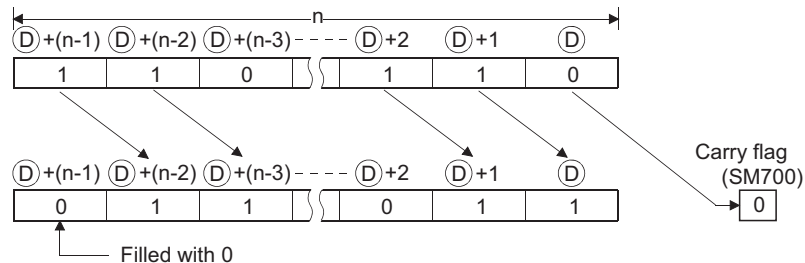
Ⓧ : Head number of the devices to be shifted (bits)  
n : Number of devices to which shift is executed (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ	○								—
n	○				○				—

### ★ Function

#### BSFR

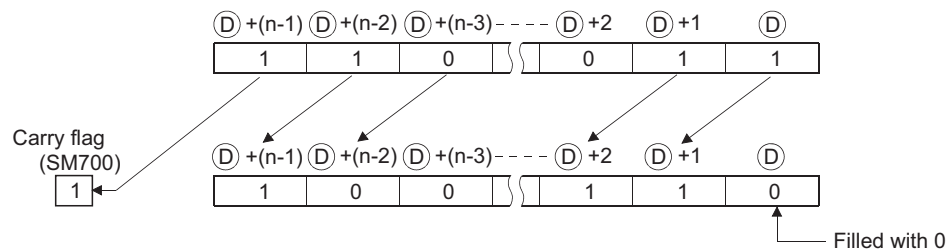
- (1) Shifts the data in n points from the device designated by Ⓧ to the right by one bit.



- (2) The device designated by Ⓧ + (n - 1) is filled with 0.

#### BSFL

- (1) Shifts the data in n points from the device designated by Ⓧ to the left by one bit.



- (2) The device designated by Ⓧ is filled with 0.

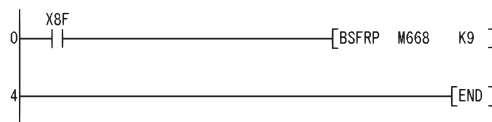
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range of the device n points from a device designated by Ⓣ, or exceeds the relevant device. (Error code: 4101)

## Program Example

- (1) The following program shifts the data at M668 to M676 to the right when X8F is turned ON.

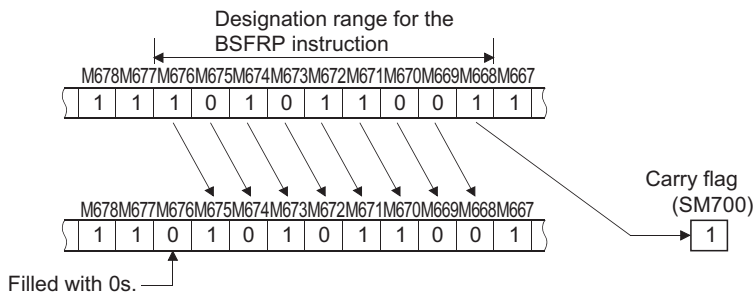
[Ladder Mode]



[List Mode]

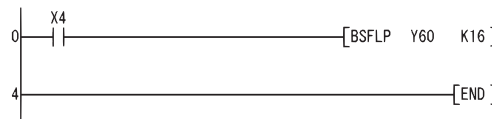
Step	Instruction	Device
0	LD	X8F
1	BSFRP	M668 K9
4	END	

[Operation]



- (2) The following program shifts the data at Y60 to Y6F to the left when X4 is turned ON.

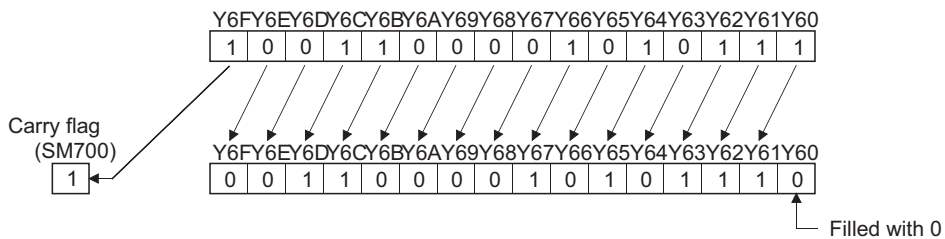
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X4
1	BSFLP	Y60 K16
4	END	

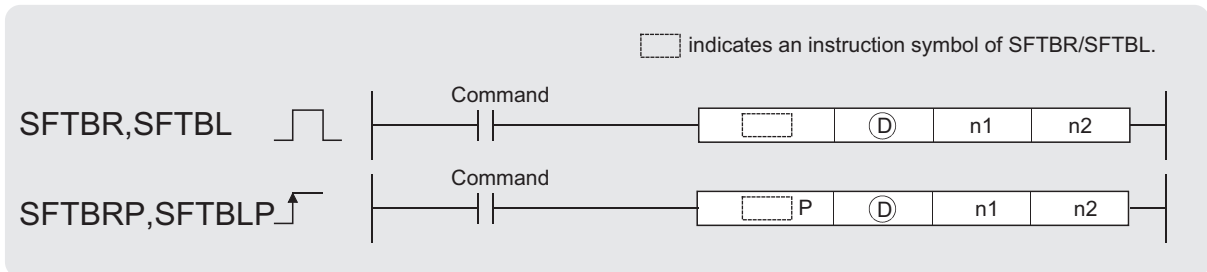
[Operation]



### 7.3.3 n-bit shift to right or left of n-bit data (SFTBR(P),SFTBL(P))



QnU(D)H)CPU: The serial number (first five digits) is "10102" or later.  
 QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓧ : Head number of the devices to be shifted (bits)  
 n1 : Number of bits to be shifted (BIN 16 bits)  
 n2 : Number of shifts (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ	○*1	—	○			—			—
n1	—	○	○			○			—
n2	—	○	○			○			—

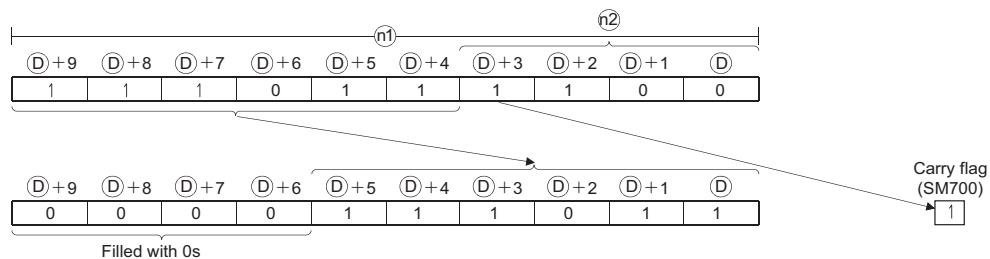
\*1 : T, C, ST, and S devices are not available.

## ★ Function

### SFTBR(P)

- (1) This instruction shifts the n1 bits data in the devices starting from the device specified by Ⓧ to the right by n2 bits.

n1=10, n2=4



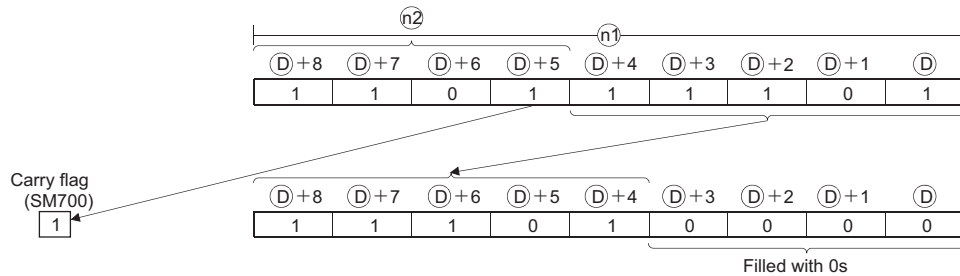
- (2) n1 and n2 are specified under the condition that n1 is larger than n2. If the value of n2 is equal to or larger than the value of n1, the remainder of n2 / n1 (n2 divided by n1) is used for a shift.
- (3) This instruction specifies n1 ranged from 1 to 64.
- (4) Bits starting from the highest bit to n2th bit are filled with 0s. If the value of n2 is larger than the value of n1, the remainder of n2 / n1 will be 0.
- (5) If the value specified by n1 or n2 is 0, the instruction will be not processed.

7.3 Shift instruction  
7.3.3 n-bit shift to right or left of n-bit data (SFTBR(P),SFTBL(P))

**SFTBL(P)**

- (1) This instruction shifts the  $n_1$  bits data in the devices starting from the device specified by  $\text{D}$  to the left by  $n_2$  bits.

$n_1=10, n_2=4$



- (2)  $n_1$  and  $n_2$  are specified under the condition that  $n_1$  is larger than  $n_2$ . If the value of  $n_2$  is equal to or larger than the value of  $n_1$ , the remainder of  $n_2 / n_1$  ( $n_2$  divided by  $n_1$ ) is used for a shift. However, if the remainder of  $n_2 / n_1$  is 0, the instruction will be not processed.
- (3) This instruction specifies  $n_1$  ranged from 1 to 64.
- (4) Bits starting from the lowest bit to  $n_2$ th bit are filled with 0s. If the value of  $n_2$  is larger than the value of  $n_1$ , the remainder of  $n_2 / n_1$  will be 0.
- (5) If the value specified by  $n_1$  or  $n_2$  is 0, the instruction will be not processed.

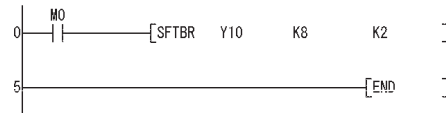
**Operation Error**

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The value specified by  $n_1$  is other than 0 to 64. (Error code: 4100)
  - The value data specified by  $n_2$  is negative. (Error code: 4100)
  - The range of devices specified by  $n_1$  exceeds the range of devices specified by  $\text{D}$ . (Error code: 4101)

## Program Example

- (1) The following program shifts the data of Y10 to Y17 (8 bits) specified by ① to the right by 2 bits (n2), when M0 is turned on.

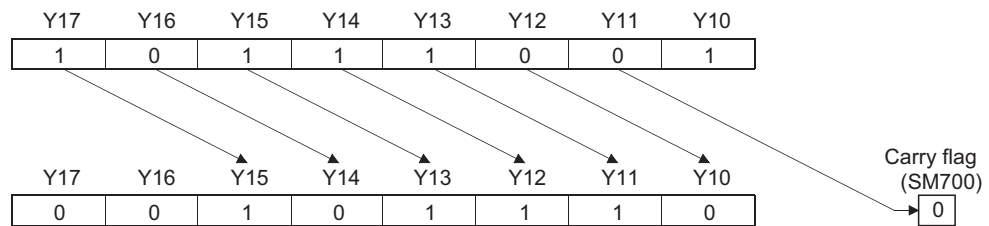
[Ladder Mode]



[List Mode]

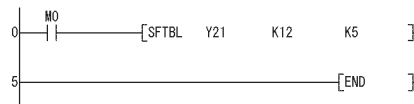
Step	Instruction	Device
0	LD	M0
1	SFTBR	Y10 K8 K2
5	END	

[Operation]



- (2) The following program shifts the data of Y21 to Y2C (12 bits) specified by ① to the left by 5 bits (n5), when M0 is turned on.

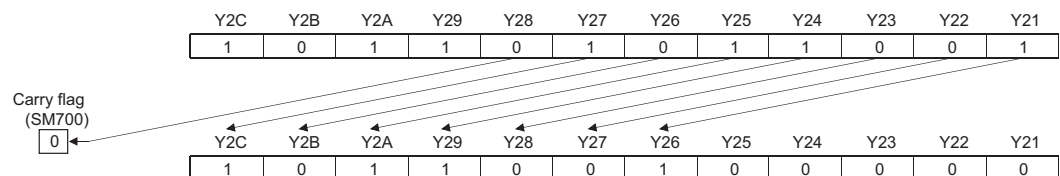
[Ladder Mode]



[List Mode]

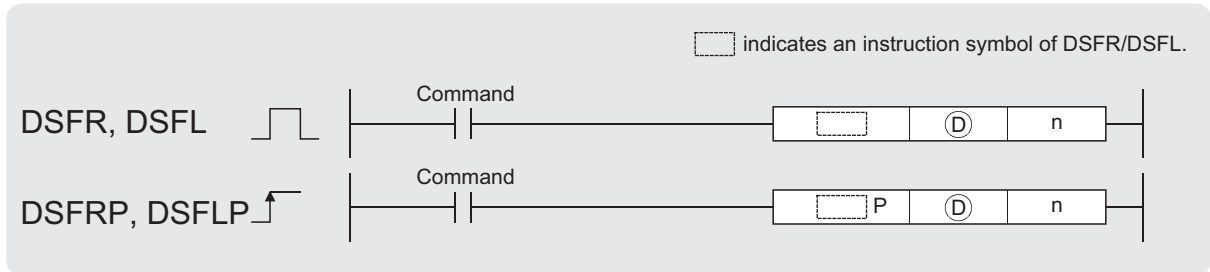
Step	Instruction	Device
0	LD	M0
1	SFTBL	Y21 K12 K5
5	END	

[Operation]



### 7.3.4 1-word shift to right or left of n-word data (DSFR(P),DSFL(P))

Basic High performance Process Redundant Universal



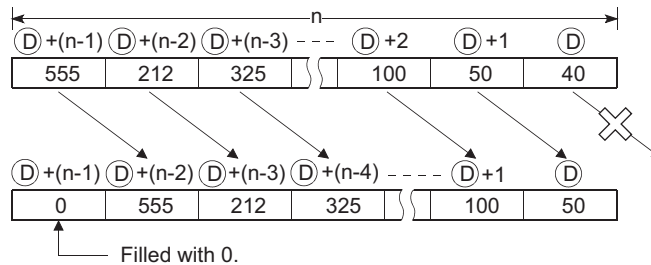
Ⓧ : Head number of the devices to be shifted (BIN 16 bits)  
 n : Number of devices to which shift is executed (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ	—	○				—			—
n	○	○				○			—

## ★ Function

### DSFR

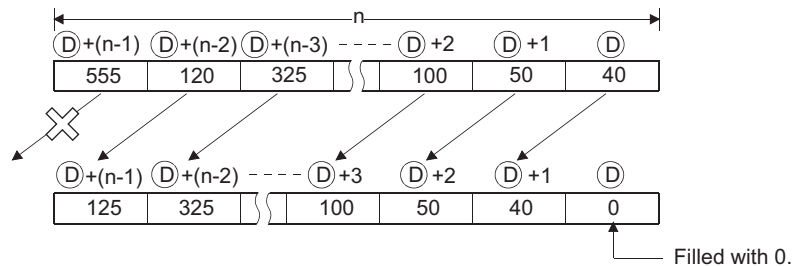
(1) Shifts data n points from device designated by Ⓧ 1-word to the right.



(2) The device designated by  $D + (n - 1)$  is filled with 0.

### DSFL

(1) Shifts data n points from device designated by Ⓧ 1-word to the left.



(2) The device designated by Ⓧ is filled with 0.

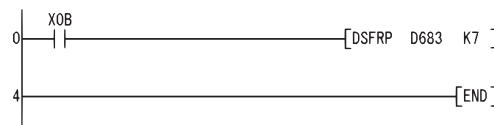
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range of the device n points from a device designated by  $\textcircled{D}$ , or exceeds the relevant device. (Error code: 4101)

## Program Example

- (1) The following program shifts the contents of D683 to D689 to the right when XB is turned ON.

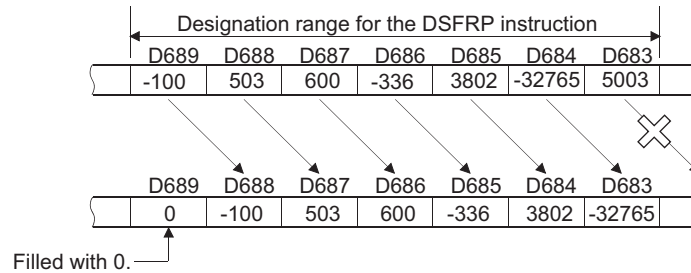
[Ladder Mode]



[List Mode]

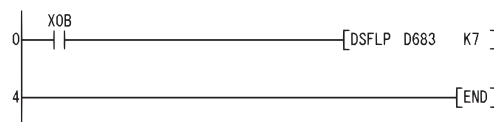
Step	Instruction	Device
0	LD	X0B
1	DSFRP	D683 K7
4	END	

[Operation]



- (2) The following program shifts the contents of D683 to D689 to the left when XB is turned ON.

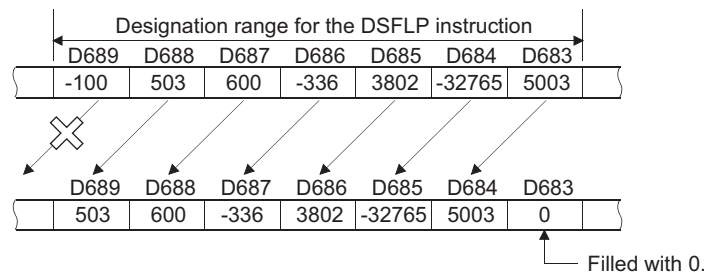
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0B
1	DSFLP	D683 K7
4	END	

[Operation]

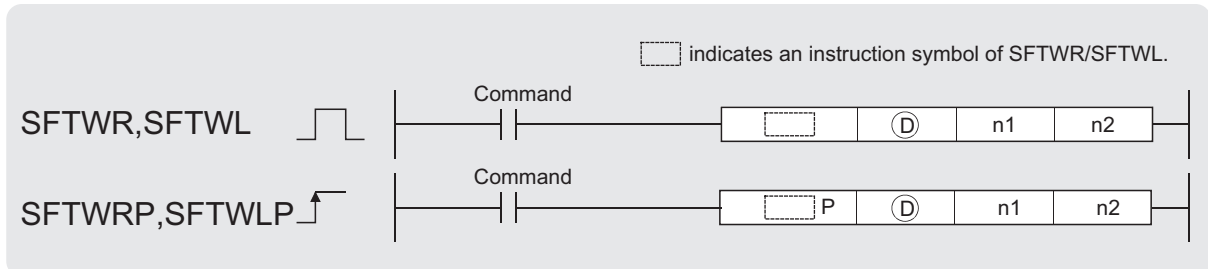


## 7.3.5 n-bit shift to right or left of n-word data (SFTWR(P),SFTWL(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓣ : Head number of the devices to be shifted (BIN 16 bits)

n1 : Number of words to be shifted (BIN 16 bits)

n2 : Number of shifts (BIN 16 bits)

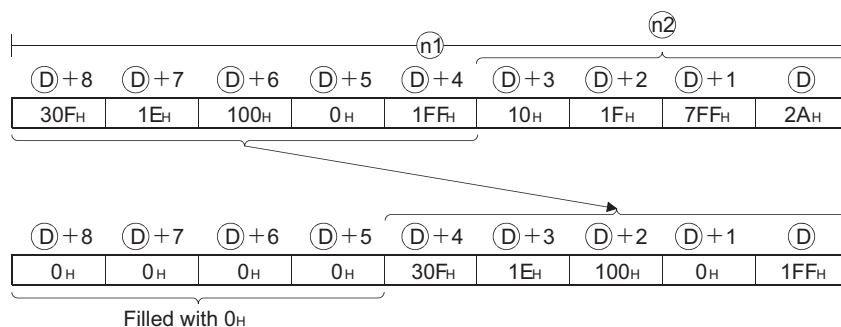
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓣ	—	○	○			—			—
n1	—	○	○			○			—
n2	—	○	○			○			—

### ★ Function

#### SFTWR(P)

- (1) This instruction shifts n1 words data in the devices starting from the device specified by Ⓣ to the right by n2 words.

n1=9, n2=4



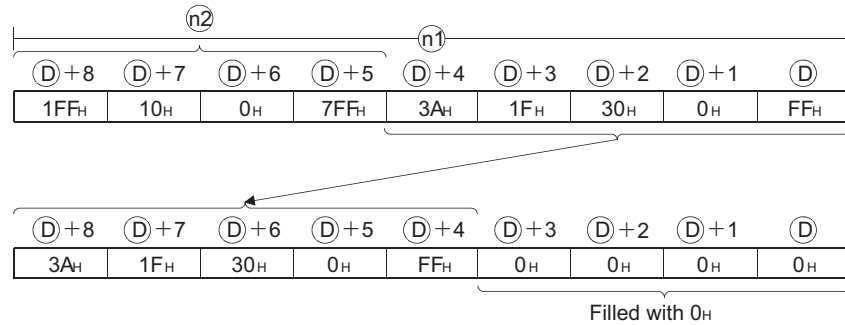
- (2) The n2 words data in the devices starting from the highest device are filled with 0s.
- (3) If the value specified by n1 or n2 is 0, the instruction will be not processed.
- (4) If the value of n2 is equal to or larger than the value of n1, the n1 words data in the devices starting from the device specified by Ⓣ will be filled with 0s.



## SFTWL(P)

- (1) This instruction shifts the  $n1$  words data in the devices starting from the device specified by  $\textcircled{D}$  to the left by  $n2$  words.

$n1=9, n2=4$



- (2) The  $n2$  words in the devices starting from the lowest device are filled with 0s.  
 (3) If the value specified by  $n1$  or  $n2$  is 0, the instruction will be not processed.  
 (4) If the value of  $n2$  is equal to or greater than the value of  $n1$ , the  $n1$  words devices starting from the device specified by  $\textcircled{D}$  will be filled with 0s.

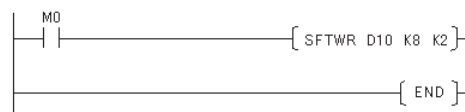
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- $n1$  or  $n2$  is negative value. (Error code: 4100)
  - The range of devices specified by  $n1$  exceeds the range of devices specified by  $\textcircled{D}$ . (Error code: 4101)

## Program Example

- (1) The following program shifts the 8 words ( $n1$ ) data stored in the devices starting from D10 specified by  $\textcircled{D}$  to the right by 2 words ( $n2$ ), when M0 is turned on.

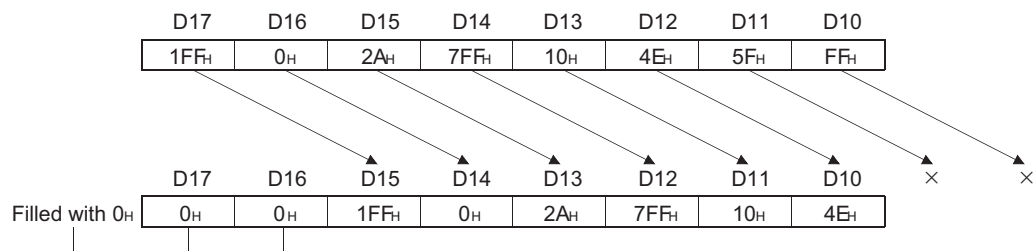
[Ladder Mode]



[List Mode]

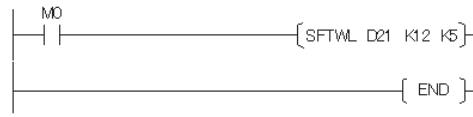
Step	Instruction	Device
0	LD	M0
1	SFTWR	D10 K8 K2
5	END	

[Operation]



(2) The following program shifts the 12 words (n1) data in the devices starting from D21 specified by Ⓢ to the left by 5 words (n2), when M0 is turned on.

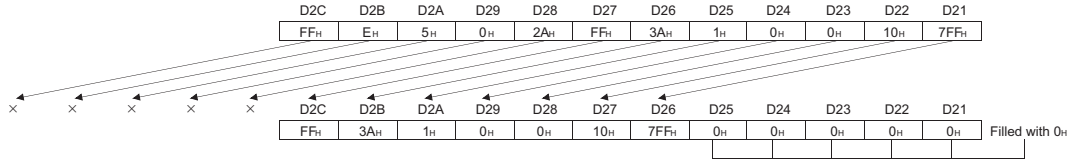
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	SFTWL	D21 K12 K5
5	END	

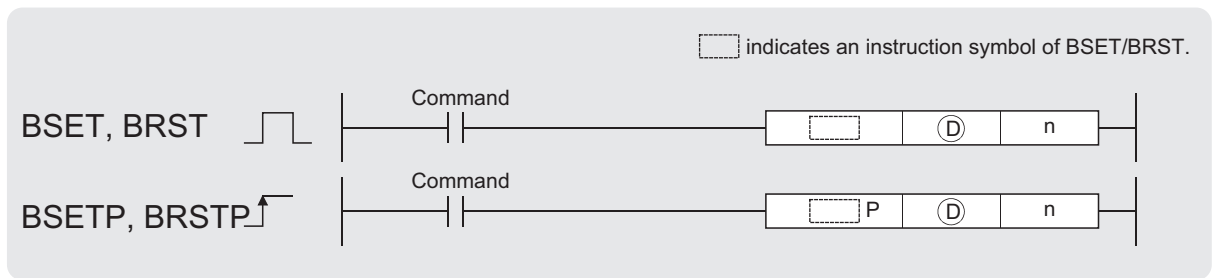
[Operation]



# 7.4 Bit processing instructions

## 7.4.1 Bit set and reset for word devices (BSET(P),BRST(P))

Basic High performance Process Redundant Universal



(D) : Number of the device whose bits are set/reset (BIN 16 bits)  
 n : Number of the bit to be set/reset (0 to 15) (BIN 16 bits)

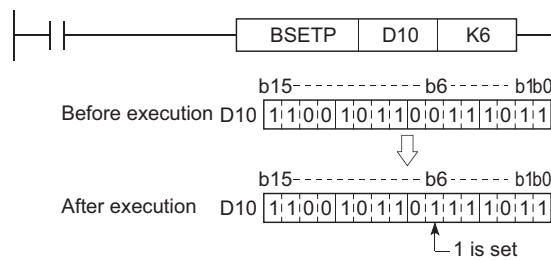
Setting Data	Internal Devices		R, ZR	J0A0		U0AG0	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
(D)								—	—
n								○	—

7

### ★ Function

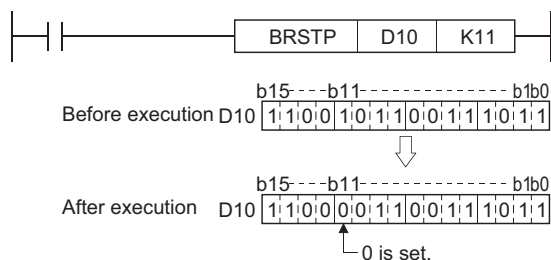
#### BSET

- (1) Sets (sets "1" at) the nth bit in the word device designated at (D).
- (2) If n exceeds "15", bit set/reset is performed with the lower 4 bits of the data.



#### BRST

- (1) Resets the nth bit of a word device designated by (D) to 0.
- (2) If n exceeds "15", bit set/reset is performed with the lower 4 bits of the data.



7.4 Bit processing instructions  
7.4.1 Bit set and reset for word devices (BSET(P),BRST(P))

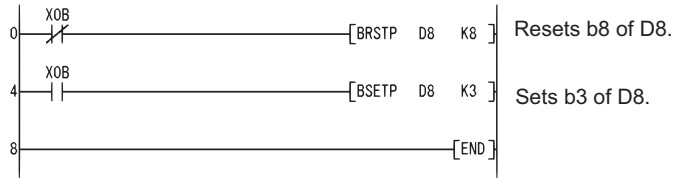
## Operation Error

(1) There are no operation errors associated with the BSET(P) or BRST(P) instructions.

## Program Example

(1) The following program resets the 8th bit of D8 (b8) to 0 when XB is OFF, and sets the 3rd bit of D8 (b3) to 1 when XB is ON.

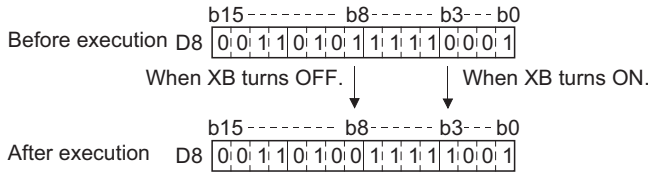
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LDI	X0B
1	BRSTP	D8 K8
4	LD	X0B
5	BSETP	D8 K3
8	END	

[Operation]

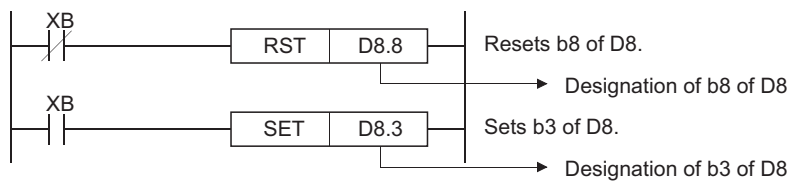


### Remark

Bit set or reset of word devices can also be conducted by bit designation of word devices.

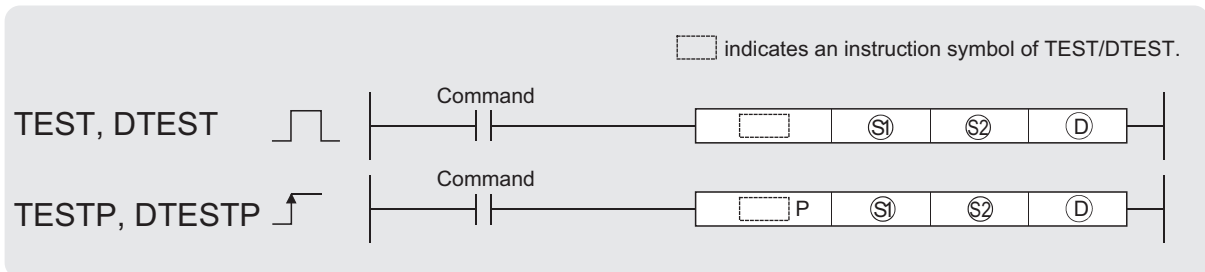
- For the bit specification for word devices, link direct devices, refer to the QnUCPU User's Manual(Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual(Function Explanation, Program Fundamentals)

The processing of program example (1) would be conducted as shown below if bit designation of a word device had been used:



## 7.4.2 Bit tests (TEST(P),DTEST(P))

Basic High performance Process Redundant Universal



Ⓢ<sub>1</sub>: Number of the device where bit data to be extracted is stored (BIN 16 bits)

Ⓢ<sub>2</sub>: Location of the bit data to be extracted (0 to 15 (TEST)/0 to 31 (DTEST)) (BIN 16/32 bits)

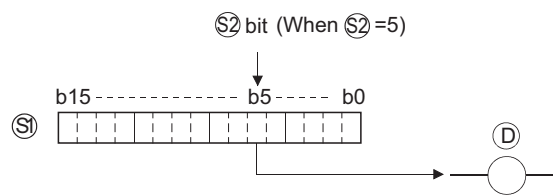
Ⓧ: Number of the bit device where the extracted data will be stored (bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>								—	—
Ⓢ <sub>2</sub>								○	—
Ⓧ								—	—

### ★ Function

#### TEST

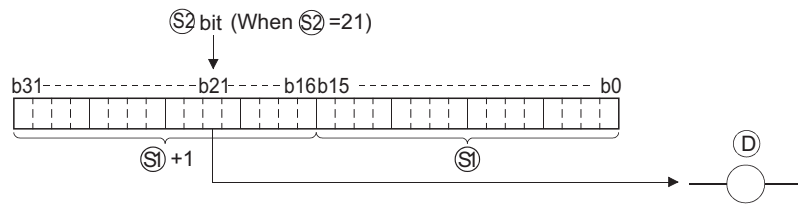
- (1) Fetches bit data at the location designated by Ⓢ<sub>2</sub> within the word device designated by Ⓢ<sub>1</sub>, and writes it to the bit device designated by Ⓧ.
- (2) The bit device designated by Ⓧ is OFF when the relevant bit is "0" and ON when it is "1".
- (3) The position designated by Ⓢ<sub>2</sub> indicates the position of an individual bit in a 1-word data block (0 to 15). When 16 or more is designated at Ⓢ<sub>2</sub>, the target is the bit data at the position indicated by the remainder of  $n / 16$ . For example, when  $n = 18$ , the target is the data at b2 since the remainder of  $18 / 16 = 1$  is "2".



#### DTEST

- (1) Fetches bit data at the location designated by Ⓢ<sub>2</sub> within the 2-word device designated by Ⓢ<sub>1</sub>, or Ⓢ<sub>1</sub>+1, and writes it to the bit device designated by Ⓧ.
- (2) The bit device designated by Ⓧ is OFF when the relevant bit is "0" and ON when it is "1".

- (3) The position designated by  $\textcircled{S2}$  indicates the position of an individual bit in a 2-word data block (0 to 31). When 32 or more is designated at  $\textcircled{S2}$ , the target is the bit data at the position indicated by the remainder of  $n / 32$ . For example, when  $n = 34$ , the target is the data at b2 since the remainder of  $34 / 32 = 1$  is "2".



## Operation Error

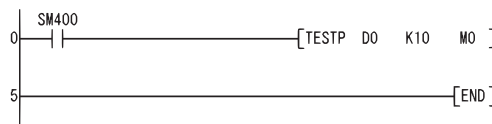
- (1) There are no operation errors associated with the TEST(P) or DTEST(P) instructions.



## Program Example

- (1) The following program turns M0 ON or OFF based on the status of the 10th bit in the 1-word data block (D0).

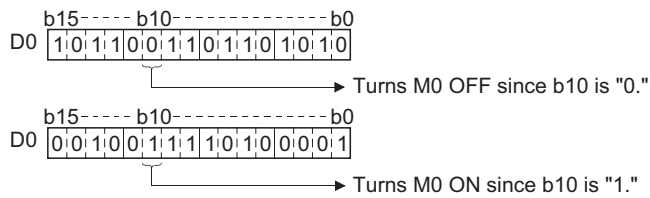
[Ladder Mode]



[List Mode]

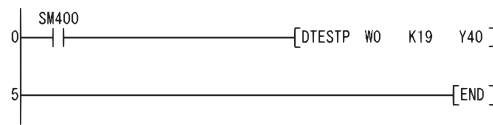
Step	Instruction	Device
0	LD	SM400
1	TESTP	D0 K10 M0
5	END	

[Operation]



- (2) The following program turns Y40 ON or OFF, depending on the status of the 19th bit of the 2-word data (W0 and W1).

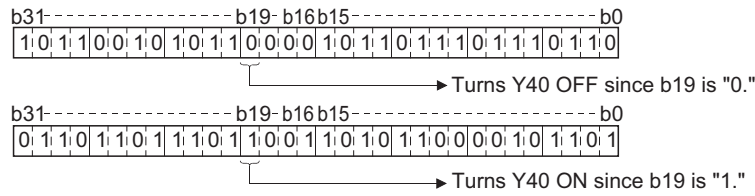
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DTESTP	W0 K19 Y40
5	END	

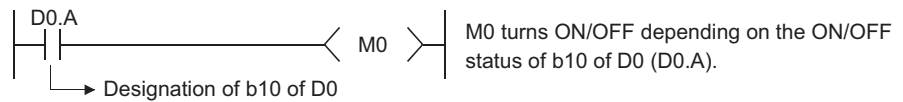
[Operation]



**Remark**

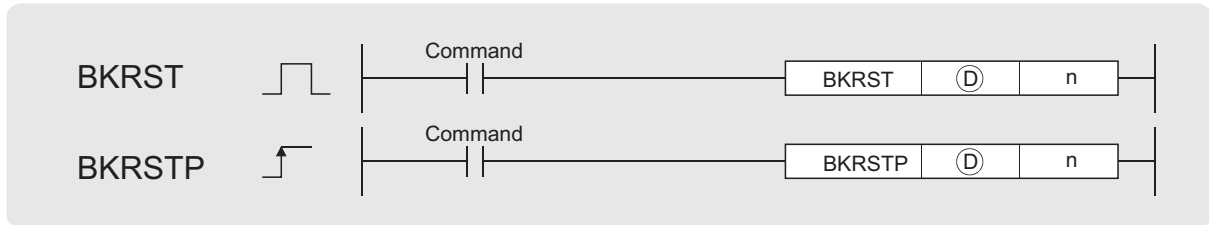
Programs using the bit test instruction can be rewritten as programs using bit designation of word devices.

If the program in example (1) were changed to use bit designation of a word device, it would appear as follows:



## 7.4.3 Batch reset of bit devices (BKRST(P))

Basic High performance Process Redundant Universal



Ⓧ : Head number of the devices to be reset (bits)  
 n : Number of the devices to be reset (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ		○				—			—
n		○				○			—

### ★ Function

(1) Resets bit device n-points from the bit device designated by Ⓧ.

Device	Status
Annunciator (F)	<ul style="list-style-type: none"> <li>• Turns device n-points from annunciator (F) number designated by Ⓧ OFF.</li> <li>• Deletes annunciator number turned OFF from SD64 to SD79 and compresses remaining data forward.</li> <li>• Stores number of annunciators stored from SD64 to SD79 at SD63.</li> </ul>
Timer (T) Counter (C)	<ul style="list-style-type: none"> <li>• Sets the current value n-points from timer (T) or counter c designated by (C) to 0, and turns coil contact OFF.</li> </ul>
Bit devices other than the above	<ul style="list-style-type: none"> <li>• Turns OFF coil or contact n-points from the device designated by Ⓧ.</li> </ul>

(2) If the designated device is OFF, the device status will not change.

### ! Operation Error

(1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The n-bit range from the Ⓧ, or device exceeds the range of that device.

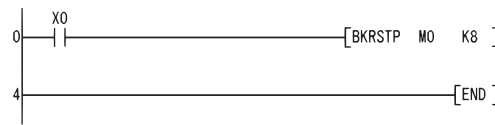
(Error code: 4101)



## Program Example

- (1) The following program turns OFF devices from M0 to M7 when X0 is turned ON.

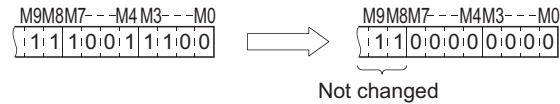
[Ladder Mode]



[List Mode]

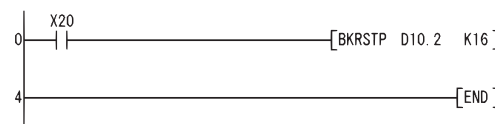
Step	Instruction	Device
0	LD	X0
1	BKRSTP	M0
4	END	K8

[Operation]



- (2) The following program sets data from 2nd bit (b2) of D10 to 1st bit (b1) of D11 to 0 when X20 is turned ON.

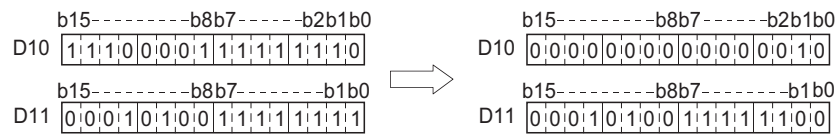
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	BKRSTP	D10.2
4	END	K16

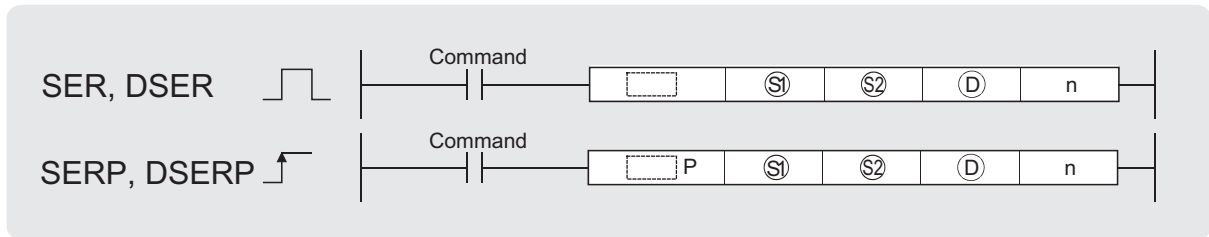
[Operation]



# 7.5 Data processing instructions

## 7.5.1 16-bit and 32-bit data searches (SER(P),DSER(P))

Basic High performance Process Redundant Universal



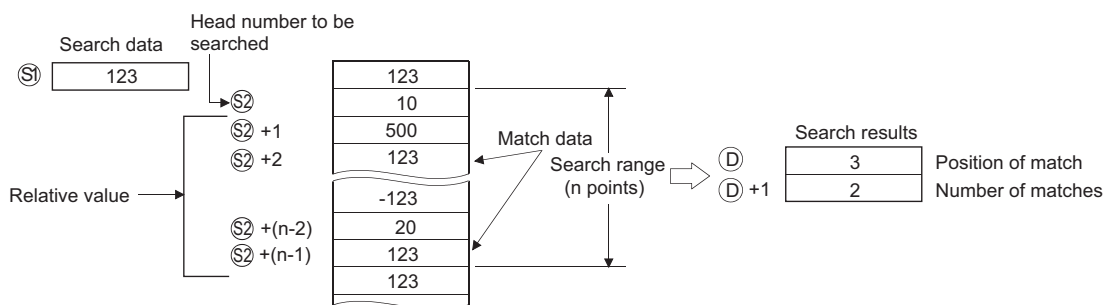
- Ⓢ<sub>1</sub> : Search data or head number of the devices where the search data is stored (BIN 16/32 bits)
- Ⓢ<sub>2</sub> : Data to be searched or head number of the devices where the data to be searched is stored (BIN 16 bits)
- Ⓧ : Head number of the devices where the search result will be stored (BIN 16 bits)
- n : Number of searches (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	○	○		○		○		○	—
Ⓢ <sub>2</sub>	—	○		—		—		—	—
Ⓧ	—	○		—		○		—	—
n	○	○		○		○		○	—

### ★ Function

#### SER

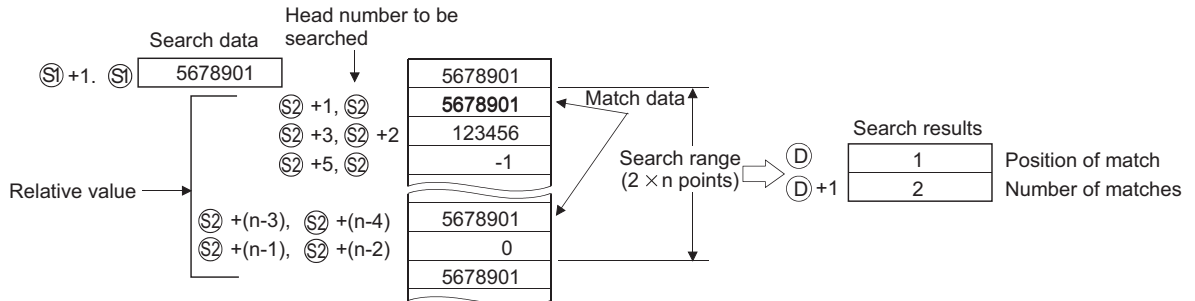
- (1) Searches n points from the 16-bit data of the device designated by Ⓢ<sub>2</sub>, regarding 16-bit data of the device designated by Ⓢ<sub>1</sub> as a keyword. Then, the number of matches with the keyword is stored at the device designated by Ⓧ + 1, and the first matched device number (in the relative number from Ⓢ<sub>2</sub>) is stored at the device designated by Ⓧ.



- (2) No processing is conducted if n is 0 or a negative value.
- (3) If no matches are found in the search, the devices designated at Ⓧ and Ⓧ + 1 become "0".

## DSER

- (1) Searches  $n$  points from the device designated by  $\textcircled{S2}$  in 32-bit units ( $2 \times n$  points in 16-bit units.) regarding 32-bit data of the device designated by  $\textcircled{S1} + 1$  and  $\textcircled{S1}$  as a keyword. Then, the number of matches with the keyword is stored at the device designated by  $\textcircled{D} + 1$ , and the first matched device number (in the relative number from  $\textcircled{S2}$ ) is stored at the device designated by  $\textcircled{D}$ .



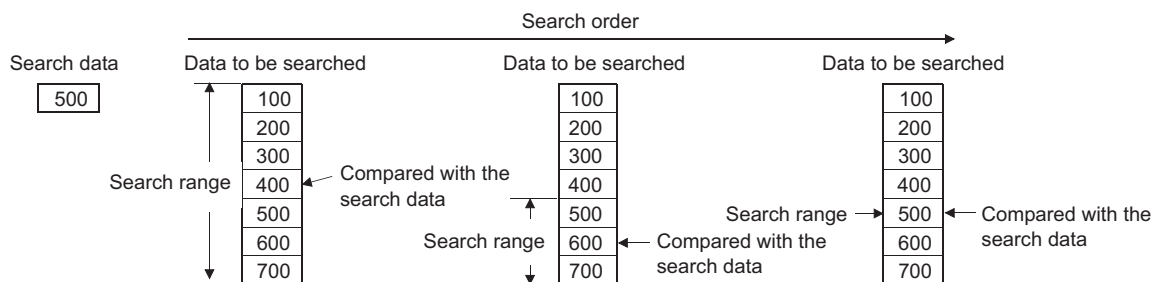
- (2) No processing is conducted if  $n$  is 0 or a negative value.
- (3) If no matches are found in the search, the devices designated at  $\textcircled{D}$  and  $\textcircled{D} + 1$  become "0".

### POINT

If the data to be searched using the SER/DSER instruction is sorted in the ascending order, searches can be accelerated by the use of the binary search method, which is activated by turning SM702 \*1 ON. However, correct search results are not obtained if SM702 is turned ON when the data to be searched is not sorted in the ascending order.

\*1: SM702 is the special relay for setting the search method.

- SM702 OFF: Sequential search method (linear search method) (Comparison with the search data starts from the beginning of the data to be searched.)
- SM702 ON: Binary search method (Obtains the center value of the sorted array and decides if the obtained value is larger or smaller than the search value, then, chooses the area for search between the larger and smaller value divisions. By repeating this process, the area for search is narrowed down.)



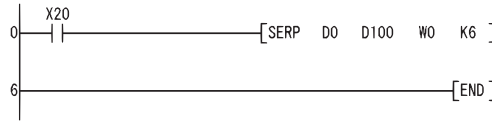
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The location  $n$ -points from the device  $\textcircled{S2}$  exceeds the designated device range. (Error code: 4101)
  - The device specified by  $\textcircled{D}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program searches D100 to D105 for the contents of D0 when X20 is ON, and stores the search results at W0 and W1.

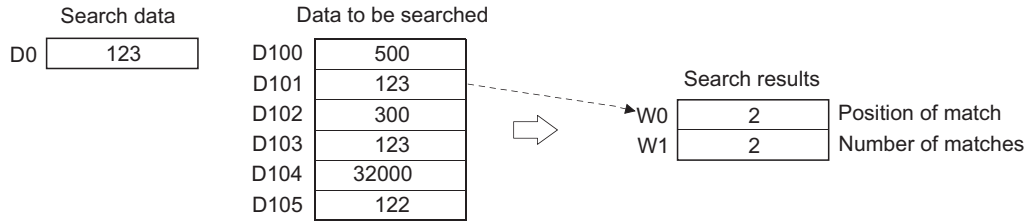
[Ladder Mode]



[List Mode]

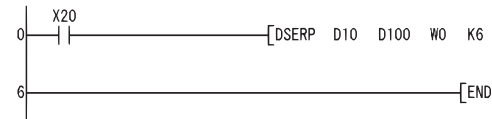
Step	Instruction	Device
0	LD	X20
1	SERP	D0 D100 W0 K6
6	END	

[Operation]



- (2) The following program searches D100 to D111 for the contents of D11 and D10 when X20 is ON, and stores the search results at W0 and W1.

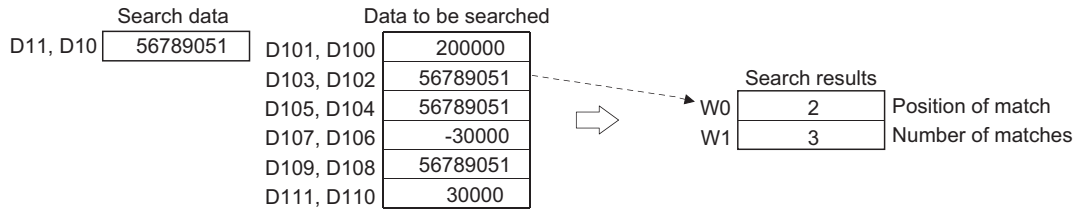
[Ladder Mode]



[List Mode]

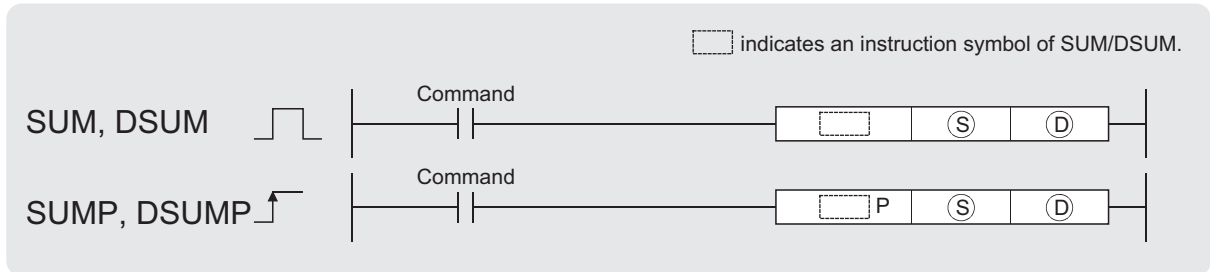
Step	Instruction	Device
0	LD	X20
1	DSERP	D10 D100 W0 K6
6	END	

[Operation]



## 7.5.2 16-bit and 32-bit data checks (SUM(P),DSUM(P))

Basic High performance Process Redundant Universal



Ⓢ: Head number of the devices where the total number of bits of "1" is counted (BIN 16/32 bits)

Ⓣ: Head number of the devices where the total number of the bits will be stored (BIN 16/32 bits)

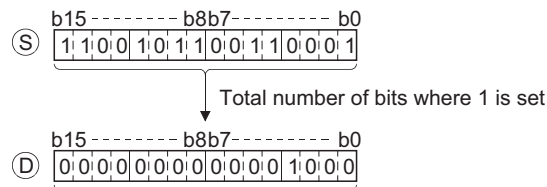
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

7

### ★ Function

#### SUM

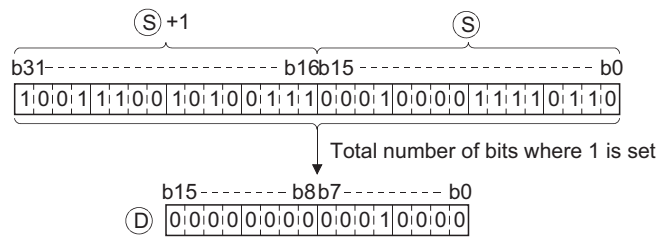
From the 16-bit data in the device designated by Ⓢ, stores the total number of bits where 1 is set, in the device designated by Ⓣ.



Stores the total number of bits where 1 is set in BIN.  
(There are 8 bits where 1 is set in the example.)

#### DSUM

From the 32-bit data in the device designated by Ⓢ, stores the total number of bits where 1 is set, in the device designated by Ⓣ.



Stores the total number of bits where 1 is set in BIN.  
(There are 16 bits where 1 is set in the example.)

7.5 Data processing instructions  
7.5.2 16-bit and 32-bit data checks (SUM(P),DSUM(P))



## Operation Error

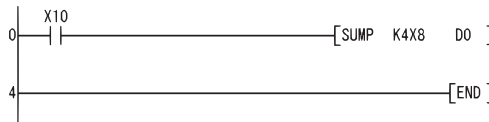
- (1) There are no operation errors associated with the SUM(P) or DSUM(P) instructions.



## Program Example

- (1) The following program stores the number of bits which are ON from X8 to X17 into D0 when X10 is turned ON.

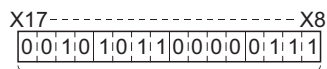
[Ladder Mode]



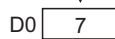
[List Mode]

Step	Instruction	Device
0	LD	X10
1	SUMP	K4X8
4	END	D0

[Operation]

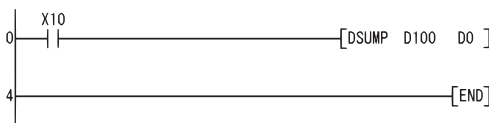


Stores the total number of bits where 1 is set at D0.



- (2) The following program stores the number of bits which are ON in D100 and D101 into D0 when X10 is turned ON.

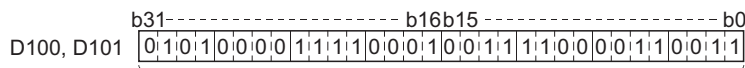
[Ladder Mode]



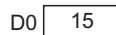
[List Mode]

Step	Instruction	Device
0	LD	X10
1	DSUMP	D100
4	END	D0

[Operation]

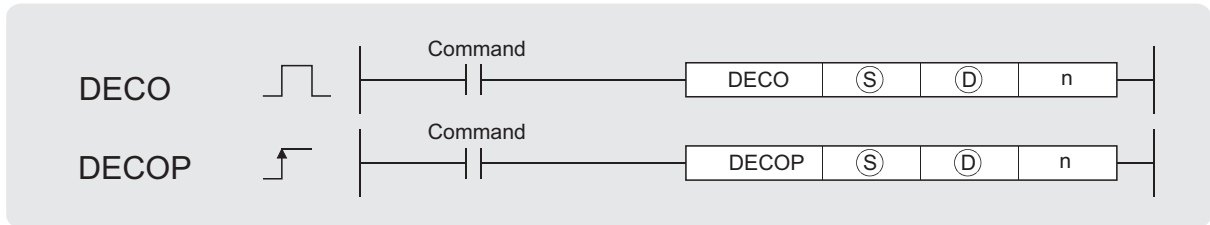


Stores the total number of bits where 1 is set into D0.



### 7.5.3 Decoding from 8 to 256 bits (DECO(P))

Basic High performance Process Redundant Universal



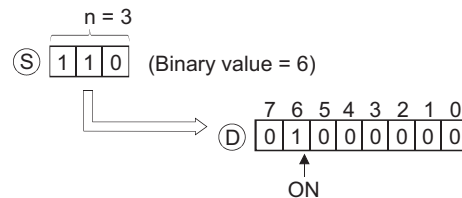
(S) : Data to be decoded or the number of the device where the data to be decoded is stored (BIN 16 bits)  
 (D) : Head number of the devices where the decoding result will be stored (Device name)  
 n : Valid bit length (1 to 8), 0: No processing (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
(S)		○				○		○	—
(D)		○				—		—	—
n		○				○		○	—

7

#### ★ Function

- (1) Turns ON the bit position of (D), which corresponds to the binary value designated by the lower n bits at (S).



- (2) The value of n can be designated between 1 and 8.
- (3) No processing is conducted if n = 0, and there are no changes in the bits 2<sup>n</sup> from the device designated at (D).
- (4) Bit devices are treated as 1 bit, and word devices as 16 bits.

7.5 Data processing instructions  
 7.5.3 Decoding from 8 to 256 bits (DECO(P))



## Operation Error

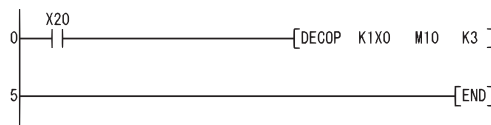
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of n is not in the 0 to 8 range. (Error code: 4100)
  - The range 2n bits from Ⓓ exceeds the range of the relevant device. (Error code: 4101)



## Program Example

- (1) The following program decodes the 3 bits from X0 and stores the results at M10 when X20 is ON.

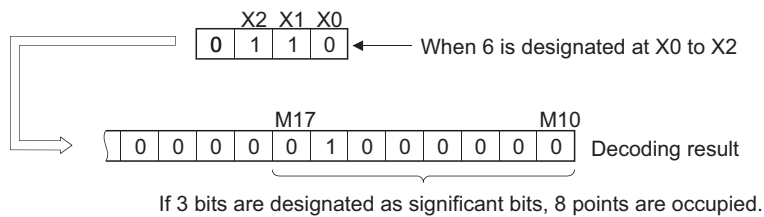
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	DECOP	K1X0 M10 K3
5	END	

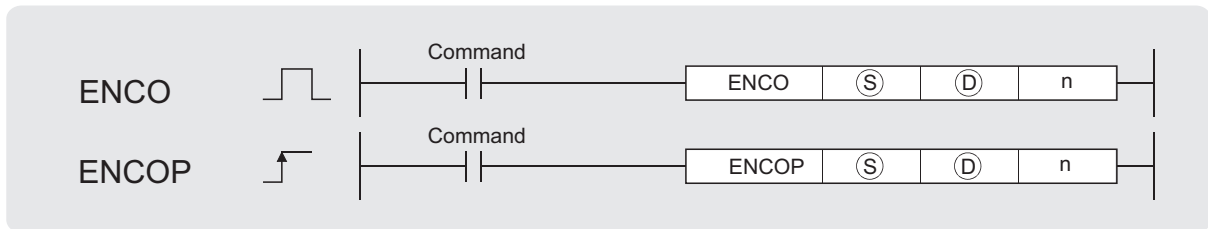
[Operation]





## 7.5.4 Encoding from 256 to 8 bits (ENCO(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the device where the data to be encoded is stored (Device name)

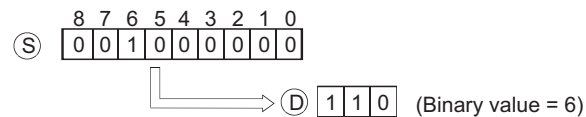
Ⓣ : Number of the device where the encoding result will be stored (BIN 16 bits)

n : Valid bit length (1 to 8), 0: No processing (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ		○				—		—	—
Ⓣ		○				○		—	—
n		○				○		○	—

### ★ Function

- (1) Stores the binary value corresponding to the bits which are "1" included in the  $2^n$ -bit data of Ⓢ to Ⓣ.



- (2) The value of n can be designated at between 1 and 8.  
 (3) If n = 0, there will be no operation, and the contents of Ⓣ will not change.  
 (4) Bit devices are treated as 1 bit, and word devices as 16 bits.  
 (5) If more than 1 bit is at 1, processing will be conducted at the upper bit location.

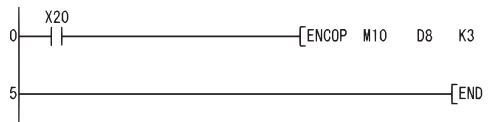
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of n is not in the 0 to 8 range. (Error code: 4100)
  - The range  $2^n$  bits from (S) exceeds the range of the relevant device. (Error code: 4101)
  - All data  $2^n$  bits from (S) is "0". (Error code: 4100)

## Program Example

- (1) The following program encodes the 3 bits from M10 when X20 is ON, and stores the results at D8.

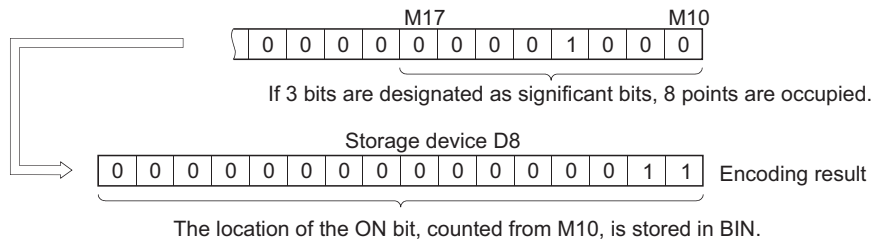
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	ENCO	M10 D8 K3
5	END	

[Operation]



# 7.5.5 7-segment decode (SEG(P))

Basic High performance Process Redundant Universal



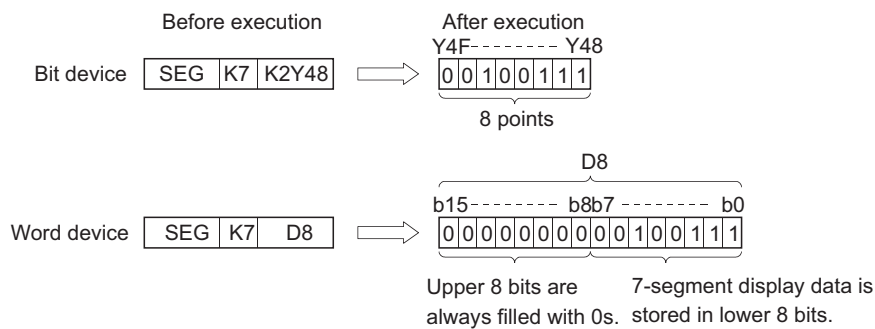
Ⓢ : Data to be decoded or head number of the devices where the data to be decoded is stored (BIN 16 bits)  
 Ⓣ : Head number of the devices where the decoding result will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

7

## ★ Function

- Decodes the data from 0 to F designated by the lower 4 bits of Ⓢ to 7-segment display data, and stores at Ⓣ.
- If Ⓣ is a bit device, indicates the head number of the devices storing the 7-segment display data; if it is a word device, indicates the number of the device storing the data.



## ! Operation Error

- There are no operation errors associated with the SEG(P) instruction.

7.5 Data processing instructions  
7.5.5 7-segment decode (SEG(P))

7-segment decode display

S		Configuration of 7 Segments	D								Display Data
Hexa-decimal	Bit Pattern		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	1	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010		0	1	1	1	0	1	1	1	A
B	1011		0	1	1	1	1	1	0	0	b
C	1100		0	0	1	1	1	0	0	1	c
D	1101		0	1	0	1	1	1	1	0	d
E	1110		0	1	1	1	1	0	0	1	e
F	1111		0	1	1	1	0	0	0	1	F

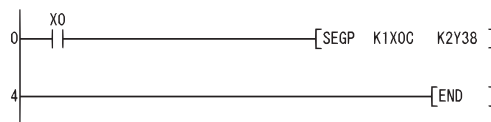


{ Head number of bit device  
 { Lowest bit of word device

Program Example

- (1) The following program converts the data from XC to XF to 7-segment display data and outputs it to Y38 to Y3F when X0 is turned ON.

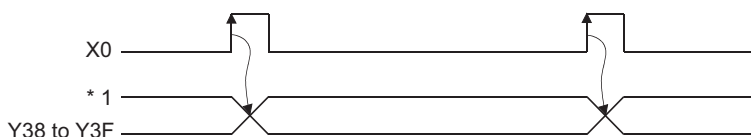
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	SEGP	K1X0C K2Y38
4	END	

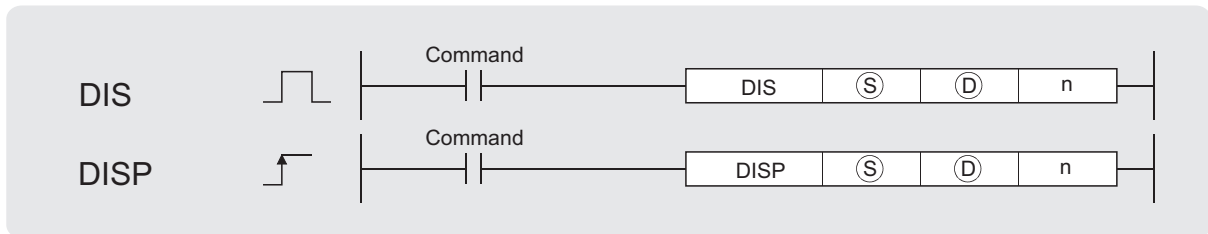
[Timing Chart]



\*1: The data Y38 to Y3F will not change until the next data is output.

## 7.5.6 4-bit dissociation of 16-bit data (DIS(P))

Basic High performance Process Redundant Universal



(S) : Head number of the devices where data to be dissociated is stored (BIN 16 bits)

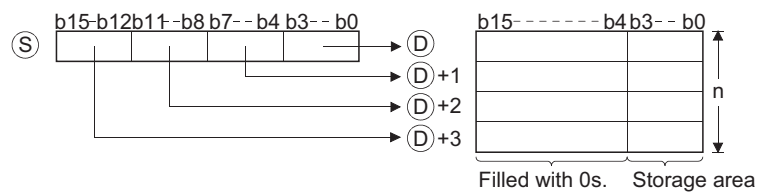
(D) : Head number of the devices where the dissociated data will be stored (BIN 16 bits)

n : Number of dissociations (1 to 4), 0: No processing (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
(S)	○	○				○			—
(D)	—	○				—			—
n	○	○				○			—

### ★ Function

- (1) Stores the lower n-digits (1 digit is 4 bits) of the 16-bit data designated by (S) at the lower 4 bits n-points from the device designated by (D).



- (2) The upper 12 bits n-points from the device designated by (S) become 0.
- (3) The value of n can be designated at between 1 and 4.
- (4) If  $n = 0$ , there will be no processing, and the contents n-points from (D) will not change.

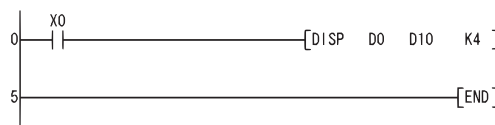
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The range n-points from  $\textcircled{D}$  exceeds the relevant device. (Error code: 4101)
  - The value of n is outside the 0 to 4 range. (Error code: 4100)

## Program Example

- (1) The following program dissociates the 16-bit data from D0 into 4-bit groups, and stores from D10 to D13 when X0 is ON.

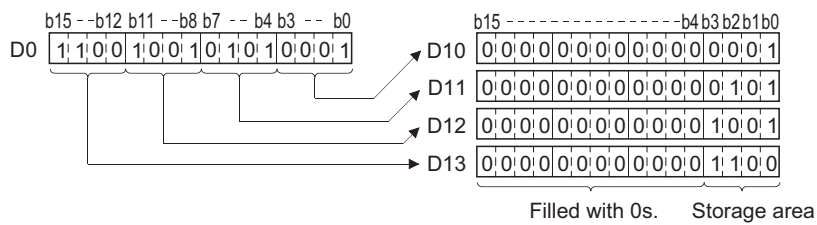
[Ladder Mode]



[List Mode]

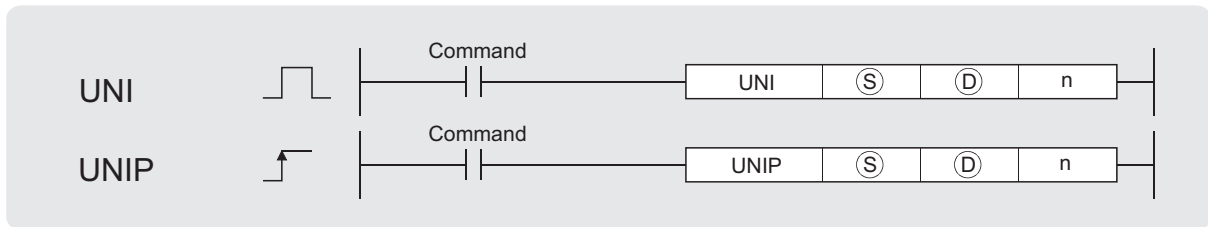
Step	Instruction	Device
0	LD	X0
1	DISP	D0 D10 K4
5	END	

[Operation]



## 7.5.7 4-bit linking of 16-bit data (UNI(P))

Basic High performance Process Redundant Universal

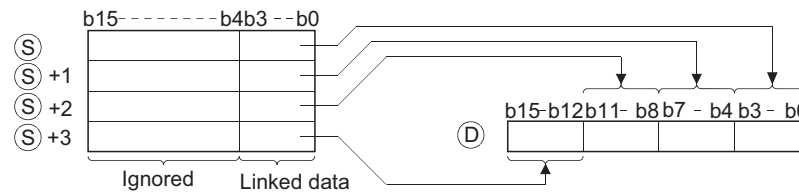


- Ⓢ : Head number of the devices where data to be linked is stored (BIN 16 bits)  
 Ⓣ : Head number of the devices where the linked data will be stored (BIN 16 bits)  
 n : Number of links (1 to 4), 0: No processing (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		—	—
Ⓣ	○	○				○		—	—
n	○	○				○		○	—

### ★ Function

- (1) Links lower 4 bits of 16-bit data n-points from device designated by Ⓢ to 16-bit device designated by Ⓣ.



- (2) The bits of the upper (4 - n) digits of the device designated by Ⓣ become 0.  
 (3) The value of n can be designated at between 1 and 4.  
 (4) If n = 0, there will be no processing, and the contents of device Ⓣ will not change.



## Operation Error

(1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

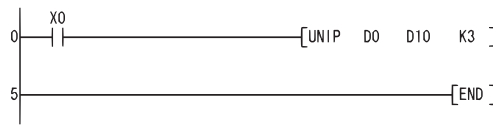
- The range n-points from (S) exceeds the relevant device. (Error code: 4101)
- The value of n is outside the 0 to 4 range. (Error code: 4100)



## Program Example

(1) The following program links the lower 4 bits of D0 to D2 when X0 is ON, and stores them at D10.

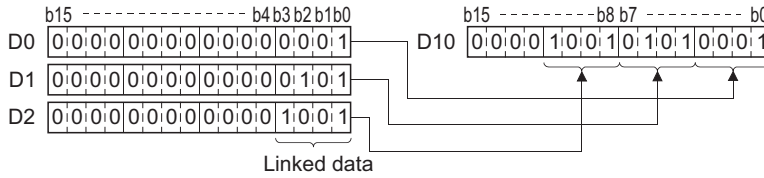
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	UNIP	D0 D10 K3
5	END	

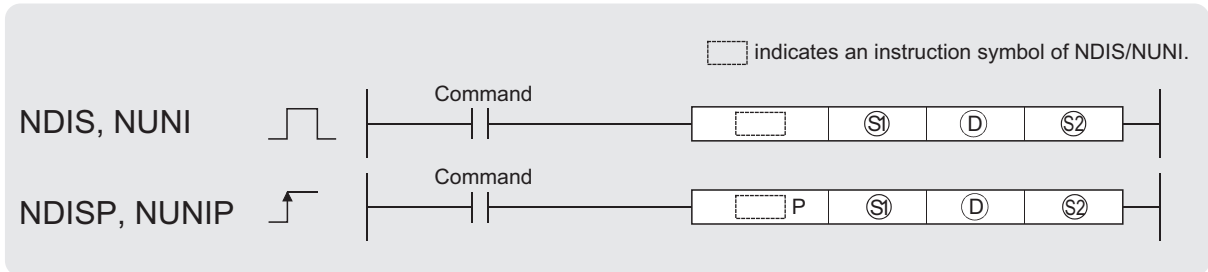
[Operation]





# 7.5.8 Dissociation or linking of random data (NDIS(P),NUNI(P))

Basic High performance Process Redundant Universal



- Ⓢ<sub>1</sub>: Head number of the devices where data to be dissociated/linked is stored (BIN 16 bits)
- Ⓢ<sub>2</sub>: Head number of the devices where the dissociated/linked data will be stored (BIN 16 bits)
- Ⓢ<sub>2</sub>: Head number of the devices where the units of dissociation/linking will be stored (BIN 16 bits)

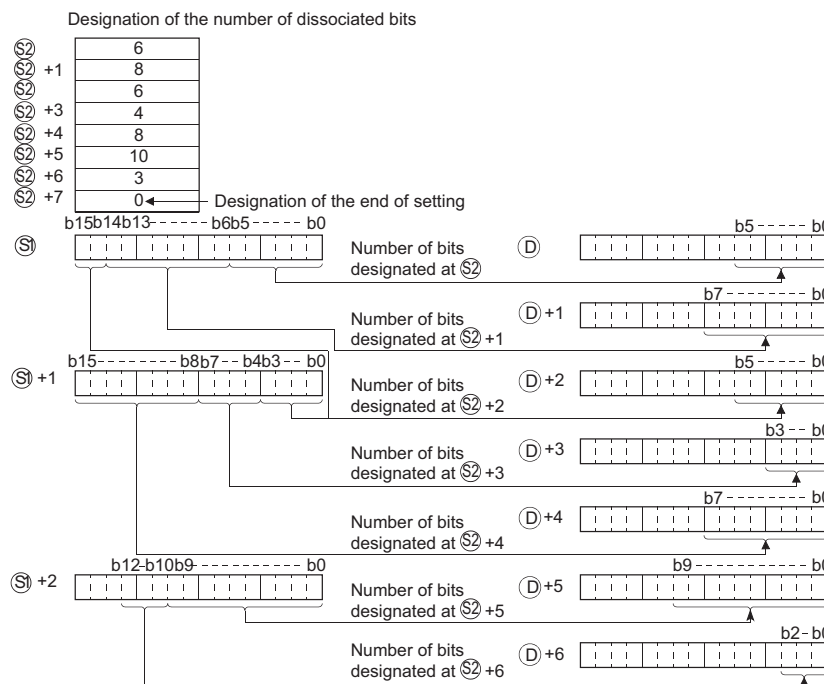
Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○					—		
Ⓢ <sub>2</sub>	—	○					—		
Ⓢ <sub>2</sub>	—	○					—		

7

## ★ Function

### NDIS

- (1) Dissociates data stored in device numbers starting from that designated at Ⓢ<sub>1</sub> into the number of individual bits designated at Ⓢ<sub>2</sub>, and stores this data in device numbers starting from that designated at Ⓢ<sub>2</sub>.

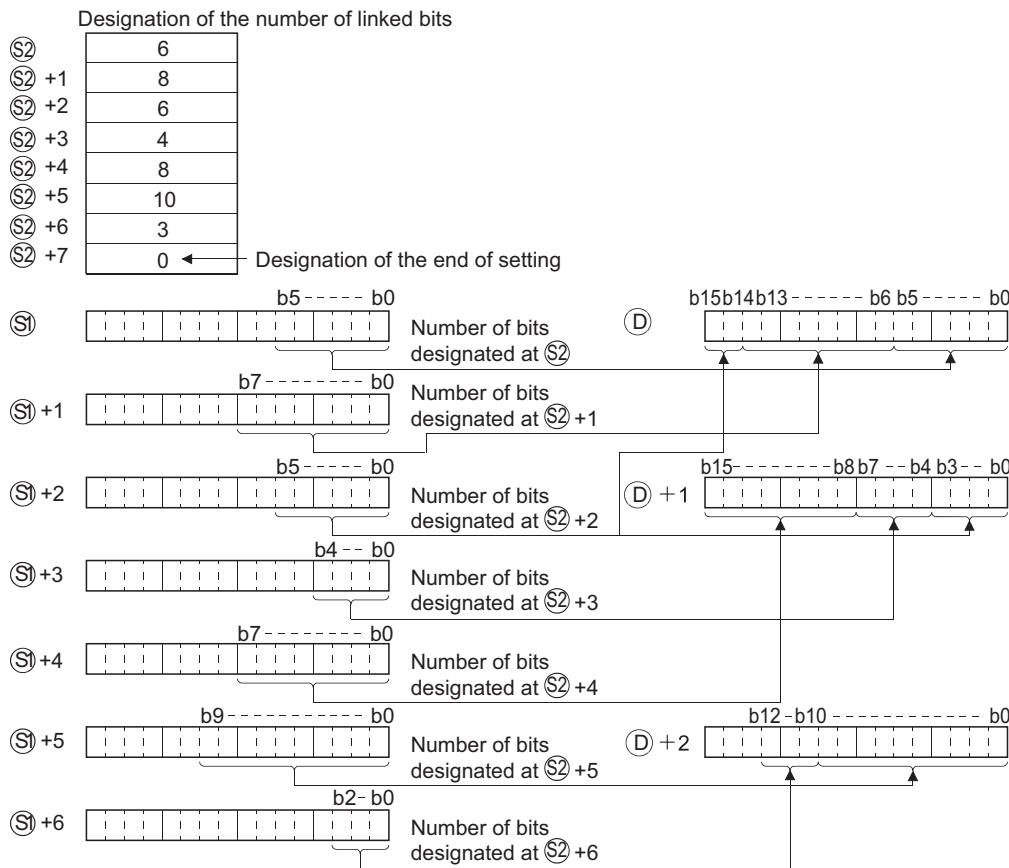


7.5.8 Dissociation or linking of random data (NDIS(P),NUNI(P))

- (2) The number of dissociated bits designated at  $\textcircled{S2}$  can be designated within a range of 1 to 16 bits.
- (3) Bits from the device number designated at  $\textcircled{S2}$  to the device number where "0" is stored are processed as dissociated bits.
- (4) Do not overlap the device range for data to be dissociated( $\textcircled{S1}$  to end range of  $\textcircled{S1}$ ) with the device range which stores the dissociated data ( $\textcircled{D}$  to end range of  $\textcircled{D}$ ). If overlapped, the correct operation result may not be obtained.
- (5) Do not specify the same device number for  $\textcircled{S1}$ ,  $\textcircled{S2}$ , and  $\textcircled{D}$ . If the same device is specified for  $\textcircled{S1}$ ,  $\textcircled{S2}$ , and  $\textcircled{D}$ , the operation does not work correctly.

**NUNI**

- (1) Links individual bits of data stored into the area starting from the device number designated by  $\textcircled{S1}$  in the number of bits specified by  $\textcircled{S2}$ , and stores them following the device number designated by  $\textcircled{D}$ .



- (2) The number of bits to be linked as designated by  $\textcircled{S2}$  can be within a range of from 1 to 16.
- (3) Processing will be performed on the number of bits to be linked from the device number designated by  $\textcircled{S2}$  to the device number storing "0".
- (4) Do not overlap the device range for data to be linked( $\textcircled{S1}$  to end range of  $\textcircled{S1}$ ) with the device range which stores the linked data ( $\textcircled{D}$  to end range of  $\textcircled{D}$ ). If overlapped, the correct operation result may not be obtained.
- (5) Do not overlap the device numbers to be designated at  $\textcircled{S1}$ ,  $\textcircled{S2}$  and  $\textcircled{D}$ . If overlapped, correct operation is not possible.

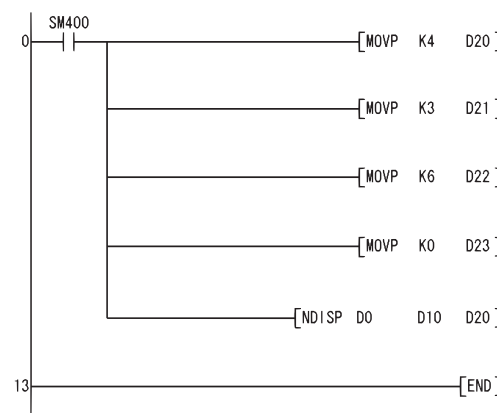
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The number of bits to be dissociated or linked as specified by  $\textcircled{S2}$ , or the device use range specified by  $\textcircled{S1}$  or  $\textcircled{D}$  exceeds the final device number of their respective devices. (Error code: 4101)
  - The number of bits for dissociation or linking specified by  $\textcircled{S2}$  has not been set within the range of from 1 to 16 bits. (Error code: 4100)

## Program Example

- (1) The following program dissociates data of 4, 3, and 6 bits respectively from the lower bits of D0, and stores them from D10 to D12.

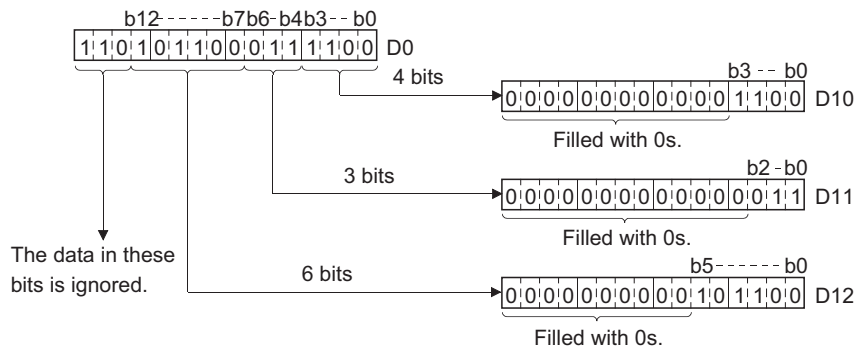
[Ladder Mode]



[List Mode]

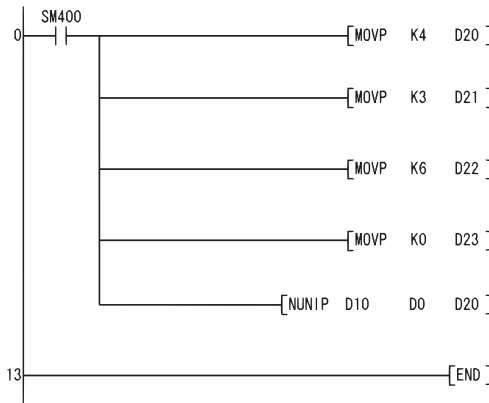
Step	Instruction	Device
0	LD	SM400
1	MOVP	K4 D20
3	MOVP	K3 D21
5	MOVP	K6 D22
7	MOVP	K0 D23
9	NDISP	D0 D10 D20
13	END	

[Operation]



(2) The following program links the lower 4 bits of data from D10, the lower 3 bits of data from D11, and the lower 6 bits of data from D12, and stores at D0.

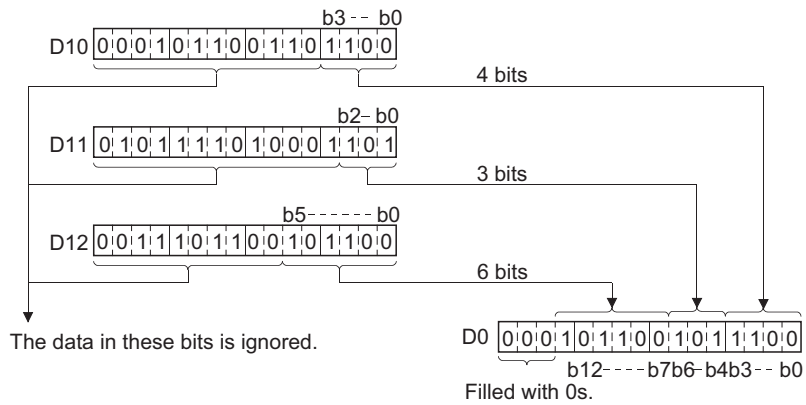
[Ladder Mode]



[List Mode]

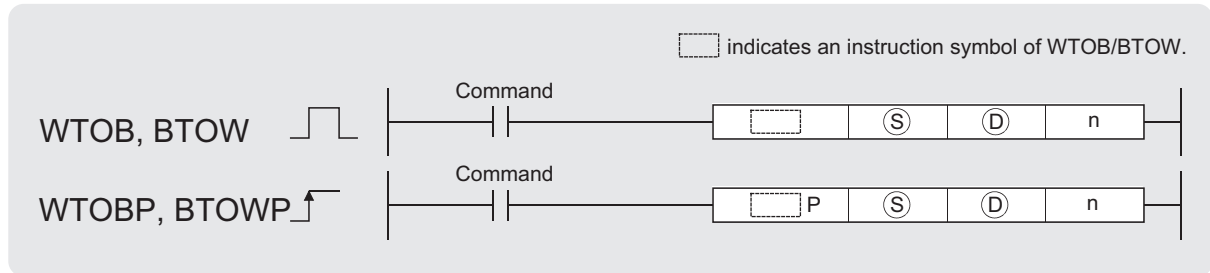
Step	Instruction	Device
0	LD	SM400
1	MOV	K4 D20
3	MOV	K3 D21
5	MOV	K6 D22
7	MOV	K0 D23
9	NUNIP	D10 D0 D20
13	END	

[Operation]



## 7.5.9 Data dissociation and linking in byte units (WTOB(P),BTOW(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the devices where data to be dissociated/linked in byte units is stored (BIN 16 bits)

Ⓣ : Head number of the devices where the result of dissociated/linking in byte units will be stored (BIN 16 bits)

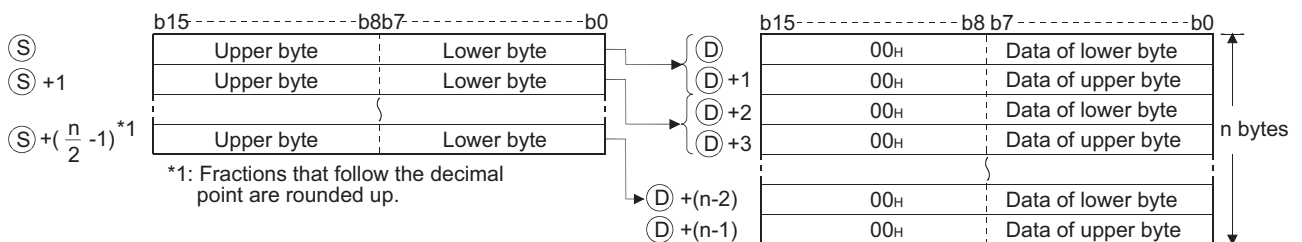
n : Number of byte data to be dissociated/linked (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
Ⓣ	—	○				—			—
n	○	○				○			—

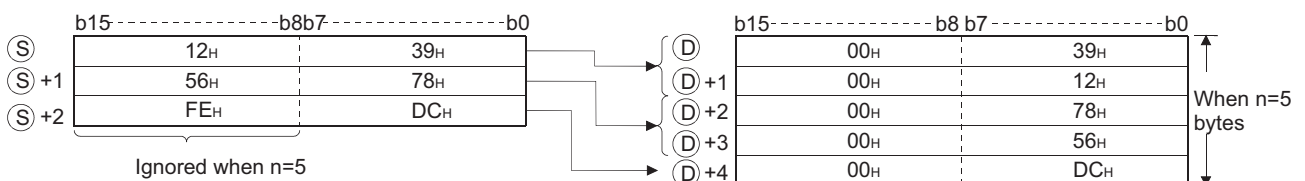
### ★ Function

#### WTOB

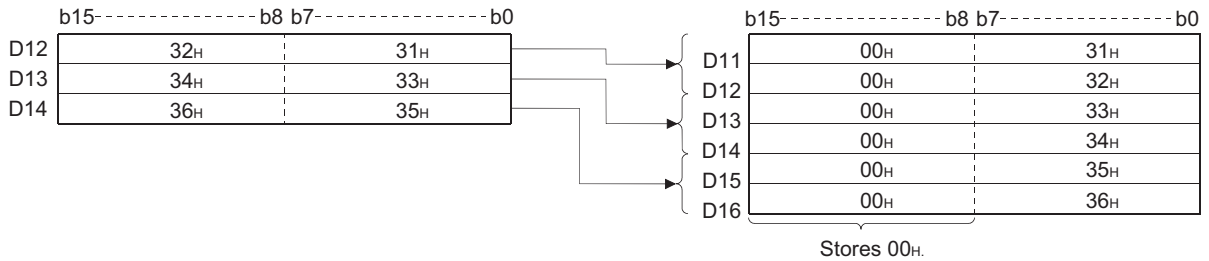
- (1) Dissociates n-bytes of the 16-bit data stored into the area starting from the device number designated by Ⓢ, and stores them following the device designated by Ⓣ.



For example, if  $n = 5$ , data through the lower 8 bits of Ⓢ to (Ⓢ + 2) would be stored from (Ⓣ) to (Ⓣ + 4).



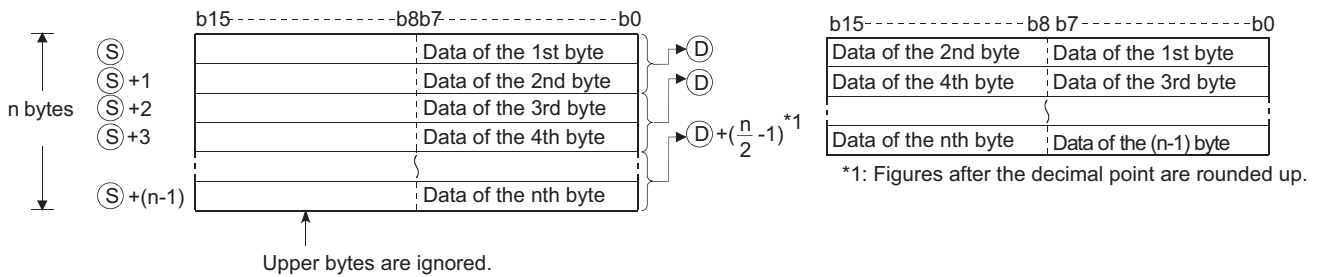
- (2) Setting the number of bytes with n automatically determines the range of the 16-bit data designated by (S) and the range of the devices to store the byte data designated by (D).
- (3) No processing will be conducted when the number of bytes designated by n is "0".
- (4) The "00H" code will automatically be stored at the upper 8 bits of the byte storage device designated by (D).



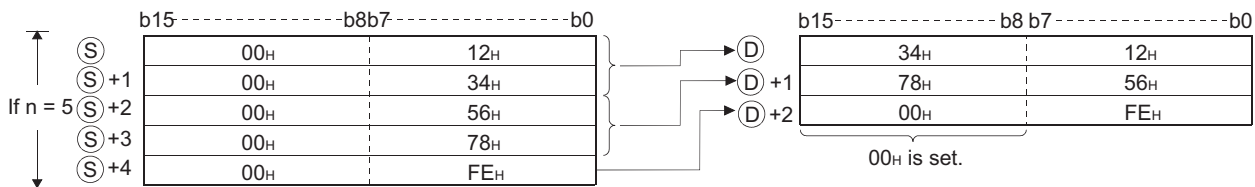
- (5) Even though the range of the device with the data to be divided ((S) to (S) + ((n/2) - 1)) is the same as the range of the device with the divided data ((D) to (D) + (n-1)), the instruction operates correctly.

**BTOW**

- (1) Links the lower 8 bits of the 16-bit data in n words stored in the area starting from the device designated by (S) in 1-word units and stores it into the area starting from the device designated by (D). The upper 8 bits of n-word data stored in the area starting from the device designated by (S) will be ignored. Further, if n is an odd number, 0 is stored at the upper 8 bits of the device where the nth byte data is stored.



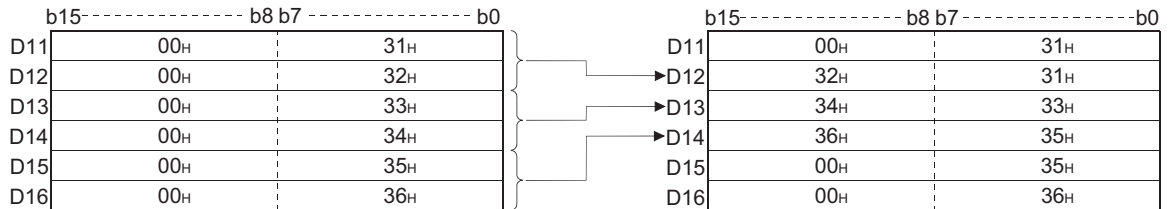
For example, if n = 5, the lower 8 bits of data from (S) to ((S) + 4) are linked and stored at (D) to ((D) + 2).



- (2) Setting the number of bytes with n automatically determines the range of the byte data designated by (S) and the range of the devices to store the linked data designated by (D).
- (3) No processing will be conducted when the number of bytes designated by n is "0".

- (4) The upper 8 bits of the byte storage device designated by  $\textcircled{S}$  are ignored, and the lower 8 bits are used.
- (5) Linking is correctly processed even when the device range ( $\textcircled{S}$  to  $\textcircled{S} + (n-1)$ ) where the data to be linked is stored overlaps with the device range ( $\textcircled{D}$  to  $\textcircled{D} + (\frac{n}{2} - 1)$ ) where the linked data will be stored.

For example, the following will take place in a case where the lower 8 bits of D11 to D16 are to be stored at D12 to D14:



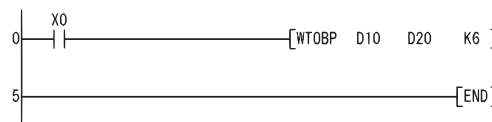
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The range of the number of bytes designated by n following the device number designated by  $\textcircled{S}$  exceeds the relevant device range. (Error code: 4101)
  - The range of the number of bytes designated by n following the device number designated by  $\textcircled{D}$  exceeds the relevant device range. (Error code: 4101)

## Program Example

- (1) The following program dissociates the data at D10 to D12 in byte units and stores it at D20 to D25 when X0 is turned ON.

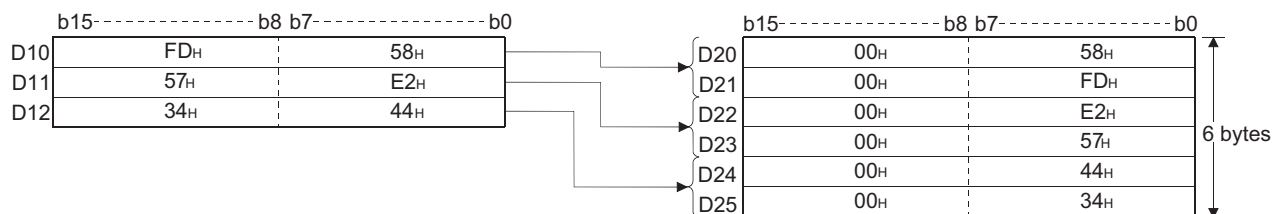
[Ladder Mode]



[List Mode]

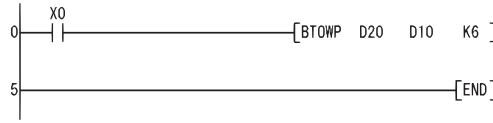
Step	Instruction	Device
0	LD	X0
1	WTOBP	D10 D20 K6
5	END	

[Operation]



(2) The following program links the lower 8 bits of data from D20 through D25 and stores the result at D10 to D12 when X0 is turned ON.

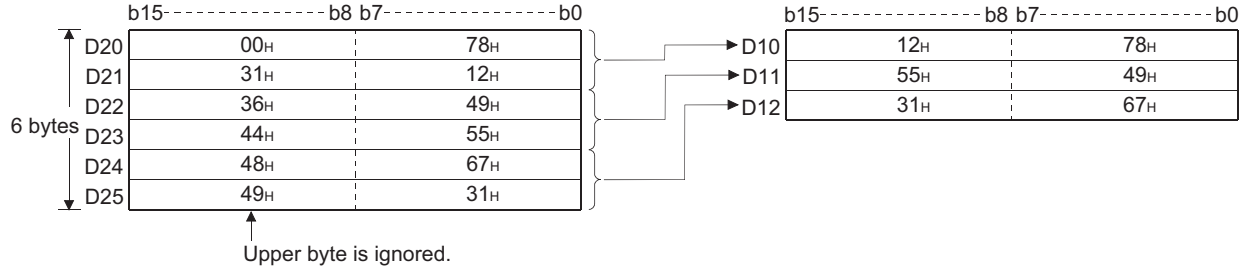
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	BTOWP	D20 D10 K6
5	END	

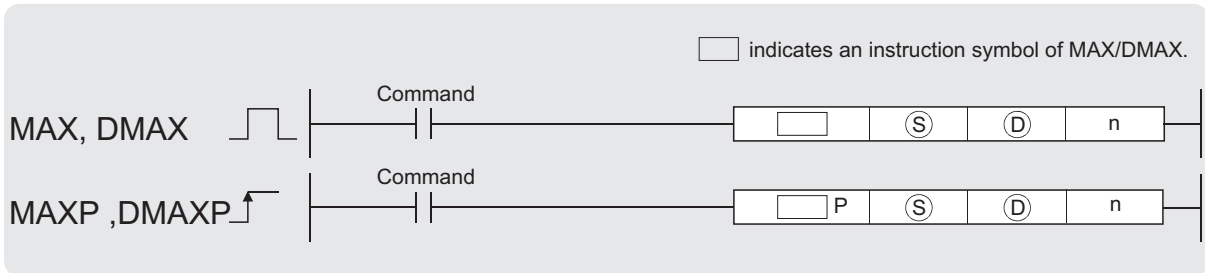
[Operation]





# 7.5.10 Maximum value search for 16- and 32-bit data (MAX(P),DMAX(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the devices where a maximum value is searched (BIN 16/32 bits)  
 Ⓣ : Head number of the devices where the maximum value search result will be stored (BIN 16/32 bits)  
 n : Number of data blocks to be searched (BIN 16 bits)

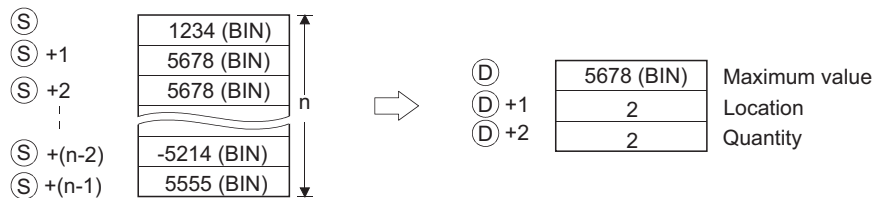
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
Ⓣ	—	○				—			—
n	○	○				○			—

7

## ★ Function

### MAX

- Searches in the n points of 16-bit BIN data, from the device designated by Ⓢ, for the maximum value and stores the searched maximum value at the device designated by Ⓣ. Starts the search from the device designated by Ⓢ and stores the location, specified in the number of points counted from Ⓢ, of the device where the maximum value is found first at Ⓣ+1 and stores the number of the found minimum values at Ⓣ+2.

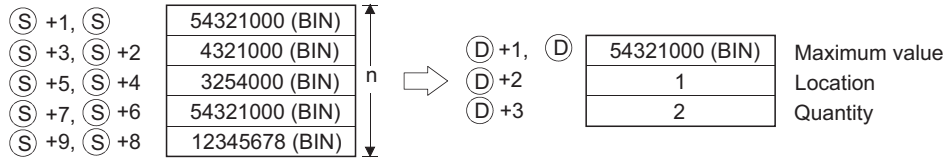


7.5 Data processing instructions  
 7.5.10 Maximum value search for 16- and 32-bit data (MAX(P),DMAX(P))

## DMAX

- (1) Searches in the n points of 32-bit BIN data, from the device designated by (S), for the maximum value and stores the searched maximum value at the device designated by (D) and (D) +1.

Starts the search from the device designated by (S) and stores the location, specified in the number of points counted from (S), of the device where the maximum value is found first at (D) +2 and stores the number of the found minimum values at (D) +3.



## Operation Error

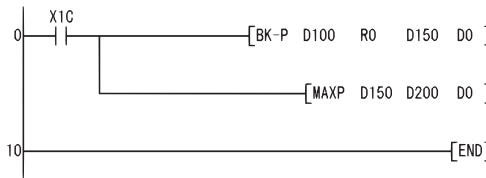
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The n-bit range from the (S), or device exceeds the range of that device. (Error code: 4101)
  - The device specified by (D) exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)



## Program Example

- (1) The following program subtracts, when X1C is turned ON, the data stored at D100 to D103 from the data stored at R0 to R3, and searches in the results of subtraction for the maximum value, then, stores it at D200 to D202.

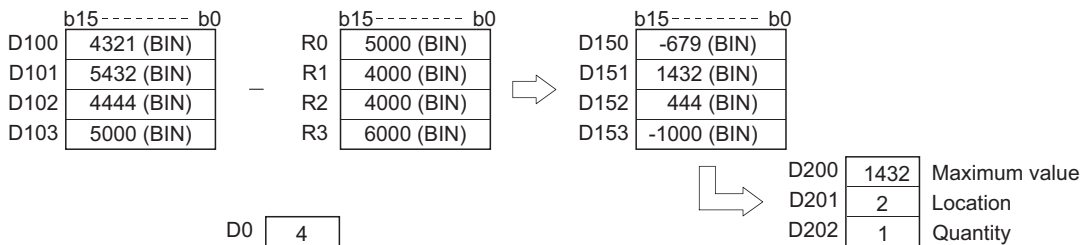
[Ladder Mode]



[List Mode]

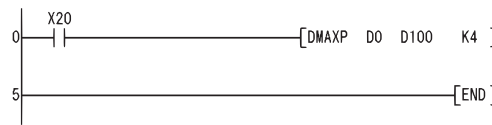
Step	Instruction	Device
0	LD	X1C
1	BK-P	D100 R0 D150 D0
6	MAXP	D150 D200 D0
10	END	

[Operation]



- (2) The following program searches for the maximum value from the 32-bit data at D0 to D7, and stores it at D100 to D103 when X20 is turned ON.

[Ladder Mode]



[List Mode]

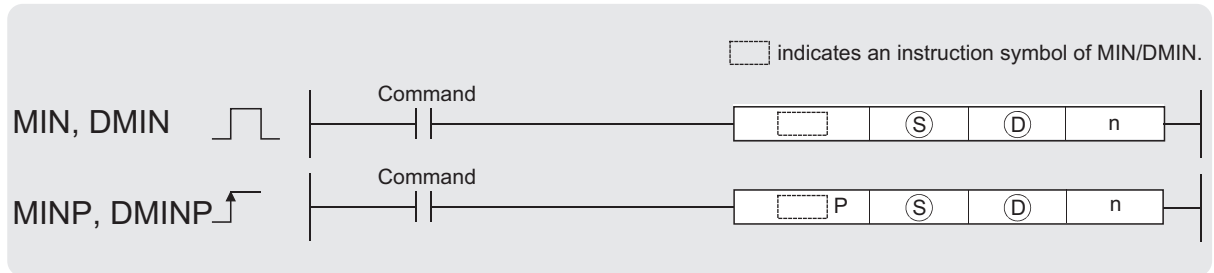
Step	Instruction	Device
0	LD	X20
1	DMAXP	D0 D100 K4
5	END	

[Operation]

D1, D0	3786213 (BIN)	⇒	D101, D100	8744740
D3, D2	-3235 (BIN)		D102	3
D5, D4	8744740 (BIN)		D103	1
D7, D6	7141821 (BIN)			

# 7.5.11 Minimum value search for 16- and 32-bit data (MIN(P),DMIN(P))

Basic High performance Process Redundant Universal



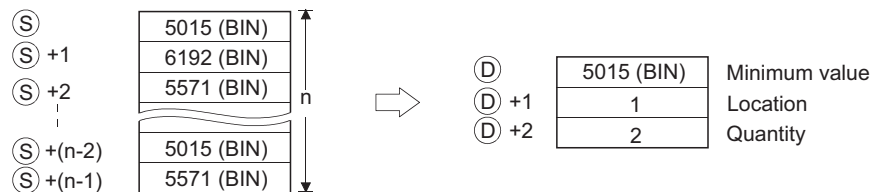
Ⓢ : Head number of the devices where a minimum value is searched (BIN 16/32 bits)  
 Ⓣ : Head number of the devices where the minimum value search result will be stored (BIN 16/32 bits)  
 n : Number of data blocks to be searched (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
Ⓣ	—	○				—			—
n	○	○				○			—

## ★ Function

### MIN

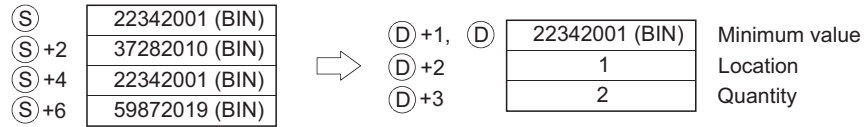
- Searches in the n points of 16-bit BIN data, from the device designated by Ⓢ, for the minimum value and stores searched minimum value at the device designated by Ⓣ. Starts the search from the device designated by Ⓢ and stores the location, specified in the number of points counted from Ⓢ, of the device where the minimum value is found first at Ⓣ+1 and stores the number of the found minimum values at Ⓣ+2.



## DMIN

- (1) Searches in the n points of 32-bit BIN data, from the device designated by (S), for the minimum value and stores searched minimum value at the devices designated by (D) and (D)+1.

Starts the search from the device designated by (S) and stores the location, specified in the number of points counted from (S), of the device where the minimum value is found first at (D)+2 and stores the number of the found minimum values at (D)+3.



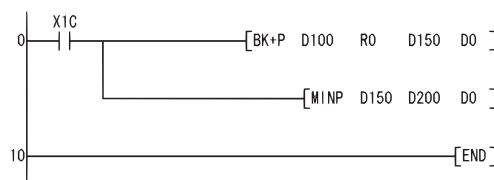
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The n-bit range from the device specified by (S) exceeds the range of the corresponding device. (Error code: 4101)
  - The device specified by (D) exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program adds, when X1C is turned ON, the data stored at D100 to D103 and the data stored at R0 to R3, and searches in the results of addition for the minimum value, then, stores it at D200 to D202.

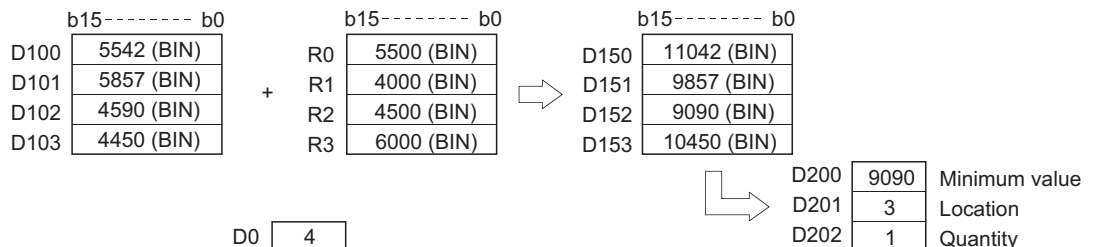
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	BK+P	D100 R0 D150 D0
6	MINP	D150 D200 D0
10	END	

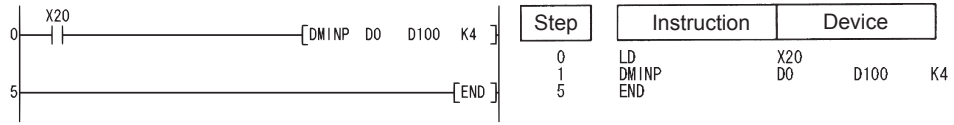
[Operation]



(2) The following program, when X20 is turned ON, searches for the minimum value from the 32-bit data contained from D0 to D7, and stores it from D100 to D103.

[Ladder Mode]

[List Mode]

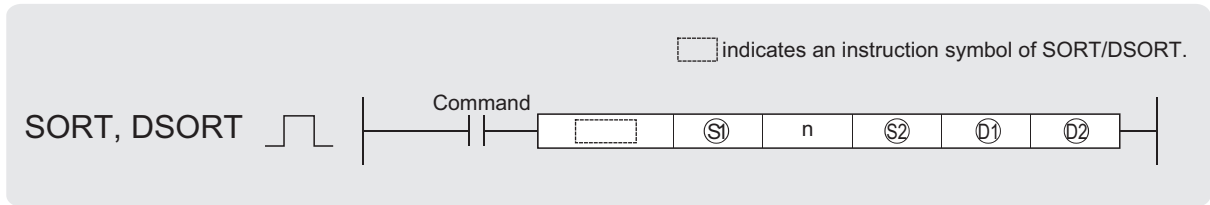


[Operation]

D1,D0	57020175 (BIN)	➔	D101,D100	-69386
D3,D2	2070166 (BIN)		D102	4
D5,D4	3596045 (BIN)		D103	1
D7,D6	-69386 (BIN)			

# 7.5.12 BIN 16 and 32 bits data sort operations (SORT,DSORT)

Basic High performance Process Redundant Universal



- Ⓢ<sub>1</sub> : Head device number in the table to be sorted (BIN 16/32 bits)
- n : Number of data blocks to be sorted (BIN 16 bits)
- Ⓢ<sub>2</sub> : Number of data blocks to be compared in one sort operation (BIN 16 bits)
- ⓓ<sub>1</sub> : Number of the bit device to be turned ON at the completion of the sort operation (bits)
- ⓓ<sub>2</sub> : Device reserved for the system (BIN 16 bits)

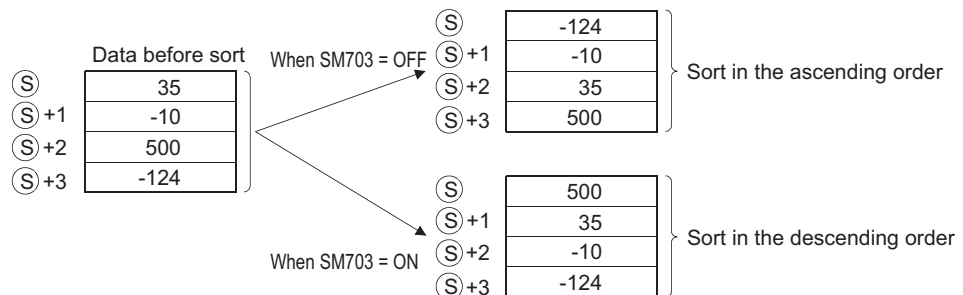
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○				—			—
n	○	○				○			—
Ⓢ <sub>2</sub>	○	○				○			—
ⓓ <sub>1</sub>	○	—				—			—
ⓓ <sub>2</sub>	—	○				—			—

7

## ★ Function

### SORT

- (1) Sorts (rearranges data) BIN 16-bit data n points from Ⓢ<sub>1</sub> in ascending or descending order. Sort order is designated by the ON/OFF status of SM703:
  - When SM703 is OFF: Ascending order sort
  - When SM703 is ON : Descending order sort



7.5.12 BIN 16 and 32 bits data sort operations (SORT,DSORT)

- (2) Several scans are required for sorts performed by the SORT instruction. The number of scans executed until completion is the value obtained by dividing the maximum number of times executed until the completion of the sort by the number of data blocks compared at one execution designated by  $\textcircled{S2}$ . (Decimal fractions are rounded up.)When the value of  $\textcircled{S2}$  is increased, the number of scans until completion of the sort is reduced, but the amount of time per scan is lengthened.
- (3) The maximum number of executions until completion of the sort should be calculated according to the following equation:

$$\text{The maximum number of executions until completion} = (n) \times (n - 1) / 2 \text{ [times executed]}$$

**Example**

When  $n=10$ , the number of executions is obtained as  $10 \times (10 - 1) / 2=45$  [times executed].

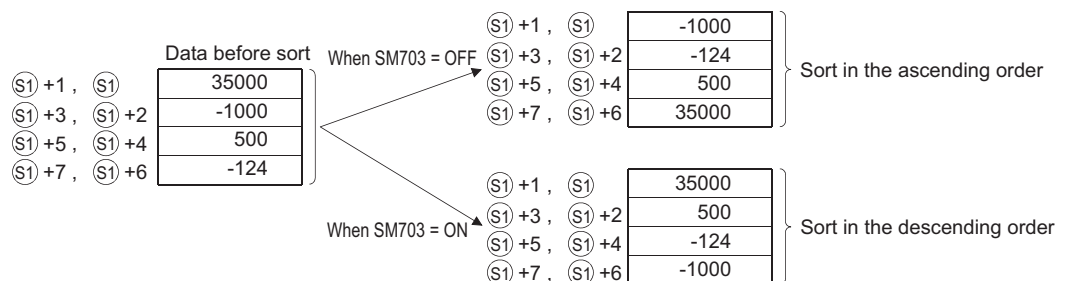
If  $\textcircled{S2}=2$ , then the number of scans until the completion of sort is calculated as  $45/2=22.5 \rightarrow 23$  [scans].

- (4) The device designated by  $\textcircled{D1}$  (the completion device) is turned OFF by the execution of the SORT instruction, and turned ON when the sort is completed. Because the device designated by  $\textcircled{D1}$  is maintained in the ON state after the completion of the sort, the user must turn it OFF if required.
- (5) The 2 points from the device designated by  $\textcircled{D2}$  are used by the system during the execution of the SORT instruction. These 2 points from the device designated by  $\textcircled{D2}$  should therefore not be used by the user.  
Changing these points may cause an error code to be returned (Error code: 4100).
- (6) If the value of  $n$  is changed during the execution of the SORT instruction, the sort will be conducted in accordance with the number of sort data blocks after the change.
- (7) If the execution command is turned OFF during the execution of the SORT instruction, the sort is suspended. The sort resumes from the beginning when the execution command is turned ON again.
- (8) To execute another sort operation immediately after the completion of the previous sort, turn OFF the execution command once, then turn it ON.

**DSORT**

- (1) Sorts (rearranges data) BIN 32-bit data  $n$  points from  $\textcircled{S1}$  in ascending or descending order. Sort order is designated by the ON/OFF status of SM703:

- When SM703 is OFF : Ascending order sort
- When SM703 is ON : Descending order sort





- (2) Several scans are required for sorts performed by the DSORT instruction. The number of scans executed until completion is the value obtained by dividing the maximum number of times executed until the completion of the sort by the number of data blocks compared at one execution designated by  $\textcircled{S2}$ . (Decimal fractions are rounded up.) When the value of  $\textcircled{S2}$  is increased, the number of scans until completion of the sort is reduced, but the amount of time per scan is lengthened.
- (3) The maximum number of executions until completion of the sort should be calculated according to the following equation:

The maximum number of executions until completion =  $(n) \times (n-1)/2$  [times executed]

#### Example

When  $n=10$ , the number of executions is obtained as  $10 \times (10-1)/2=45$  [times executed].  
If  $S2=2$ , then the number of scans until the completion of sort is calculated as  $45/2=22.5 \rightarrow 23$  [scans].

- (4) The device designated by  $\textcircled{D1}$  (the completion device) is turned OFF by the execution of the SORT instruction, and turned ON when the sort is completed. Because the device designated by  $\textcircled{D1}$  is maintained in the ON state after the completion of the sort, the user must turn it OFF if required.
- (5) The 2 points from the device designated by  $\textcircled{D2}$  are used by the system during the execution of a DSORT instruction. These 2 points from the device designated by  $\textcircled{D2}$  should therefore not be used by the user.  
Changing these points may cause an error code to be returned (Error code: 4100).
- (6) If the value of  $n$  is changed during the execution of the SORT instruction, the sort will be conducted in accordance with the number of sort data blocks after the change.
- (7) If the execution command is turned OFF during the execution of the SORT instruction, the sort is suspended. The sort resumes from the beginning when the execution command is turned ON again.
- (8) To execute another sort operation immediately after the completion of the previous sort, turn OFF the execution command once, then turn it ON.



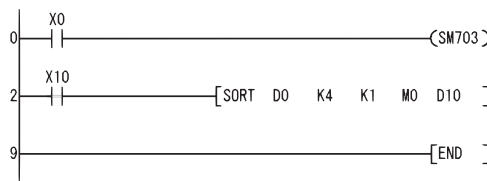
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- For the SORT(P) instruction, the range for  $n$  points starting from the device at  $\textcircled{S1}$  exceeds the corresponding device range. (Error code: 4101)
  - For the DSORT(P) instruction, the range for  $2 \times n$  points starting from the device at  $\textcircled{S1}$  exceeds the corresponding device range. (Error code: 4101)
  - The device range of the  $(n/2 \times n)$  points starting from the device designated by  $\textcircled{S1}$  overlaps with the device range of the 2 points starting from the device designated by  $\textcircled{D2}$ . (Error code: 4101)
  - $\textcircled{S2}$  is 0 or is a negative value. (Error code: 4100)

## Program Example

- (1) The following program sorts the BIN 16-bit data in 10 points from D0 in the ascending/descending order when X10 is turned ON.

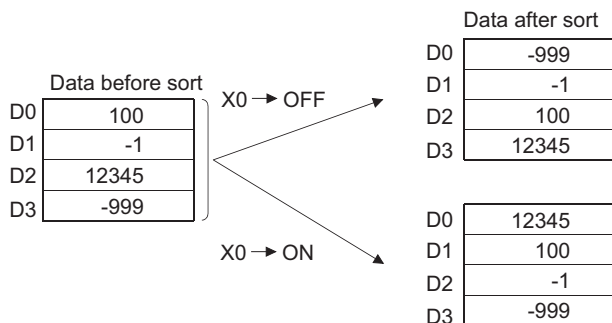
[Ladder Mode]



[List Mode]

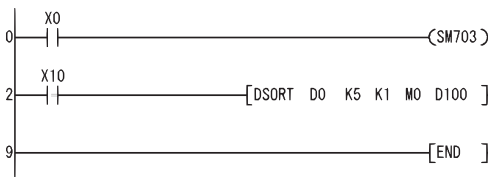
Step	Instruction	Device
0	LD	X0
1	OUT	SM703
2	LD	X10
3	SORT	D0 K4 K1 MO D10
9	END	

[Operation]



- (2) The following program sorts the BIN 32-bit data in 20 points from D0 in ascending/descending order when X10 is turned ON.

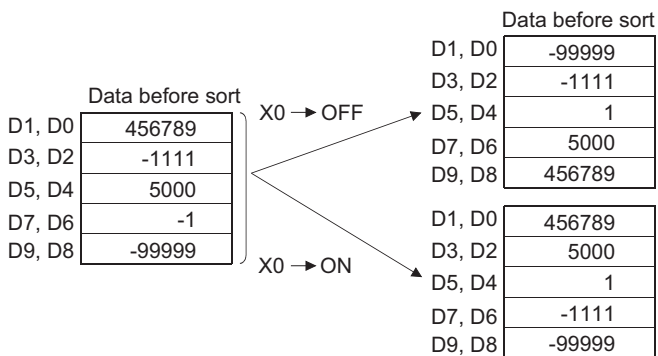
[Ladder Mode]



[List Mode]

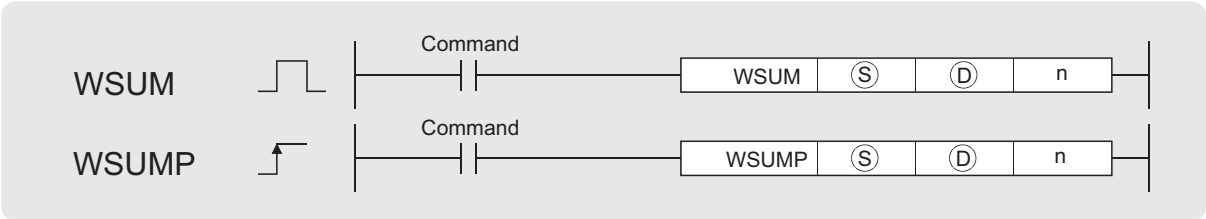
Step	Instruction	Device
0	LD	X0
1	OUT	SM703
2	LD	X10
3	DSORT	D0 K5 K1 MO D100
9	END	

[Operation]



# 7.5.13 Calculation of totals for 16-bit data (WSUM(P))

Basic High performance Process Redundant Universal



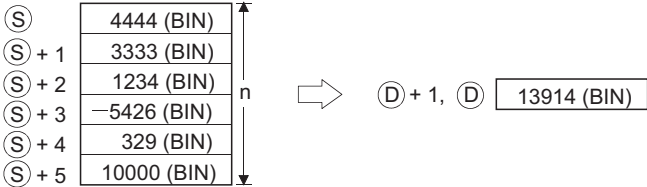
Ⓢ : Head number of the devices where data to be summed are stored (BIN 16 bits)  
 Ⓣ : Head number of the devices where the sum will be stored (BIN 32 bits)  
 n : Number of data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		—	—
Ⓣ	○	○				○		—	—
n	○	○				○		○	—

7

## ★ Function

- (1) Adds all 16-bit BIN data for n blocks from the device designated at Ⓢ, and stores it in the device designated at Ⓣ.



## ! Operation Error

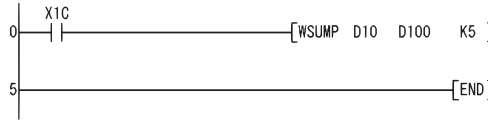
- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The n-bit range from the Ⓢ, or device exceeds the range of that device. (Error code: 4101)

7.5 Data processing instructions  
7.5.13 Calculation of totals for 16-bit data (WSUM(P))

## Program Example

- (1) The following program adds the 16-bit BIN data from D10 to D14, and stores it in D100 and D101 when X1C is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	WSUMP	D10 D100 K5
5	END	

[Operation]

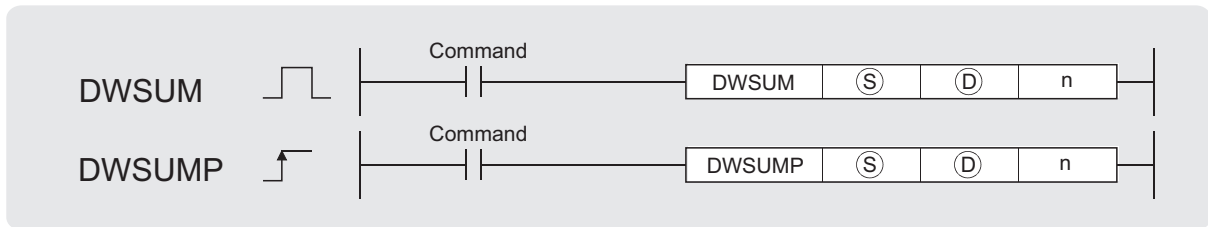
D10	4500 (BIN)
D11	2500 (BIN)
D12	-3276 (BIN)
D13	6780 (BIN)
D14	4444 (BIN)



D101,D100 14948 (BIN)

## 7.5.14 Calculation of totals for 32-bit data (DWSUM(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the devices where data to be summed are stored (BIN 32 bits)

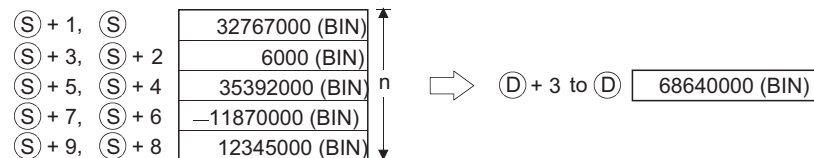
Ⓣ : Head number of the devices where the sum will be stored (BIN 64 bits)

n : Number of data blocks (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		—	—
Ⓣ	○	○				—		—	—
n	○	○			○			○	—

### ★ Function

- (1) Adds all 32-bit BIN data stored in n points of devices starting from the one designated by Ⓢ, and stores the result to 4 points of devices (4 words) starting from the one designated by Ⓣ.



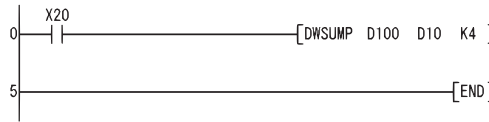
### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The n-bit range from the Ⓢ, or device exceeds the range of that device. (Error code: 4101)
  - The device specified by Ⓣ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program adds the 32-bit BIN data at D100 to D107, and stores the result at D10 and D13 when X20 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	DWSUMP	D100 D10 K4
5	END	

[Operation]

D101,D100	11245600 (BIN)
D103,D102	27543200 (BIN)
D105,D104	558800 (BIN)
D107,D106	-15675000 (BIN)



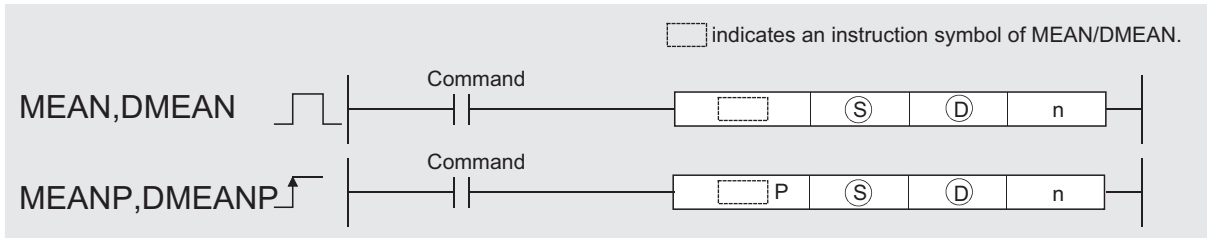
D13 to D10 

23672600 (BIN)
----------------

# 7.5.15 Calculation of averages for 16-bit or 32-bit data (MEAN(P),DMEAN(P))



- QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.
- QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



- Ⓢ : Head number of the devices where the data to be averaged are stored (BIN16/32 bits)
- Ⓣ : Head number of the devices where the average will be stored (BIN 16/32 bits)
- n : Number of data or number of the devices where the number of data are stored (Setting range: 1 to 32767) (BIN 16 bits)

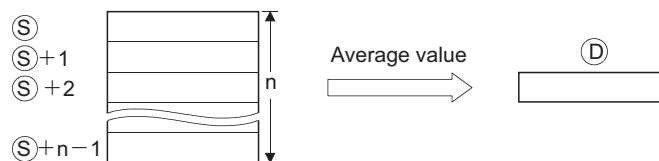
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	○			—		—	—
Ⓣ	—	○	○			—		—	—
n	—	○	○			○		○	—

7

## ★ Function

### MEAN(P)

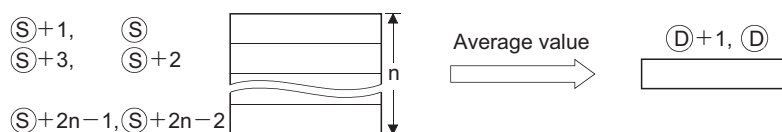
- (1) This instruction calculates the mean of 16-bit BIN data stored in n-point devices starting from the device specified by Ⓢ, and then stores the result into the device specified by Ⓣ.



- (2) If the value calculated is not integer, this instruction will drop the number of decimal places.
- (3) If the value specified by n is 0, the instruction will be not processed.

### DMEAN(P)

- (1) This instruction calculates the mean of 32-bit BIN data stored in n-point devices starting from the device specified by Ⓢ, and then stores the result into the device specified by Ⓣ.



- (2) If the value calculated is not integer, this instruction will drop the number of decimal places.
- (3) If the value specified by n is 0, the instruction will be not processed.

7.5.15 Calculation of averages for 16-bit or 32-bit data (MEAN(P),DMEAN(P))

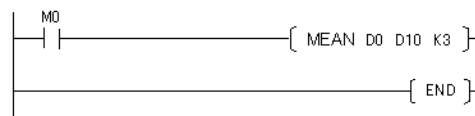
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The value specified by n is other than 0 to 32767. (Error code: 4100)
  - The range of the n-point devices starting from the device specified by  $\textcircled{S}$  exceeds the range of the devices specified by  $\textcircled{D}$ . (Error code: 4101)

## Program Example

- (1) The following program stores the average value of 16-bit data stored from D0 to D2 into D10, when M0 is turned on.

[Ladder Mode]



[List Mode]

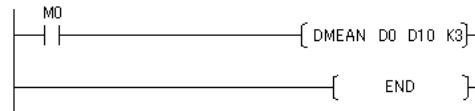
Step	Instruction	Device
0	LD	M0
1	MEAN	D0 D10 K3
5	END	

[Operation]

D0	105 (BIN)	⇒	D10	550 (BIN)
D1	555 (BIN)			
D2	990 (BIN)			

- (2) The following program stores the average value of 32-bit data stored from D0 to D5 into D10 and D11, when M0 is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	DMEAN	D0 D10 K3
5	END	

[Operation]

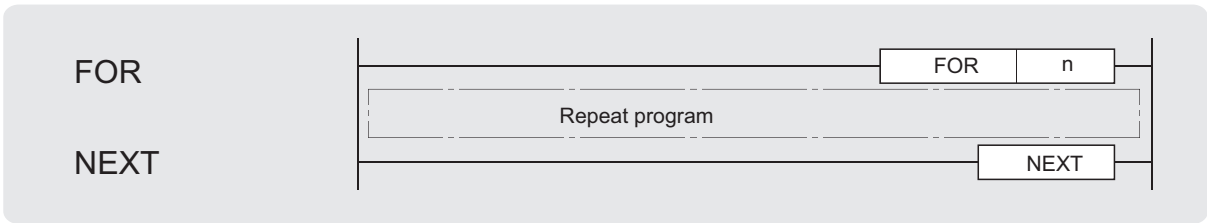
D1,D0	623541 (BIN)	⇒	D11,D10	2101176 (BIN)
D3,D2	4753647 (BIN)			
D5,D4	926342 (BIN)			



# 7.6 Structure creation instructions

## 7.6.1 FOR to NEXT instruction loop (FOR,NEXT)

Basic High performance Process Redundant Universal



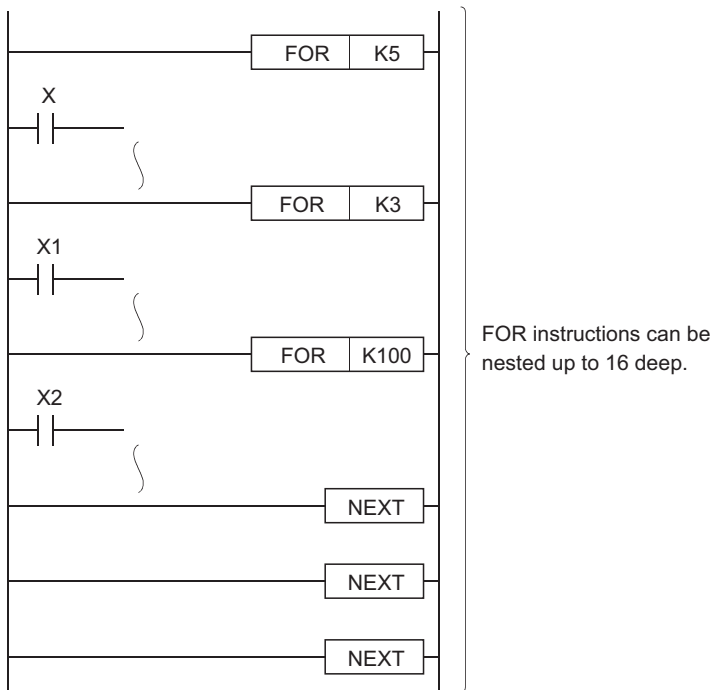
n : Number of repetitions of FOR to NEXT loop (1 to 32767) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n									—

7

### ★ Function

- (1) When the processing in the FOR to NEXT loop is executed n-times without conditions, the step following the NEXT instruction will be executed.
- (2) The value of n can be designated at between 1 and 32767. If it is designated from -32768 to 0, the processing which is executed when n=1 will be performed.
- (3) If you do not desire to execute the processing called for within the FOR to NEXT loop, use the CJ or SCJ instruction to jump.
- (4) FOR instructions can be nested up to 16 deep.



7.6 Structure creation instructions  
7.6.1 FOR to NEXT instruction loop (FOR,NEXT)

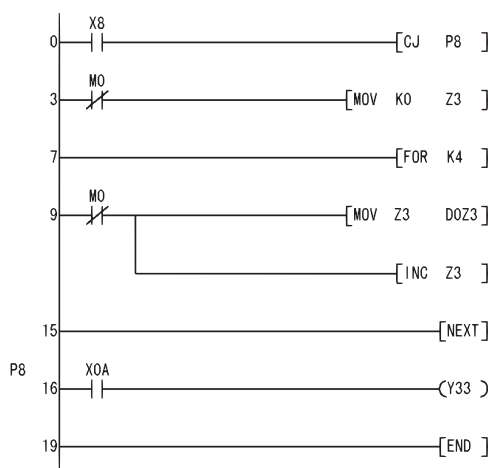
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- An END, FEND or GOEND instruction was executed before the execution of a NEXT instruction and after the execution of a FOR instruction. (Error code: 4200)
  - A NEXT instruction is executed prior to the execution of a FOR instruction. (Error code: 4201)
  - A STOP instruction has been inserted within the FOR to NEXT loop. (Error code: 4200)
  - The 17th FOR instruction is executed when FOR instructions have been nested. (Error code: 4202)

## Program Example

- (1) The following program executes the FOR to NEXT loop when X8 is OFF, and does not execute it when X8 is ON.

[Ladder Mode]

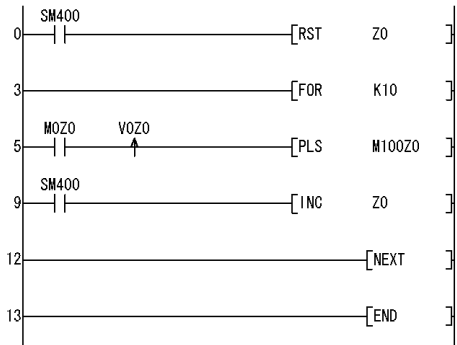


[List Mode]

Step	Instruction	Device
0	LD	X8
1	CJ	P8
3	LDI	MO
4	MOV	K0 Z3
7	FOR	K4
9	LDI	MO
10	MOV	Z3 D0Z3
13	INC	Z3
15	NEXT	
16	P8	
17	LD	X0A
18	OUT	Y33
19	END	

Remark

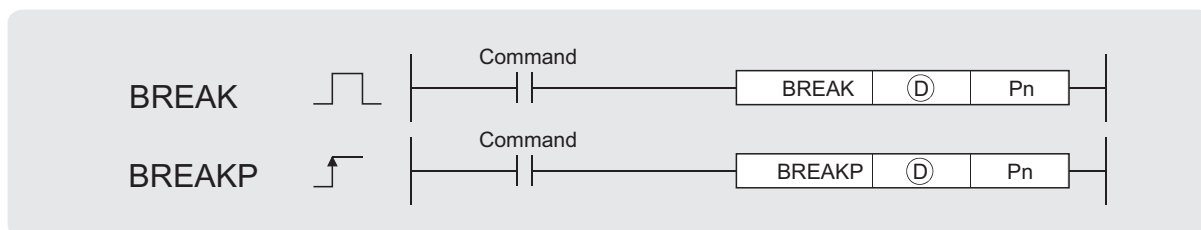
1. To force an end to the repetitious execution of the FOR to NEXT loop during the execution of the loop, insert a BREAK instruction. See 7.6.2 for details concerning the use of the BREAK instruction.
2. Use the EGP/EGF instruction to perform the pulse operation of an index-modified program between the FOR and NEXT instructions. Refer to 5.2.5 for details of the EGP/EGF instruction. The program samples are shown below:



3. Branching into a FOR to NEXT loop using a JMP or other branch instruction from the outside of the FOR to NEXT loop is not possible.

## 7.6.2 Forced end of FOR to NEXT instruction loop (BREAK(P))

Basic High performance Process Redundant Universal



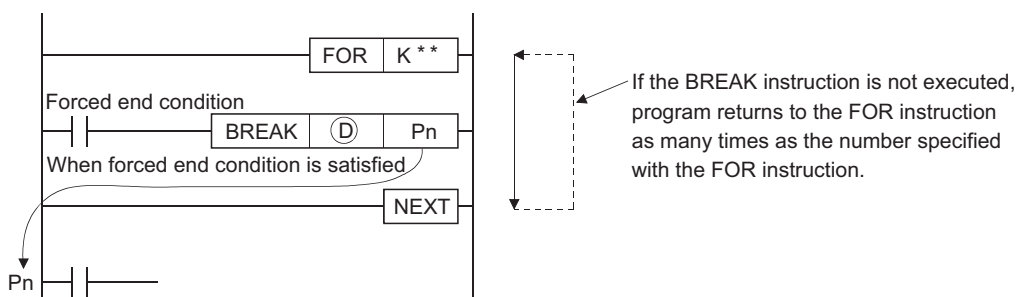
ⓐ : Number of the device where the remaining number of loops will be stored (BIN 16 bits)

Pn : Number of the pointer (device name (pointer)) where the program is branched at the forced end of a loop.

Setting Data	Internal Devices		R, ZR	JGO		UNGO	Zn	Constants	Other P
	Bit	Word		Bit	Word				
ⓐ					○			—	—
Pn					—			—	○

### ★ Function

- (1) Forces an end to a FOR to NEXT instruction loop and shifts the operation to the pointer specified by Pn. Only a pointer within the same program file can be assigned to Pn. If a pointer of the other program file is used, an operation error will be returned.



- (2) The remaining number of the FOR to NEXT instruction loop times is stored at ⓐ. Note that the remaining number includes the operation when the BREAK instruction is executed.
- (3) The BREAK instruction can be used only during the execution of a FOR to NEXT instruction loop.
- (4) The BREAK instruction can be used only when there is only one level of nesting. When an end is forced to the multiple nesting levels, execute the same number of BREAK instructions for the nesting levels.

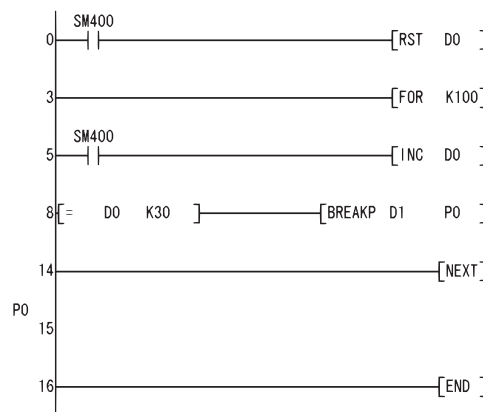
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The BREAK instruction is used in a case other than with the FOR to NEXT instruction loop. (Error code: 4203)
  - The jump destination for the pointer designated by Pn does not exist. (Error code: 4210)
  - The pointer of another program file is designated for Pn. (Error code: 4210)

## Program Example

- (1) The following program forces the FOR to NEXT loop to end when the value of D0 reaches 30 (when the FOR to NEXT loop has been executed 30 times).

[Ladder Mode]



[List Mode]

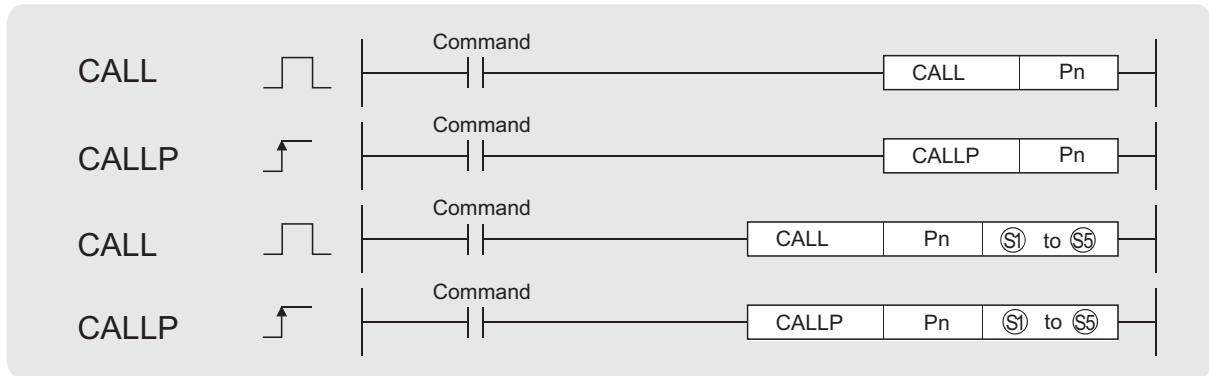
Step	Instruction	Device
0	LD	SM400
1	RST	D0
3	FOR	K100
5	LD	SM400
6	INC	D0
8	LD=	D0 K30
11	BREAKP	D1 PO
14	NEXT	
15	PO	
16	END	

### Remark

The value 71 is stored at D1 when the BREAK instruction is executed.

## 7.6.3 Subroutine program calls (CALL(P))

Basic High performance Process Redundant Universal



Pn : Head pointer number of a subroutine program (Device name)

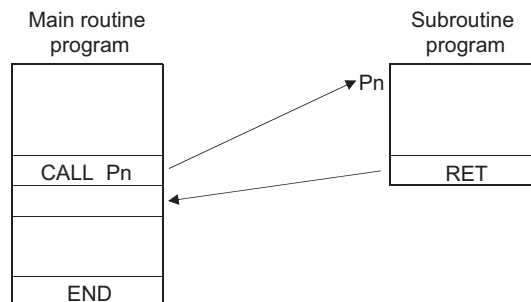
S1 to S5 : Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

Setting Data	Internal Devices		R, ZR	JGO		UJGO	Zn	Constants K, H	Other P
	Bit	Word		Bit	Word				
Pn	—	—				—			○
S1 to S5	○ (Other than F)	○				○			—

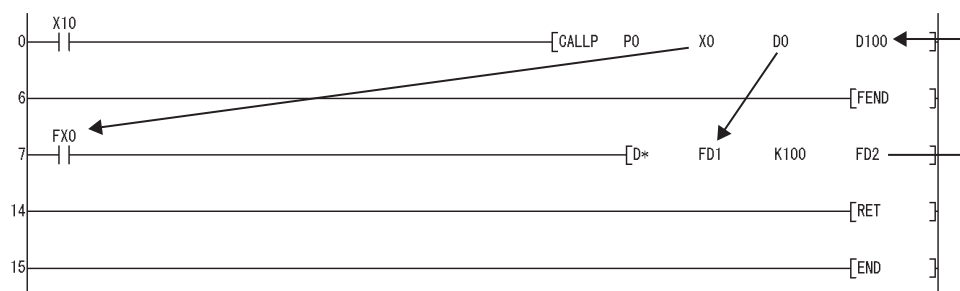
### ★ Function

- (1) When the CALL (P) instruction is executed, executes the subroutine program of the program specified by Pn.

[ The CALL (P) instruction can execute subroutine programs specified by a pointer within the same program file and subroutine programs specified by a common pointer. ]



- (2) When function devices (FX, FY, FD) are used by a subroutine program, specify a device with ① to ⑤ corresponding to the function device. The contents to the devices specified by ① to ⑤ are as indicated below.



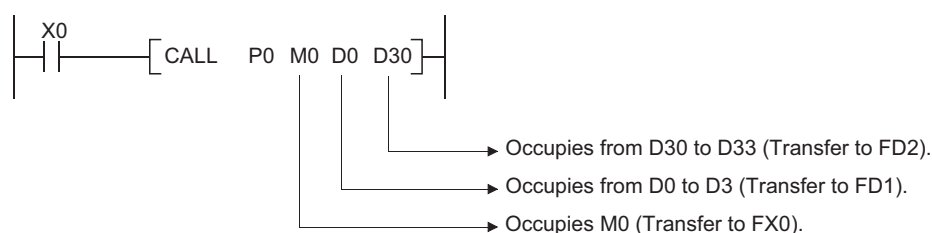
- (a) Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- (b) After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- (c) The processing units for the function devices are as follows:
- FX, FY: Bits
  - FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data Size	Remark
• FX • FY	Bit device	1 point	—
	When bit designation is made for word device	1 bit	
• FD	When digit designation of a bit device is used *1	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

\*1: An error will not occur even when the device number specified by ① to ⑤ is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]



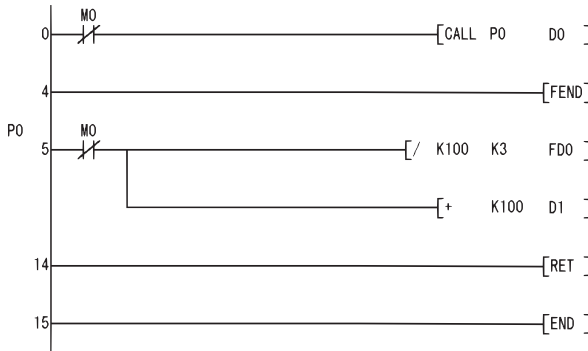
- (3) ① to ⑤ can be used with the CALL (P) instruction.
- (4) The number of function devices to be used by a subroutine program must be identical to the number of arguments in the CALL (P) instruction.  
Also, the types of the function device and CALL (P) argument used should be identical.
- (5) Device numbers specified by the CALL (P) instruction should not overlap.  
If they do overlap, it will not be possible to obtain accurate calculations.

- (6) The device used in the argument of the CALL (P) instruction should not be used in a subroutine program. If used, it will not be possible to obtain accurate calculations. (Refer to the following program example.)
- (7) When the device, either timer or counter, is used in the argument of the CALL(P) instruction, only the current value is transmitted/received.

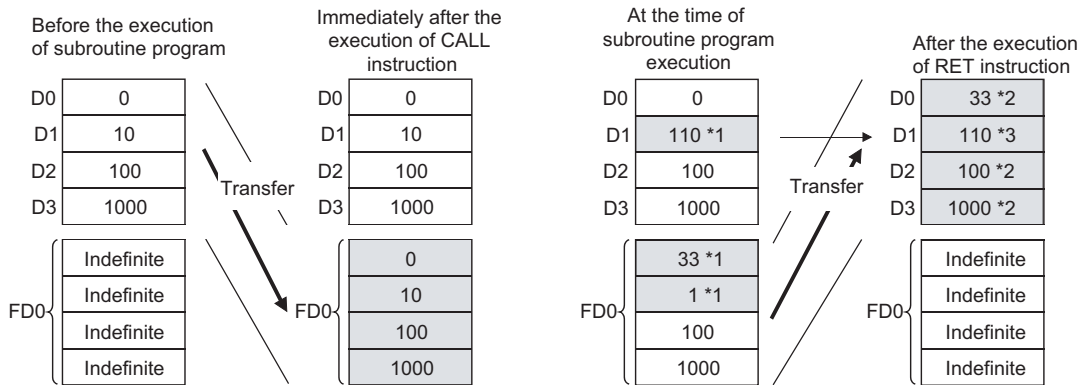
**Incorrect operation example**

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.

[Program example]



[Operation performed after subroutine program execution]



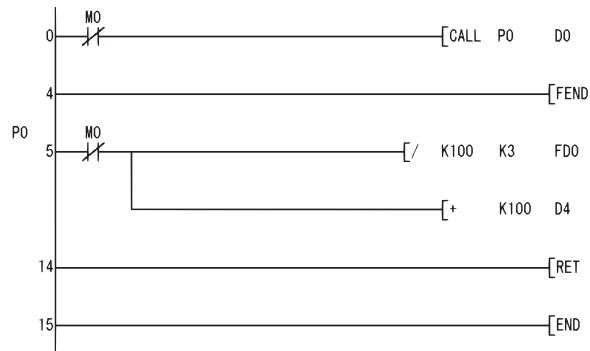
- \*1: Stores the execution result of the subroutine program.
- \*2: Replaced by the value of the function device.
- \*3: D1 does not reflect the value of the function device.



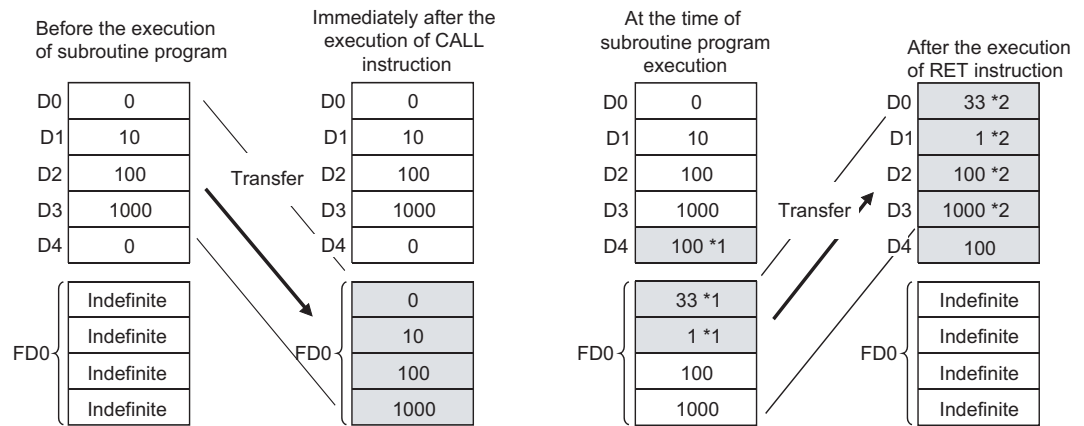
### Correct operation example

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.

[Program example]



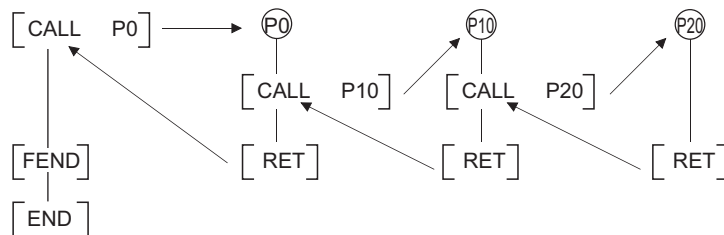
[Operation performed after subroutine program execution]



\*1: Stores the execution result of the subroutine program.

\*2: Replaced by the value of the function device.

- (8) Up to 16 nesting levels are possible with the CALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



- (9) Devices which are turned ON within subroutine programs will be latched even if the subroutine program is not executed. Devices which are turned ON during the execution of a subroutine program can be turned OFF by the execution of the FCALL(P) instruction.



## Operation Error

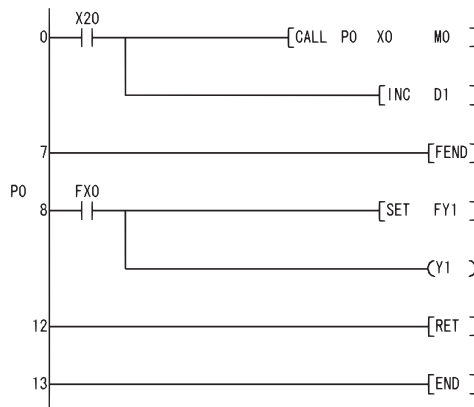
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified for the argument cannot be secured for the data size. (Error code: 4101)
  - Following the execution of the CALL (P) instruction, an END, FEND, GOEND, or STOP instruction is executed before the execution of the RET instruction. (Error code: 4211)
  - An RET instruction is executed prior to the execution of the CALL (P) instruction. (Error code: 4212)
  - A 17th nesting level is executed. (Error code: 4213)
  - There is no subroutine program for the pointer specified in the CALL (P) instruction. (Error code: 4210)



## Program Example

- (1) The following program executes a subroutine program with argument when X20 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	CALL	P0 X0 M0
5	INC	D1
7	FEND	
8	PO	
9	LD	FX0
10	SET	FY1
11	OUT	Y1
12	RET	
13	END	

## 7.6.4 Return from subroutine programs (RET)

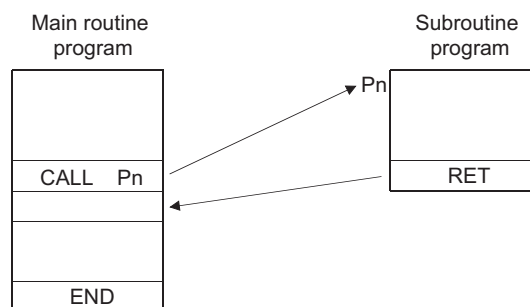
Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	J:G:O		U:G:O	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

- (1) Indicates end of subroutine program
- (2) When the RET instruction is executed, returns to the step following the CALL (P), FCALL (P), ECALL (P), EFCALL (P) or XCALL instruction which called the subroutine program.

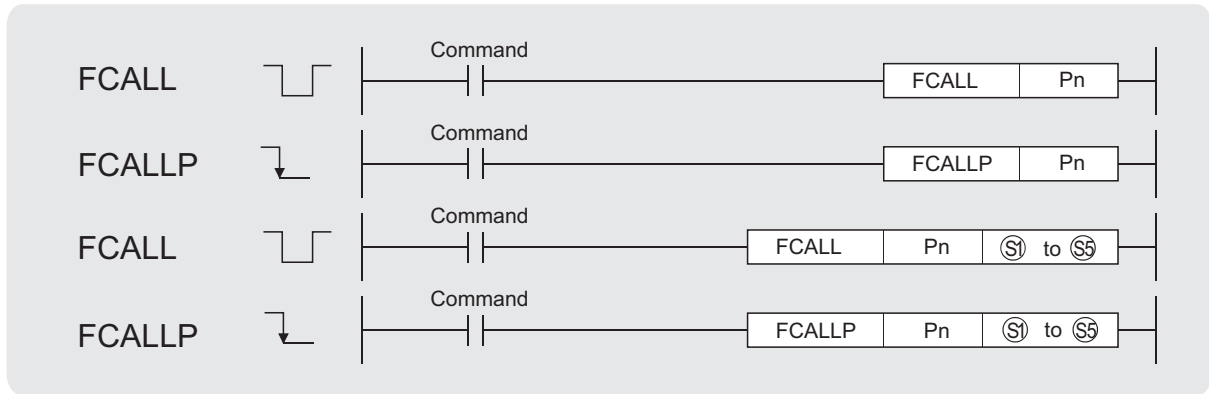


### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - Following the execution of the CALL(P), FCALL (P), ECALL (P), EFCALL (P) or XCALL instruction, an END, FEND, GOEND, or STOP instruction is executed before the execution of the RET instruction. (Error code: 4211)
  - An RET instruction is executed prior to the execution of the CALL (P), FCALL (P), ECALL (P), EFCALL (P) or XCALL instruction. (Error code: 4212)

## 7.6.5 Subroutine program output OFF calls (FCALL(P))

Basic High performance Process Redundant Universal



Pn : Head pointer number of a subroutine program (Device name)

S1 to S5 : Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

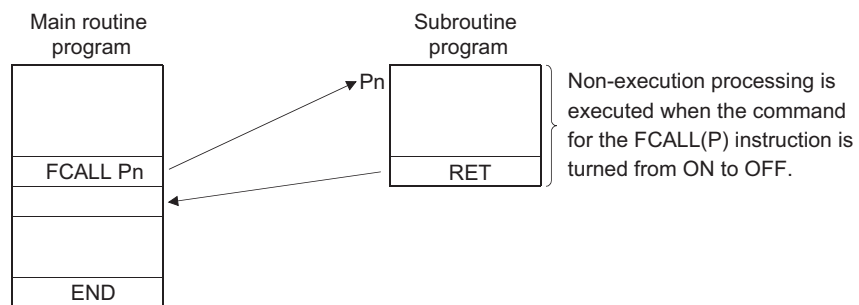
Setting Data	Internal Devices		R, ZR	JOG		UJOG	Zn	Constants	Other P
	Bit	Word		Bit	Word				
Pn	—	—				—			○
S1 to S5	○ (Other than F)	○				○			—

### ★ Function

- (1) When FCALL(P) is executed, the non-execution processing of the subroutine program of the pointer designated by Pn is performed.

[ The FCALL (P) instruction can execute subroutine programs designated by a pointer within the same program file, and subroutine programs designated by common pointers. ]

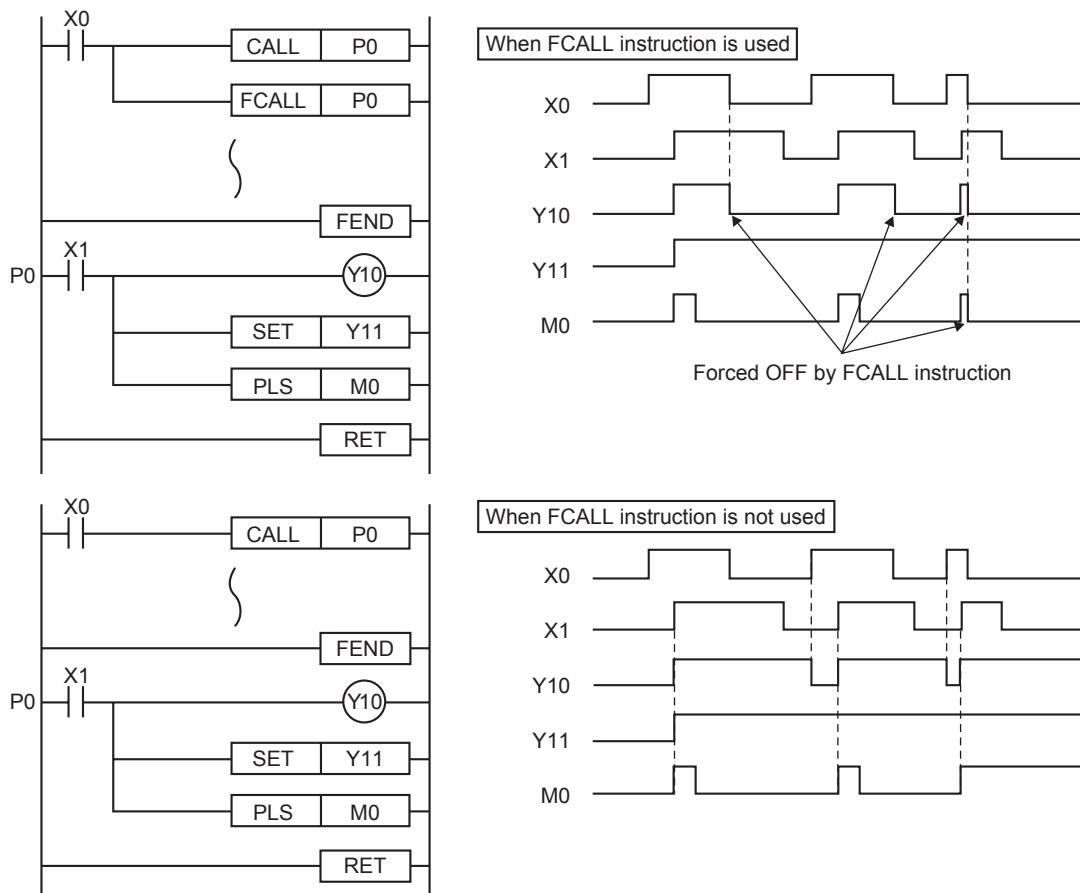
- (a) Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.



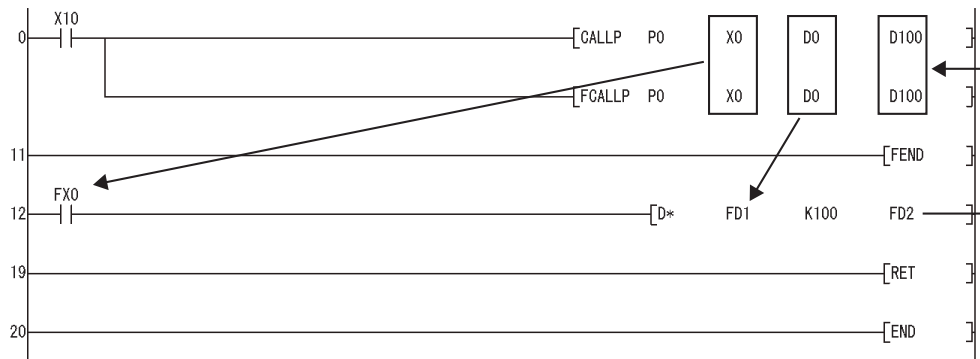
- (b) The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

OUT instruction	.....	Forced OFF
SET instruction	}	..... Maintains status
RST instruction		
SFT instruction		
Basic instructions		
Application instructions		
PLS instruction	}	Processing identical to
Pulse generation instruction (□ P)		
Present value of low speed/high speed timers	.....	0
Present value of retentive timer	}	..... Preserves
Present value of counter		

- (2) The FCALL (P) instruction is used in conjunction with the CALL(P) instruction.
- (3) If the FCALL (P) instruction is used in conjunction with the CALL(P) instruction, non-execution processing of a subroutine program is performed when the execution command is turned OFF, enabling forcible turning OFF of the OUT instruction and the PLS instruction (including □ P instructions).  
In case the FCALL (P) instruction is not used in conjunction with the CALL(P) instruction, non-execution processing of a subroutine program is not performed even if the execution command is turned OFF. Therefore, output status of the individual coil instructions remains unchanged.



(4) When function devices (FX, FY, FD) are used by a subroutine program, specify a device with ⑤1 to ⑤⑤ corresponding to the function device. The contents to the devices specified by ⑤1 to ⑤⑤ are as indicated below.



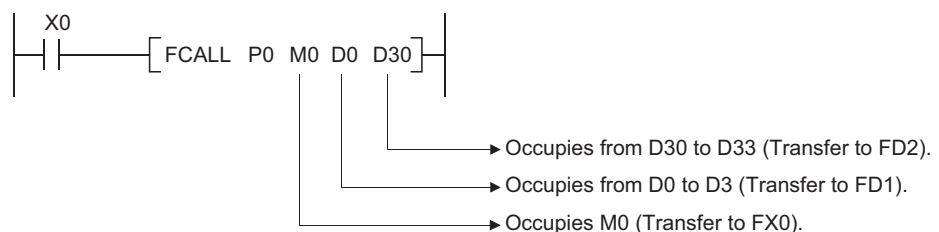
- (a) Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- (b) After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- (c) The processing units for the function devices are as follows:
  - FX, FY: Bits
  - FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data Size	Remark
• FX • FY	Bit device	1 point	—
	When Bit Designation has been Made for Word Device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The upper 2 words of FD become 0
	Word device	4 words	—

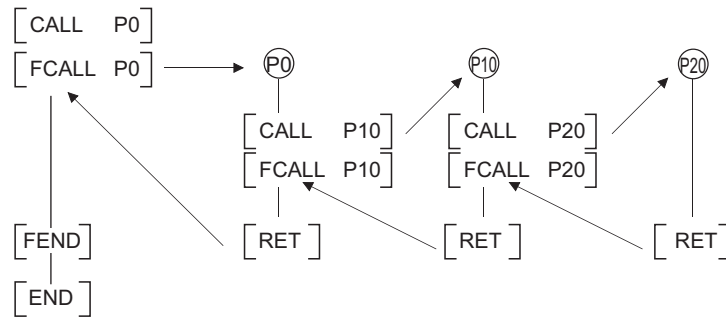
\*1: An error will not occur if the device number specified by ⑤1 to ⑤⑤ is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]



(5) The FCALL (P) instruction can use from ⑤1 to ⑤⑤ .

- (6) Up to 16 nesting levels are possible with the FCALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



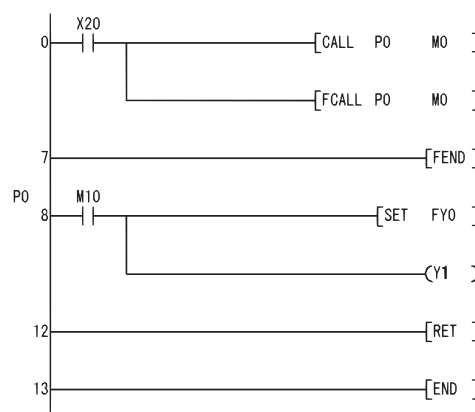
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified for the argument cannot be secured for the data size. (Error code: 4101)
  - Following the execution of the CALL (P) instruction, an END, FEND, GOEND, or STOP instruction is executed before the execution of the RET instruction. (Error code: 4211)
  - An RET instruction is executed prior to the execution of the FCALL (P) instruction. (Error code: 4212)
  - A 17th nesting level is executed. (Error code: 4213)
  - The subroutine program of the pointer designated by the FCALL (P) instruction does not exist. (Error code: 4210)

## Program Example

- (1) The following program executes a subroutine program with argument when X20 is turned ON, and forces non-execution processing when X20 is turned from ON to OFF.

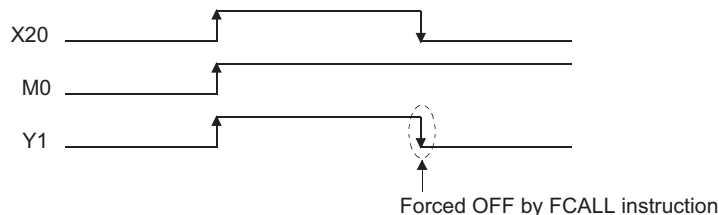
[Ladder Mode]



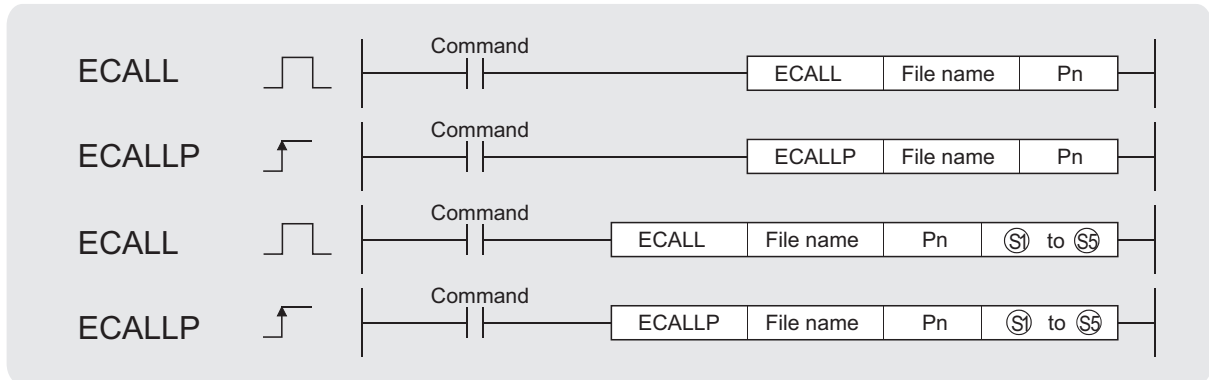
[List Mode]

Step	Instruction	Device
0	LD	X20
1	CALL	P0 M0
4	FCALL	P0 M0
7	FEND	
8	PO	
9	LD	M10
10	SET	FY0
11	OUT	Y1
12	RET	
13	END	

[Operation]



## 7.6.6 Subroutine calls between program files (ECALL(P))



File name : Name of the program file to be called (character string)

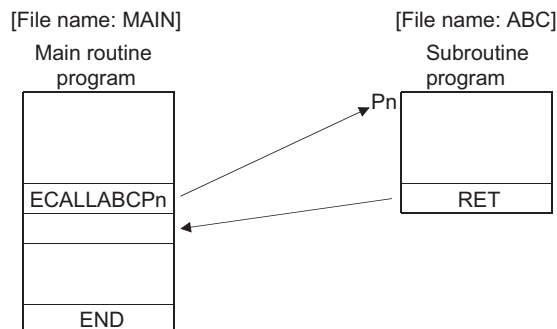
Pn : Head pointer number of a subroutine program (Device name)

S1 to S5 : Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants		Other P
	Bit	Word		Bit	Word			K, H	\$	
File name	—	○				—		○	—	
Pn	—	—				—		—	○	
S1 to S5	○ (Other than F)	○				○		—	—	

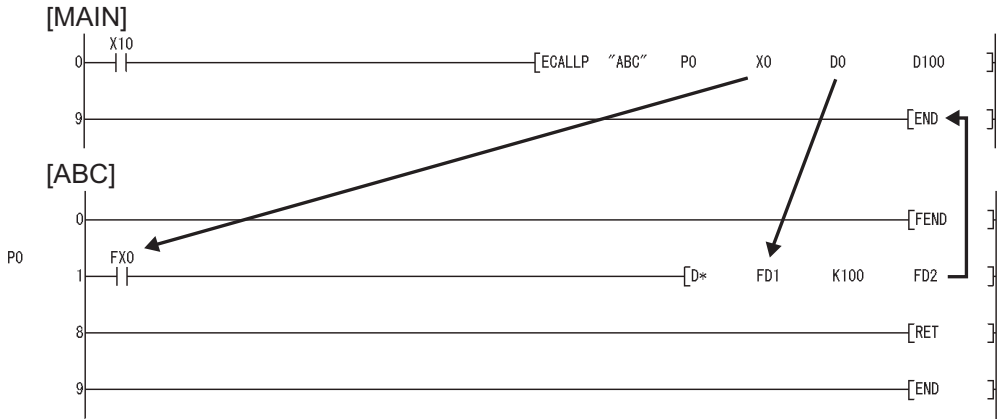
### ★ Function

- (1) Executes the subroutine program of the pointer designated by Pn in the designated program file name when the ECALL (P) instruction is executed. The ECALL(P) instruction can be used to call a subroutine program that uses a local pointer from a different program file.





- (2) Only the file name of a program file stored in the drive 0 (program memory/internal RAM) can be designated for a file name.
- (3) It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)
- (4) When function devices (FX, FY, FD) are used by a subroutine program, specify a device corresponding to the function device with (S1) to (S5). The contents of the devices specified by (S1) to (S5) are as indicated below.



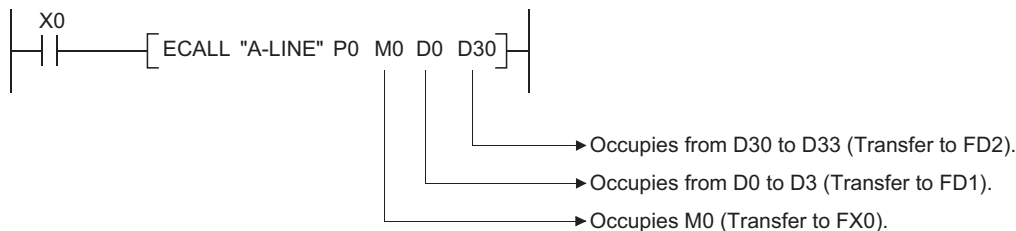
- (a) Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- (b) After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- (c) The processing units for the function devices are as follows:
  - FX, FY: Bits
  - FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data Size	Remark
• FX • FY	Bit device	1 point	—
	When Bit Designation has been Made for Word Device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

\*1: An error will not occur even when the device number specified by (S1) to (S5) is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]

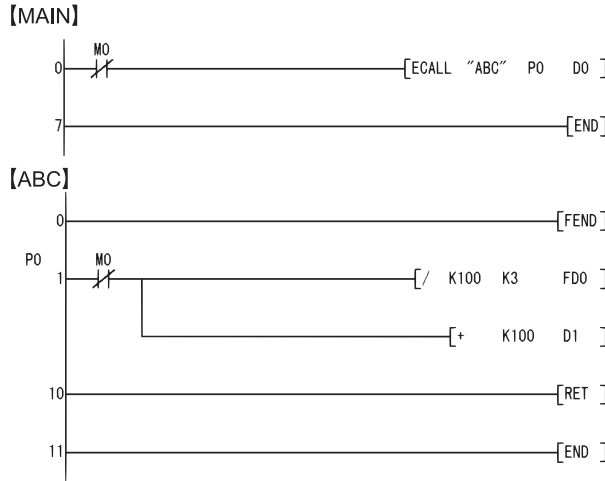


- (5) From S1 to S9 can be used by the ECALL instruction.
- (6) The device used in the argument of the ECALL instruction should not be used in a subroutine program.  
If used, it will not be possible to obtain accurate calculations. (Refer to the following program example.)

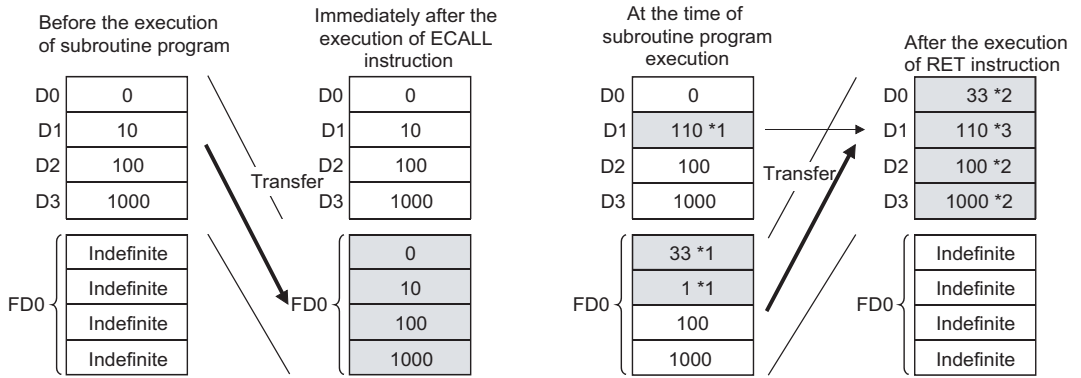
**Incorrect operation example**

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D1 is used in the subroutine program.

[Program example]



[Operation performed after subroutine program execution]

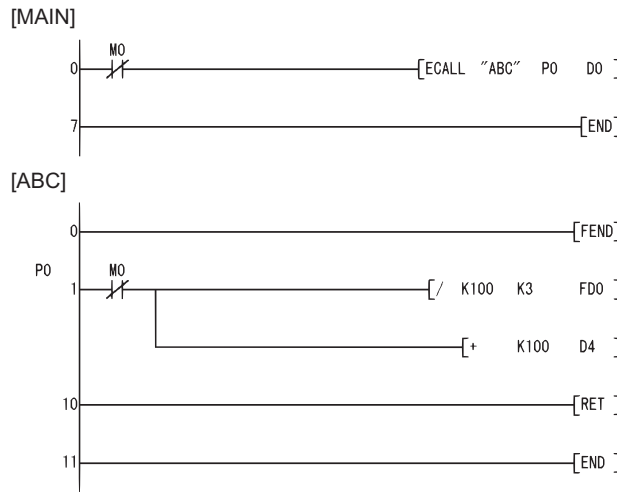


- \*1: Stores the execution result of the subroutine program.
- \*2: Replaced by the value of the function device.
- \*3: D1 does not reflect the value of the function device.

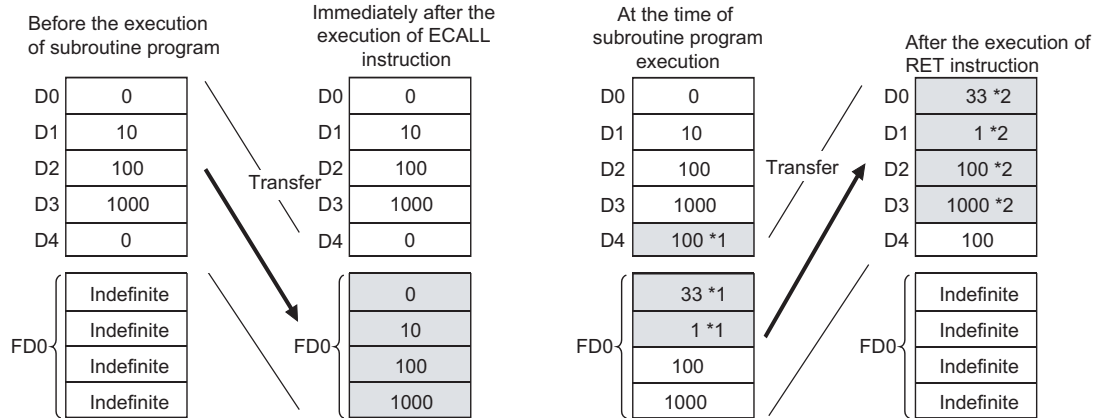
**Correct operation example**

The following example shows the operation performed when D0 is specified for FD0 in the subroutine program and D4 is used in the subroutine program.

[Program example]

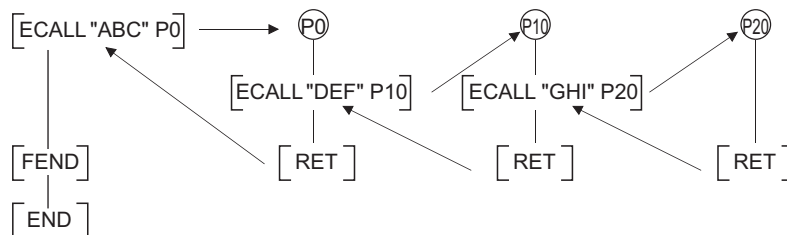


[Operation performed after subroutine program execution]



\*1: Stores the execution result of the subroutine program.  
 \*2: Replaced by the value of the function device.

- (7) The numbers of the devices designated by the arguments in the ECALL(P) instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.
- (8) Up to 16 levels of nesting can be used with the ECALL(P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



- (9) Devices which are turned ON within subroutine programs will be latched even if the subroutine program is not executed. Devices turned ON during the execution of a subroutine program can be turned OFF by the EFCALL(P) instruction.

## Operation Error

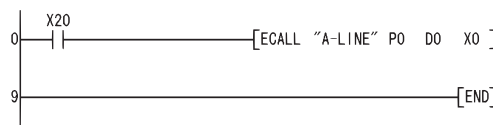
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified for the argument cannot be secured for the data size. (Error code: 4101)
  - Following the execution of the ECALL (P) instruction, an END, FEND, GOEND, or STOP instruction is executed before the execution of the RET instruction. (Error code: 4211)
  - An RET instruction is executed prior to the execution of the ECALL(P) instruction. (Error code: 4212)
  - A 17th nesting level is executed. (Error code: 4213)
  - The subroutine program of the pointer designated by the ECALL(P) instruction does not exist. (Error code: 4210)
  - The designated file does not exist. (Error code: 2410)
  - The designated file cannot be executed. (Error code: 2411)

## Program Example

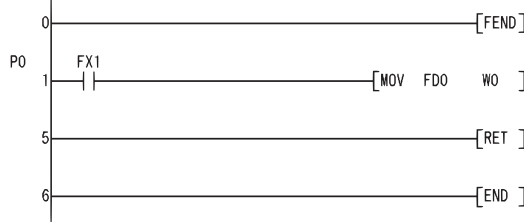
- (1) The following program executes program block P0 of the program A-LINE when X20 is turned ON.

[Ladder Mode]

[MAIN]



[A-LINE]

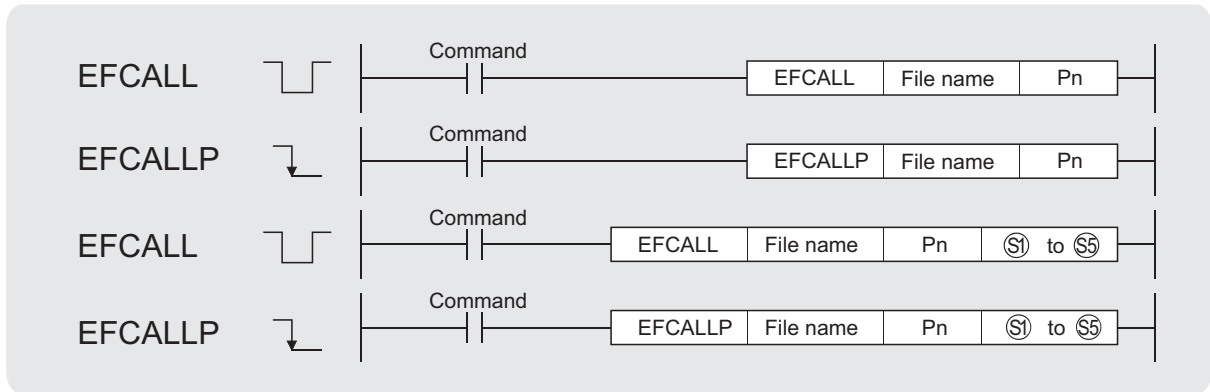


[List Mode]

Step	Instruction	Device
0	LD	X20
1	ECALL	"A-LINE" P0 D0 XO
9	END	

Step	Instruction	Device
0	FEND	
1	P0	
2	LD	FX1
3	MOV	FDO W0
5	RET	
6	END	

## 7.6.7 Subroutine output OFF calls between program files (EFCALL(P))



File name : Name of the program file to be called (character string)  
 Pn : Head pointer number of a subroutine program (Device name)  
 (S1) to (S5) : Number of the device to be passed as an argument to a subroutine program  
 (bits, BIN 16 bits, BIN 32 bits)

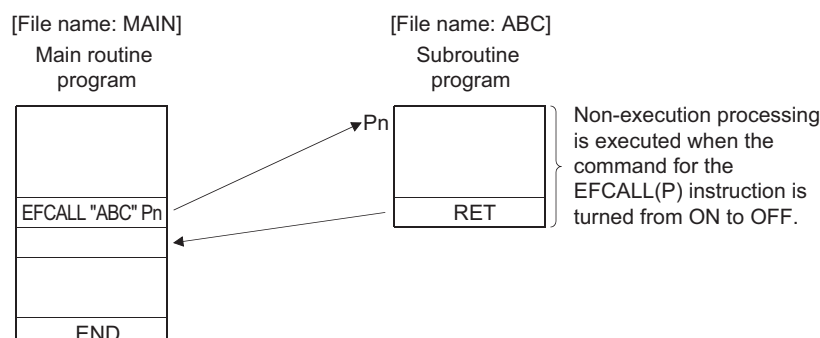
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants		Other P
	Bit	Word		Bit	Word			K, H	\$	
File name	—		○			—			○	—
Pn	—		—			—			—	○
(S1) to (S5)	○ (Other than F)		○			○			—	—

### ★ Function

- When the EFCALL(P) instruction is executed, the non-execution processing of the subroutine program of the pointer designated by Pn is performed.

[ The EFCALL (P) can also be used to call a subroutine program that uses a local pointer from a different program file. ]

- Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.



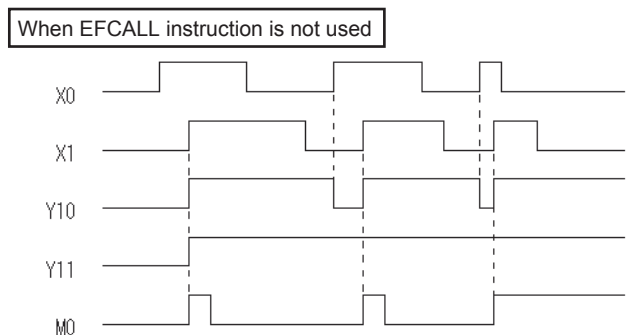
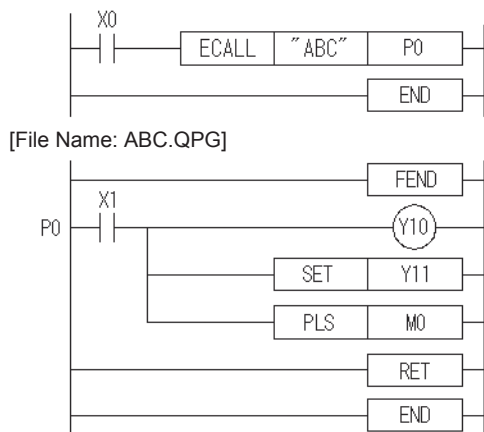
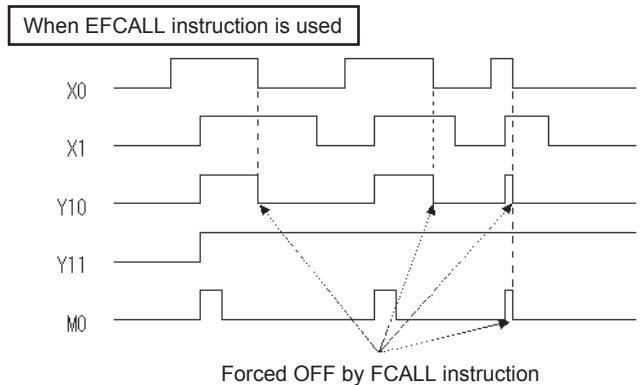
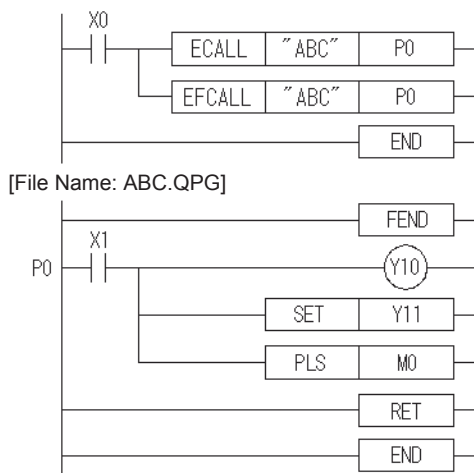
(b) The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

OUT instruction	.....	Forced OFF
SET instruction	}	..... Maintains status
RST instruction		
SFT instruction		
Basic instructions		
Application instructions		
PLS instruction	}	Processing identical to when condition contacts are OFF
Pulse generation instruction (□P)		
Present value of low speed/high speed timers	.....	0
Present value of retentive timer	}	..... Preserves
Present value of counter		

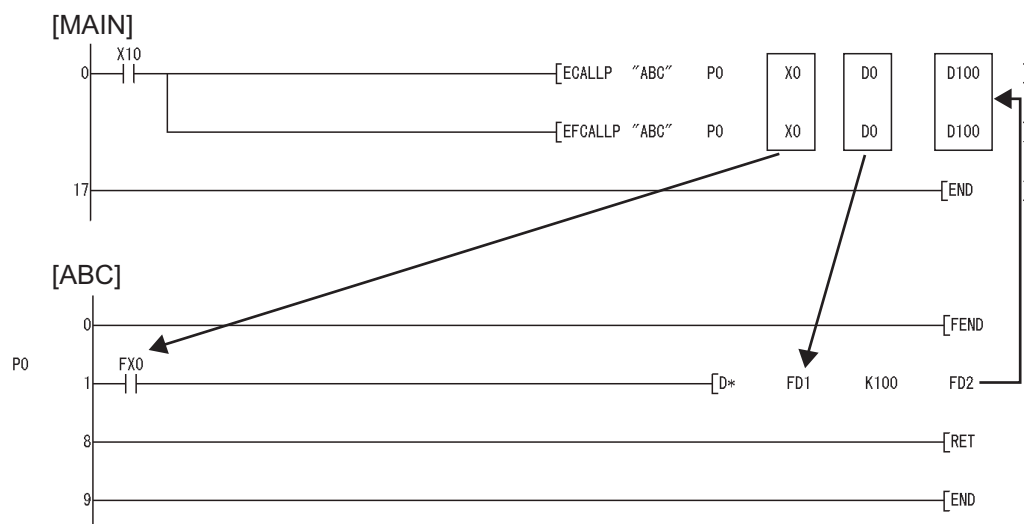
(2) The EFCALL (P) instruction is used in combination with the ECALL (P) instruction.

(3) If the EFCALL(P) instruction is used in conjunction with the ECALL(P) instruction, non-execution processing of a subroutine program is performed when the execution command is turned OFF, enabling forcible turning OFF of the OUT instruction and the PLS instruction (including □P instructions).

In case the EFCALL(P) instruction is not used in conjunction with the ECALL(P) instruction, non-execution processing of a subroutine program is not performed even if the execution command is turned OFF. Therefore, output status of the individual coil instructions remains unchanged.



- (4) Only the file name of a program file stored in the drive 0 (program memory/internal RAM) can be designated for a file name.
- (5) It is not necessary to designate the extension (.QPG) with the file name. (Only .QPG files will be acted on.)
- (6) When function devices (FX, FY, FD) are used by a subroutine program, specify a device corresponding to the function device with ⑤1 to ⑤5.



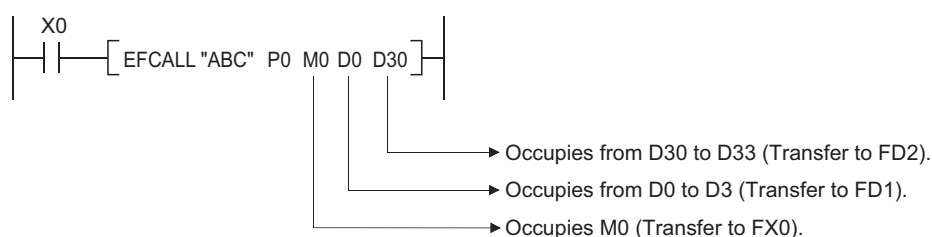
- (a) Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- (b) After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- (c) The processing units for the function devices are as follows:
  - FX, FY: Bits
  - FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

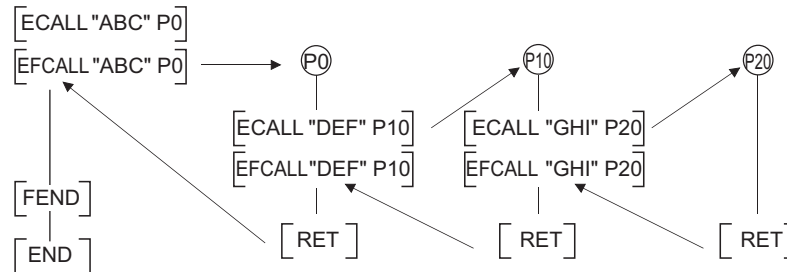
Function devices	Device	Data Size	Remark
• FX • FY	Bit device	1 point	—
	When Bit Designation has been Made for Word Device	1 bit	
• FD	When digit designation of a bit device is used*1	4 words	The upper 2 words of FD become 0
	Word device	4 words	—

\*1: An error will not occur even when the device number specified by ⑤1 to ⑤5 is not a multiple of 16 at the digit designation of the bit device.

[Main routine program]



- (7) ⑤① to ⑤⑤ can be used with the EFCALL (P) instruction.
- (8) The number of function devices used by subroutine programs must be identical to the number of arguments used by the EFCALL (P) instruction. Further, the function devices should be identical to the types of arguments used by the EFCALL (P) instruction.
- (9) Up to 16 levels of nesting can be used with the EFCALL (P) instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.



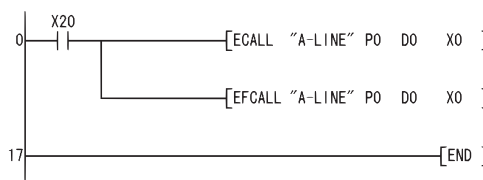
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified for the argument cannot be secured for the data size. (Error code: 4101)
  - Following the execution of the EFCALL (P) instruction, an END, FEND, GOEND, or STOP instruction is executed before the execution of the RET instruction. (Error code: 4211)
  - An RET instruction is executed prior to the execution of the EFCALL (P) instruction. (Error code: 4212)
  - A 17th nesting level is executed. (Error code: 4213)
  - The subroutine program of the pointer designated by the EFCALL (P) instruction does not exist. (Error code: 4210)
  - The designated file does not exist. (Error code: 4210)
  - The designated file cannot be executed. (Error code: 2411)

## Program Example

- (1) The following program executes a subroutine program with argument when X0 is ON, and forces non-execution processing when X20 is turned ON to OFF.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	ECALL	"A-LINE" P0 D0 X0
9	EFCALL	"A-LINE" P0 D0 X0
17	END	



## 7.6.8 Subroutine program call (XCALL)



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Pn : Head pointer number of a subroutine program (Device name)

S1 to S5 : Number of the device to be passed as an argument to a subroutine program (bits, BIN 16 bits, BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other P
	Bit	Word		Bit	Word				
P	—	—				—			○
S1 to S5	○ (Other than F)	○				○			—

### ★ Function

(1) XCALL instruction executes the subroutine program and performs non-execution processing of the subroutine program.

(a) Execution of subroutine program

Executes each coil instruction according to ON/OFF status of the condition contacts.

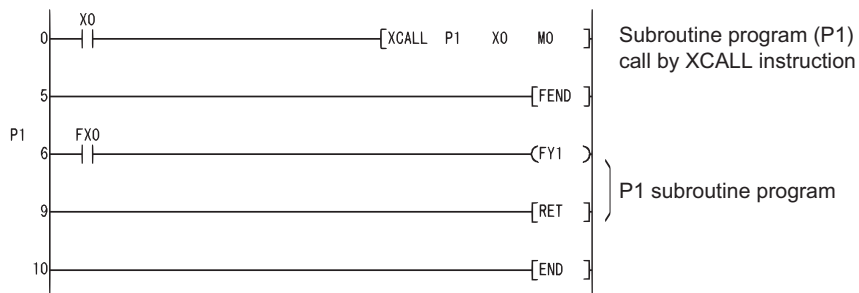
(b) Non-execution of subroutine program

Performs the same processing for each coil instruction as when the condition contacts are OFF status. The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

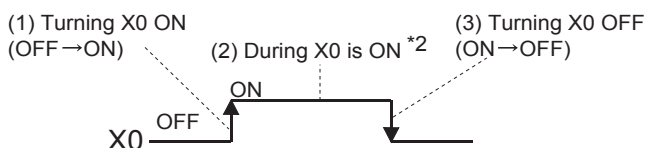
OUT instruction	.....	Forced OFF
SET instruction	}	..... Maintains status
RST instruction		
SFT instruction		
Basic instructions		
Application instructions	}	..... Processing identical to when condition contacts are OFF
PLS instruction		
Pulse generation instruction (□ P)		
Present value of low speed/high speed timers	.....	0
Present value of retentive timer	}	..... Preserves
Present value of counter		

(2) Operation of XCALL instruction varies according to the CPU module type. The following program example shows the operation of XCALL instruction for each CPU module.

[Program example]



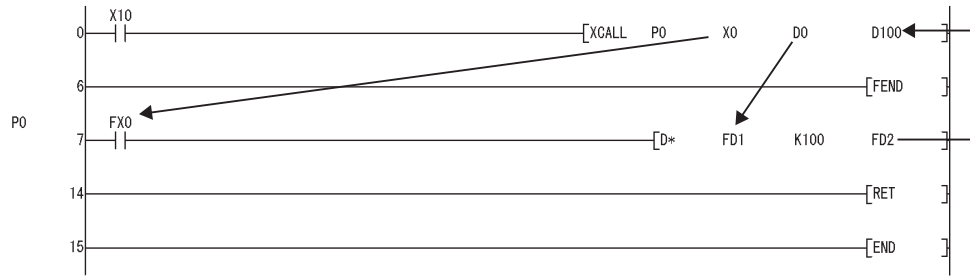
[ON/OFF timing of X0]



\*2: Time during X0 is ON(2)) does not include the time when turning X0 ON (1)).

Component	Operation of XCALL instruction
<ul style="list-style-type: none"> <li>• Process CPU (serial No. of first 5 digits : 07031 or earlier)</li> <li>• High performance model QCPU (serial No. of first 5 digits: 06081 or earlier.)</li> </ul>	<ol style="list-style-type: none"> <li>1) When X0 is turned ON: Without process (Do not execute subroutine program of "P1".)</li> <li>2) During X0 is ON: Execute subroutine program of "P1".</li> <li>3) When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".</li> </ol>
<ul style="list-style-type: none"> <li>• High performance model QCPU (serial No. of first 5 digits: 06082 or later.)</li> <li>• Process CPU (serial No. of first 5 digits : 07032 or later)</li> </ul>	<ol style="list-style-type: none"> <li>1) Using SM734 (XCALL instruction executing condition designation) to select operation when X0 is turned ON.                             <ul style="list-style-type: none"> <li>• When SM734 is OFF: Without process (Do not execute subroutine program of "P1".)</li> <li>• When SM734 is ON: Execute subroutine program of "P1".</li> </ul> </li> <li>2) During X0 is ON: Execute subroutine program of "P1".</li> <li>3) When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".</li> </ol>
<ul style="list-style-type: none"> <li>• Redundant CPU</li> <li>• Basic model QCPU</li> <li>• Universal model QCPU</li> </ul>	<ol style="list-style-type: none"> <li>1) When X0 is turned ON: Execute subroutine program of "P1".</li> <li>2) During X0 is ON: Execute subroutine program of "P1".</li> <li>3) When X0 is turned OFF: Perform "Non-execution processing" of subroutine program of "P1".</li> </ol>

- (3) When function devices (FX, FY, FD) are used by a subroutine program, specify a device with ④① to ④⑤ corresponding to the function device. The contents to the devices specified by ④① to ④⑤ are as indicated below.



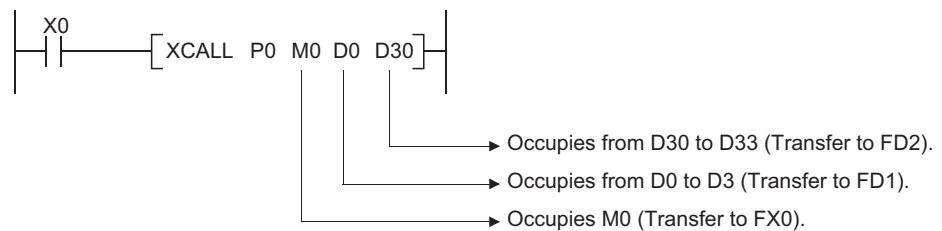
- (a) Prior to execution of the subroutine program, bit data is transmitted to FX, and word data is transmitted to FD.
- (b) After the execution of the subroutine program, the contents of FY and FD are transmitted to the corresponding devices.
- (c) The processing units for the function devices are as follows:
  - FX, FY: Bits
  - FD : 4-word units

The size of the data to be dealt with will differ depending on the device specified in the argument. The device specified as a function device should be secured for the data size. An error will occur if it cannot be secured for the data size.

Function devices	Device	Data Size	Remark
• FX • FY	Bit device	1 point	—
	When Bit Designation has been Made for Word Device	1 bit	
• FD	When digit designation of a bit device is used*3	4 words	The data size varies depending on the instruction to be used.
	Word device	4 words	

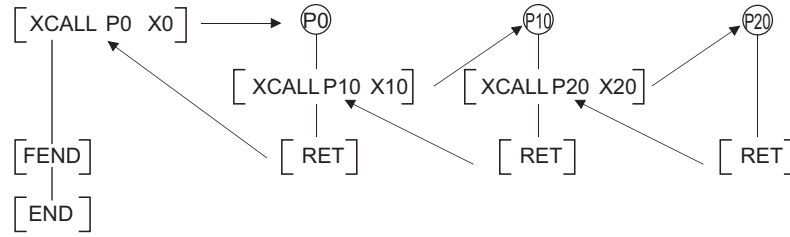
\*3: An error will not occur even when the device number specified by ④① to ④⑤ is not a multiple of 16 at the digit specification of the bit device.

[Main routine program]



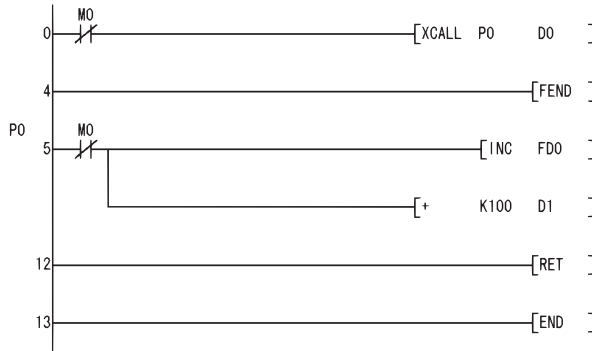
- (4) ④① to ④⑤ can be used by the XCALL instruction.
- (5) The number of function devices used by a subroutine program must be identical to the number of arguments in the XCALL instruction. Also, the function device and the type of XCALL argument should be identical.
- (6) Device numbers specified in the argument of the XCALL instruction should not overlap. If they do overlap, it will not be possible to obtain accurate calculations.

- (7) Up to 16 nesting levels can be used with the XCALL instruction. However, this 16 levels is the total number of levels in the CALL(P), FCALL(P), ECALL(P), EFCALL(P), and XCALL instructions.

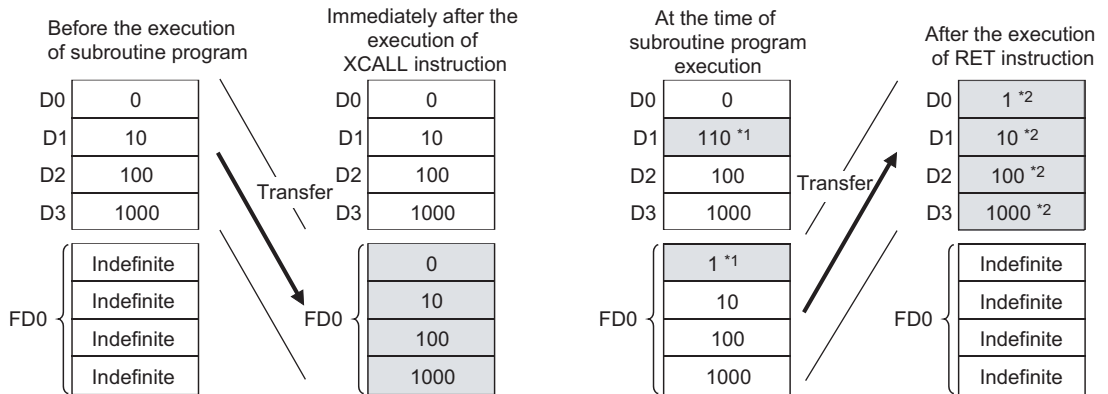


- (8) The device used for the argument of the XCALL instruction must not be used in a subroutine program. If used, it will not be possible to perform correct calculations. (Refer to the following program example.)  
The processing to be executed when D1 is used in a subroutine program with D0 designated for FD0 in a subroutine program is shown below.

[Program example]



[Operation performed after subroutine program execution]



\*1: Stores the execution result of the subroutine program.

\*2: Replaced by the value of the function device. D1 does not reflect the operation result in the subroutine program.

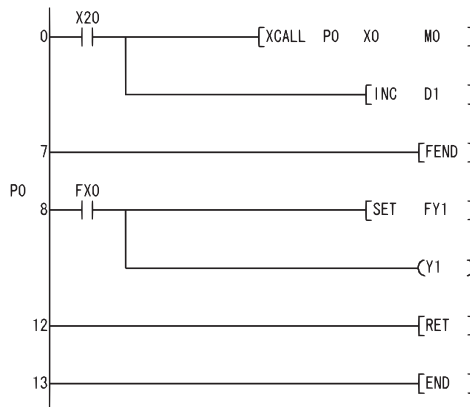
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified for the argument cannot be secured for the data size. (Error code: 4101)
  - Following the execution of the XCALL(P) instruction, the END, FEND, GOEND or STOP instruction is executed before the execution of the RET instruction. (Error code: 4211)
  - The RET instruction is executed prior to the execution of the XCALL(P) instruction. (Error code: 4212)
  - A 17th nesting level is executed. (Error code: 4213)
  - There is no subroutine program for the pointer specified in the XCALL(P) instruction. (Error code: 4210)

## Program Example

- (1) The following program executes a subroutine program with argument when X20 is turned ON.

[Ladder Mode]



[List Mode]

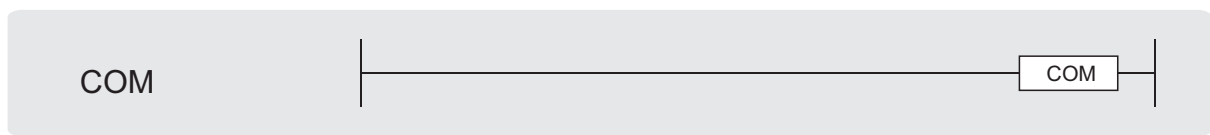
Step	Instruction	Device
0	LD	X20
1	XCALL	P0 X0 M0
5	INC	D1
7	FEND	
8	P0	
9	LD	FX0
10	SET	FY1
11	OUT	Y1
12	RET	
13	END	

## 7.6.9 Refresh instruction (COM)



Refer to Section 7.6.10 for the COM instruction of the following CPU modules.

- Basic model QCPU of serial No. 04122 or later
- High Performance model QCPU of serial No. 04012 or later
- Process CPU of serial No. 07032 or later
- Redundant CPU
- Universal model QCPU



Setting Data	Internal Devices		R, ZR	J:Q		U:G:Q	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

- (1) Use the COM instruction when:
  - (a) It is desired to increase the speed of transmission/reception processing to/from the remote I/O stations.
  - (b) It is desired to ensure reliable data transmission/reception with other stations that use different scan times during the execution of the data link.
- (2) The processing of the COM instruction differs depending on whether the special relay SM775 is ON or OFF.
  - When SM775 is OFF: Performs auto refresh and communication with a peripheral device \*1 \*2
  - When SM775 is ON: Performs communication with peripheral device only \*1

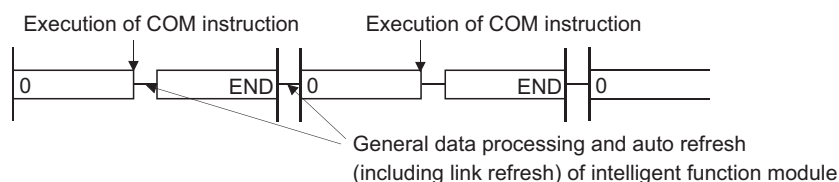
\*1: The following processing is performed in communication with peripheral device:

  - Monitor processing of other stations
  - Read processing by the serial communications module of the buffer memory of another intelligent function module

\*2: The auto refresh includes the following processing:

  - Refresh of MELSECNET/10H
  - CC-Link refresh
  - Auto refresh of intelligent function modules.

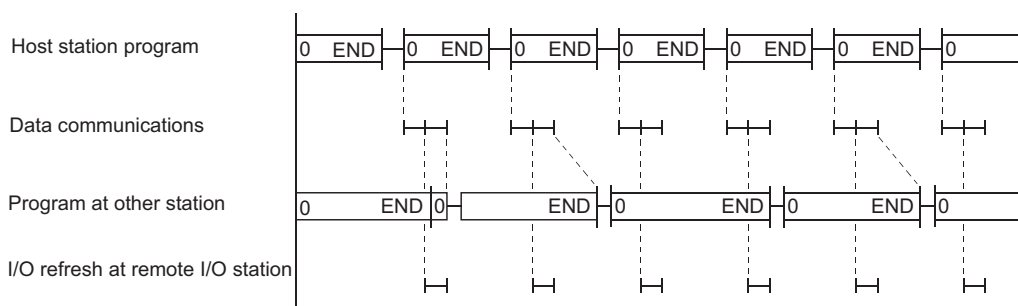
- (3) At the point of the execution of the COM instruction, the CPU module temporarily stops the processing of the sequence program, and performs the same operation as ordinary data processing as well as auto refresh of intelligent function modules (including link refreshes) at the END processing. However, the low speed cyclic refresh of MELSECNET/10 or MELSECNET/H is not performed.



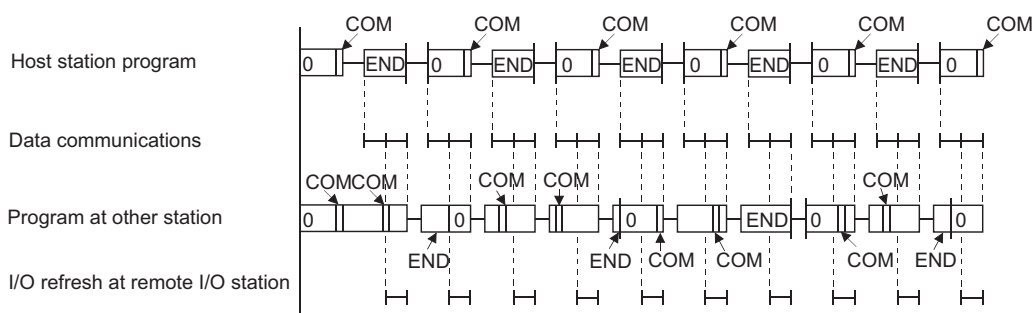
- (4) The COM instruction can be used in a sequence program any number of times. However, note that the scan time of the sequence program will be lengthened by the time taken for communication with peripheral device and the auto refresh (including the link refresh) of the intelligent function modules.

- (5) Data communications using the COM instruction

- (a) Example of data communications when COM instruction is not used



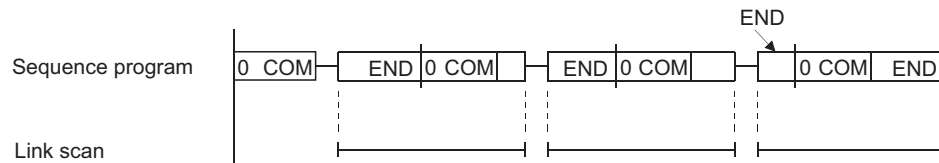
- (b) Example of data communications when COM instruction has been used



- When the COM instruction is used at the host station, it is possible to increase the number of data communication repetitions with the remote I/O station unconditionally, as shown in (b) above, and thus to speed up data communications.
- In cases where the remote station scan time is longer than the scan time of the host station, the COM instruction used at the remote station side can avoid the occurrence of timing failure in which the data cannot be fetched, as shown in (a).
- When the COM instruction has been used at the other station, a link refresh will be performed each time that station receives a command from the host station.

- Step 0                    ~COM instruction
  - COM instruction ~COM instruction
  - COM instruction ~END instruction
- } Link refresh can be performed once in each of these intervals.

- (6) If the scan time from the linked station is longer than the sequence program scan time at the host station, designating the COM instruction at the host station will not increase the speed of data communications.



---

**☒ POINT**

The programs in which the COM instruction cannot be used are shown below:

- Low-speed execution type programs
  - Interrupt programs
  - Fixed scan execution type programs
- 



## Operation Error

- (1) There are no operation errors associated with the COM instruction.



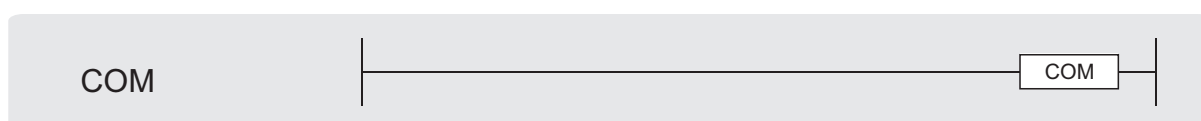
## 7.6.10 Select Refresh Instruction (COM)

Refer to Section 7.6.9. for the COM instruction of the following CPU modules.

- Basic model QCPU of serial No. 04121 or lower
- High Performance model QCPU of serial No. 04011 or lower
- Process CPU of serial No. 07031 or lower



The first 5 digits of the serial No. are "04122" or higher.  
 The first 5 digits of the serial No. are "04012" or higher.  
 The first 5 digits of the serial No. are "07032" or higher.



Setting Data	Internal Devices		Z, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

- When the COM instruction is executed, the following refresh operations can be performed.
  - I/O refresh
  - CC-Link refresh
  - CC-Link IE controller network refresh
  - MELSECNET/H refresh
  - Auto refresh of intelligent function modules
  - Auto refresh using QCPU standard area of multiple CPU system
  - Reading input/output data of all modules other than the multiple CPU system group
  - Auto refresh using the multiple CPU high speed transmission area of multiple CPU system
  - Communication with peripheral device

#### Remark

The following processing is performed in communication with peripheral device.

- Monitor processing of other station
- Read of another intelligent function module buffer memory by the serial communication module

- Turning OFF SM775 refreshes all refresh items except I/O refresh.

## (3) When selecting refresh items

- (a) Select refresh items by SD778, and set SM775 to ON.

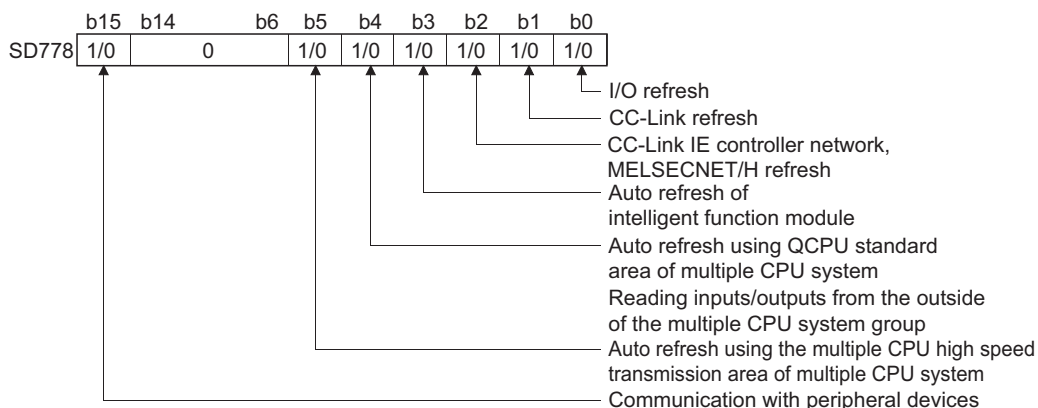
The following table shows the refresh items that can be designated by turning SM775 ON/OFF and with SD778.

Refresh Item	When SM775 is OFF	When SM775 is ON
I/O refresh	Executed	Whether to be executed or not can be selected.
CC-Link refresh		
CC-Link IE controller network refresh		
MELSECNET/H refresh		
Auto refresh of intelligent function modules		
Auto refresh using QCPU standard area of multiple CPU system		
Reading input/output data of all modules other than the multiple CPU system group		
Auto refresh using the multiple CPU high speed transmission area of multiple CPU system		
Communication with peripheral device		

- (b) Select refresh items using b0 to b5 and b15 of SD778.

Whether to execute each bit of SD778 or not can be designated as shown below:

Bit of SD778	Executed	Not Executed
b0 to b5	1	0
b15	0	1

**Example**

To make only the send/receive processing with the remote I/O station faster, designate MELSECNET/H refresh only.

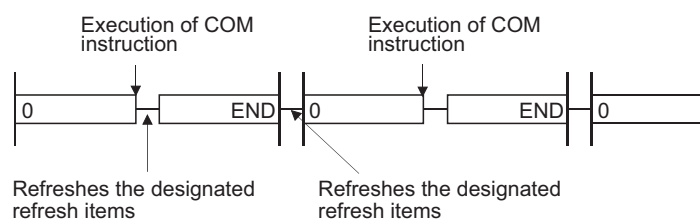
(Set only b2 and b15 of SD778 to 1 (SD778: 8004H).)

**POINT**

Refresh between the multiple CPUs by the COM instruction is performed under the following condition.

- Receiving operation from other CPUs : When b4 of SD778 (auto refresh in the CPU shared memory) is 1.
- Sending operation from host CPU : When b15 of SD778 (communication with peripheral device is executed/not executed) is 0

- (4) Upon the execution of the COM instruction, the CPU module suspends the processing of the sequence program, and refreshes the designated refresh item.



- (5) The COM instruction can be used in a sequence program any number of times. However, note that the sequence program scan time will be lengthened by the time taken for refresh time of the communication with peripheral devices and refresh item that are selected in SD778.
- (6) Only with the Universal model QCPU, interruption is enabled during the execution of the COM instruction. However, note that the data can be separated if the refresh data is used by an interrupt program etc.
- (7) With the Built-in Ethernet port QCPU, service processing time may be increased if the processing was executed by the COM instruction while the built-in Ethernet ports are in Ethernet connection.
- (8) Refresh items for the COM instruction are indicated in the following table.

CPU Module Type Name	Function Version	Serial No.	SM775	Refresh Item
Q00JCPU Q00CPU Q01CPU	A	"04021" or lower	OFF	Refreshes all of the refresh items.
			ON	Communication with peripheral device only.
Q02CPU Q02HCPU Q06HCPU Q12HCPU Q25HCPU	B	"04122" or higher	OFF	Refreshes all of the refresh items.
			ON	Refreshes the refresh items selected by SD778.
Q02PHCPU Q06PHCPU	—	—	ON/OFF	Refreshes all of the refresh items.
			OFF	Refreshes all of the refresh items.
			ON	Communication with peripheral device only.
			ON	Refreshes the refresh items selected by SD778.
Q12PHCPU Q25PHCPU	C	"04011" or lower	OFF	Refreshes all of the refresh items.
			ON	Communication with peripheral device only.
		"07032" or higher	OFF	Refreshes all of the refresh items.
			ON	Refreshes the refresh items selected by SD778.
Q12PRHCPU Q25PRHCPU	D	—	OFF	Refreshes all of the refresh items.
			ON	Refreshes the refresh items selected by SD778.
Q02UCPU Q03UDCPU Q04UDHCPU Q06UDHCPU Q03UDECPU Q04UDEHCPU Q06UDEHCPU Q13UDEHCPU Q26UDEHCPU	B	—	OFF	Refreshes the refresh items selected by SD778.
			ON	Refreshes all of the refresh items.

**☒ POINT**

---

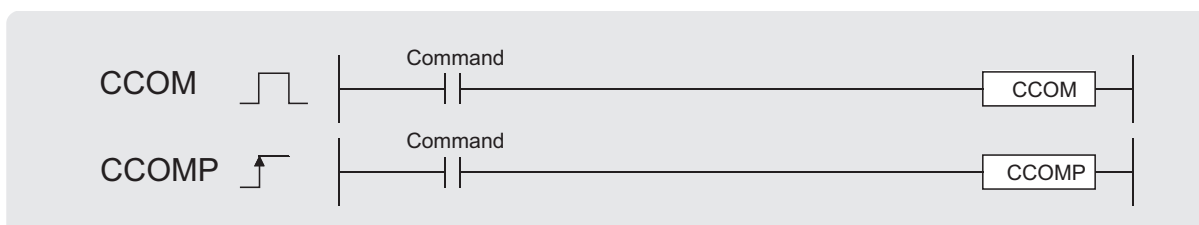
1. The COM instruction cannot be used in low speed execution type programs, fixed scan execution type programs or interrupt programs.
  2. For the redundant CPU, there are restrictions on use of the COM instruction. Refer to the manual below for details.
    - QnPRHCPU User's Manual (Redundant System)
-

## 7.6.11 Select Refresh Instruction (CCOM)



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Setting Data	Internal Devices		Z, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

7

- (1) When the CCOM(P) instruction is executed, the following refresh operations can be executed.
  - I/O refresh
  - CC-Link refresh
  - CC-Link IE controller network refresh
  - MELSECNET/H refresh
  - Auto refresh of intelligent function modules
  - Auto refresh using QCPU standard area of multiple CPU system
  - Reading input or output data of all modules other than the multiple CPU system group
  - Auto refresh using the multiple CPU high speed transmission area of multiple CPU system
  - Communication with peripheral devices
- (2) Turning off SM775 refreshes all refresh items except I/O refresh.

## (3) When refresh items are selected

## (a) Specify refresh items for SD778, and set SM775 to on.

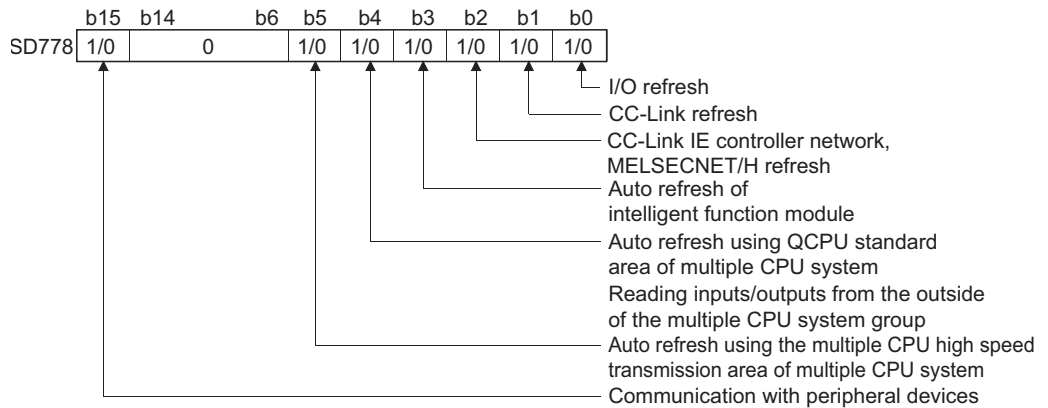
The following table shows the refresh items that can be specified by turning SM775 on or off and in SD778.

Refresh item	When SM775 is off	When SM775 is on
I/O refresh	Not executed	Whether to be executed or not can be selected.
CC-Link refresh	Executed	
CC-Link IE controller network refresh		
MELSECNET/H refresh		
Auto refresh of intelligent function modules		
Auto refresh using QCPU standard area of multiple CPU system		
Reading input or output data of all modules other than the multiple CPU system group		
Auto refresh using the multiple CPU high speed transmission area of multiple CPU system		
Communication with peripheral devices		

## (b) Select refresh items using b0 to b5 and b15 of SD778.

Whether to execute each bit of SD778 or not can be specified as shown below:

Bit of SD778	Executed	Not executed
b0 to b5	1	0
b15	0	1

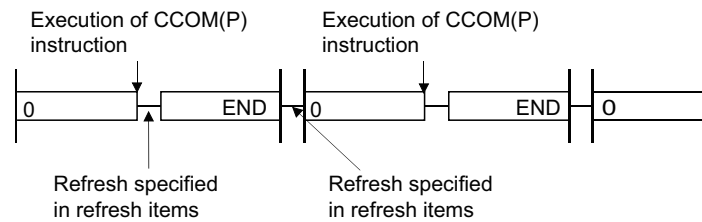


### POINT

Refresh between the multiple CPUs by the CCOM(P) instruction is executed under the following condition.

- Receiving operation from other CPUs: When b4 of SD778 (auto refresh in the CPU shared memory) is 1.
- Sending operation from host CPU: When b15 of SD778 (communication with peripheral device) is 0.

- (4) On the execution of the CCOM (P) instruction, the CPU module suspends the processing of the sequence program, and refreshes the specified refresh item.



- (5) The CCOM(P) instruction can be used in a sequence program any number of times. However, note that the sequence program scan time will be lengthened by the time taken for refresh item that are specified in SD778.
- (6) Interruption is enabled during the execution of the CCOM(P) instruction. However, note that the data can be separated if the refresh data is used for an interrupt program.
- (7) The CCOM(P) instruction is not available for the fixed scan execution type program or interrupt program.
- (8) With the Built-in Ethernet port QCPU, service processing time may be increased if the processing was executed by the CCOM instruction while the built-in Ethernet ports are in Ethernet connection.

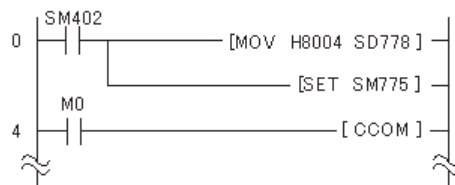
## ! Operation Error

- (1) When the CCOM(P) instruction is executed in the QnUD(H)CPU whose serial number (first five digits) is "10101" or later, an error occurs. (Error code: 4100)

## Program Example

- (1) Turning on M0 enables the program to execute the select refresh, while turning off M0 disables the program to execute the select refresh.

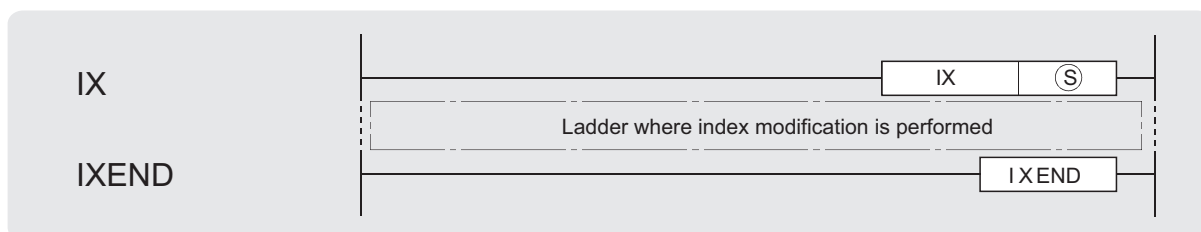
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM402
1	MOV	H8004 SD778
3	SET	SM775
4	LD	M0
5	CCOM	

## 7.6.12 Index modification of entire ladder (IX,IXEND)



Ⓢ: Head number of the devices where index modification data is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○					—		

### ★ Function

- Performs index modification on all devices in the ladder up to the IXEND instruction after the IX instruction, using the index modification value specified in the index modification table. Refer to 7.6.13 for how to configure an index modification table.

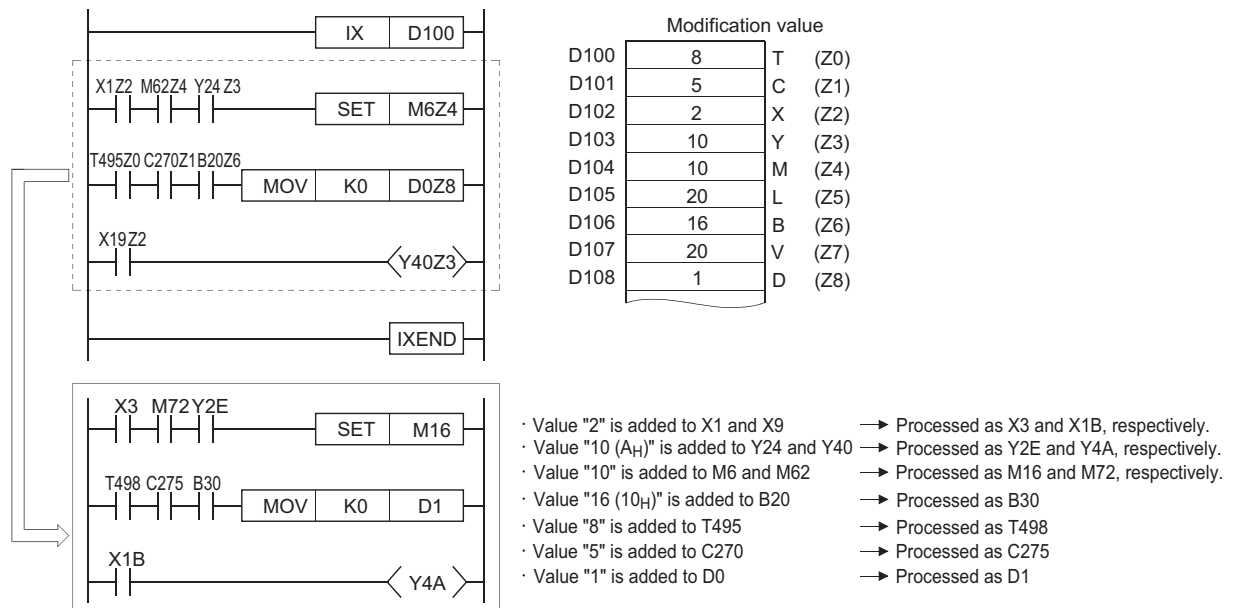
The configuration of the index modification table and the corresponding index register numbers are as shown below:

	Device name	Index register number		Device name	Index register number
Ⓢ	Modification value of timer (T)	Z0	Ⓢ + 8	Modification value of data register (D)	Z8
Ⓢ + 1	Modification value of counter (C)	Z1	Ⓢ + 9	Modification value of link register (W)	Z9
Ⓢ + 2	Modification value of input (X)	Z2	Ⓢ + 10	Modification value of file register (R)	Z10
Ⓢ + 3	Modification value of output (Y)	Z3	Ⓢ + 11	Modification value of buffer register I/O No. (U)	Z11
Ⓢ + 4	Modification value of internal relay (M)	Z4	Ⓢ + 12	Modification value of buffer register (G)	Z12
Ⓢ + 5	Modification value of latch relay (L)	Z5	Ⓢ + 13	Modification value of link direct device network No. (J)	Z13
Ⓢ + 6	Modification value of link relay (B)	Z6	Ⓢ + 14	Modification value of file register (ZR)	Z14
Ⓢ + 7	Modification value of edge relay (V)	Z7	Ⓢ + 15	Modification value of pointer (P)	Z15

\*1: When using a basic model QCPU, index registers with numbers from Z10 onward cannot be used.

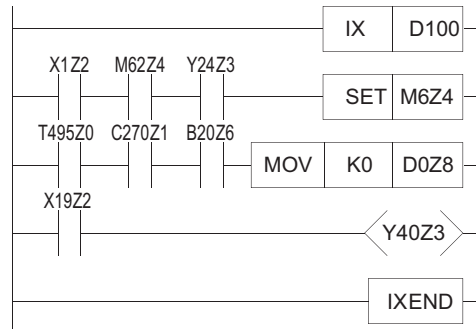


- (2) Index modification for device numbers is accomplished in the manner as below: By setting a modification value to each of the devices, the set modification values are added to the all device numbers of the devices used in the ladder between the IX and IXEND instructions. The program is executed using the index modified device numbers.



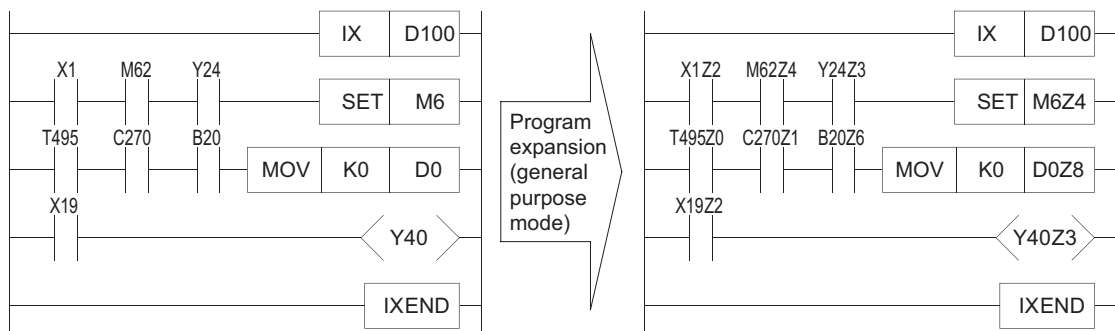
- (3) Instructions such as the PLS, PLF, and □□□P instructions, which are executed only once when input conditions have been established, cannot be index modified by using the IX to IXEND instruction loop.
- (4) In cases where adding the modification value causes the device number to exceed the device range, accurate processing will not be conducted.
- (5) Do not execute the IX or IXEND instructions during online program changes of sequence programs (write during RUN). Accurate processing will not be conducted if this happens.
- (6) Modification values are preset for random word devices as BIN values, and the initial device number for which modification values have been set is designated by ⑤.
- (7) Do not execute a scan execution type program and an interrupt program simultaneously between the IX and IXEND instructions.

- (8) Whether the program will be expanded or a user needs to create the program is depending on your GPP function software package.
- (a) When a user needs to create the program (When GX Developer is used)  
The index register should be added to the index modification ladder established with the IX and IXEND instructions. \*2



\*2 : The value of Zn is returned to the previous Zn value before the execution of the IX instruction after the IXEND instruction has been executed.

- (b) When the program is expanded (When SW  -GPPQ is used)  
The index modification ladder established with the IX and IXEND instructions will be transformed into a ladder using the index register (Zn) during the program expansion. \*3  
Index modification cannot be conducted in a program between the IX and IXEND instructions.



\*3 : The value of Zn is returned to the previous Zn value before the execution of the IX instruction after the IXEND instruction has been executed.

## POINT

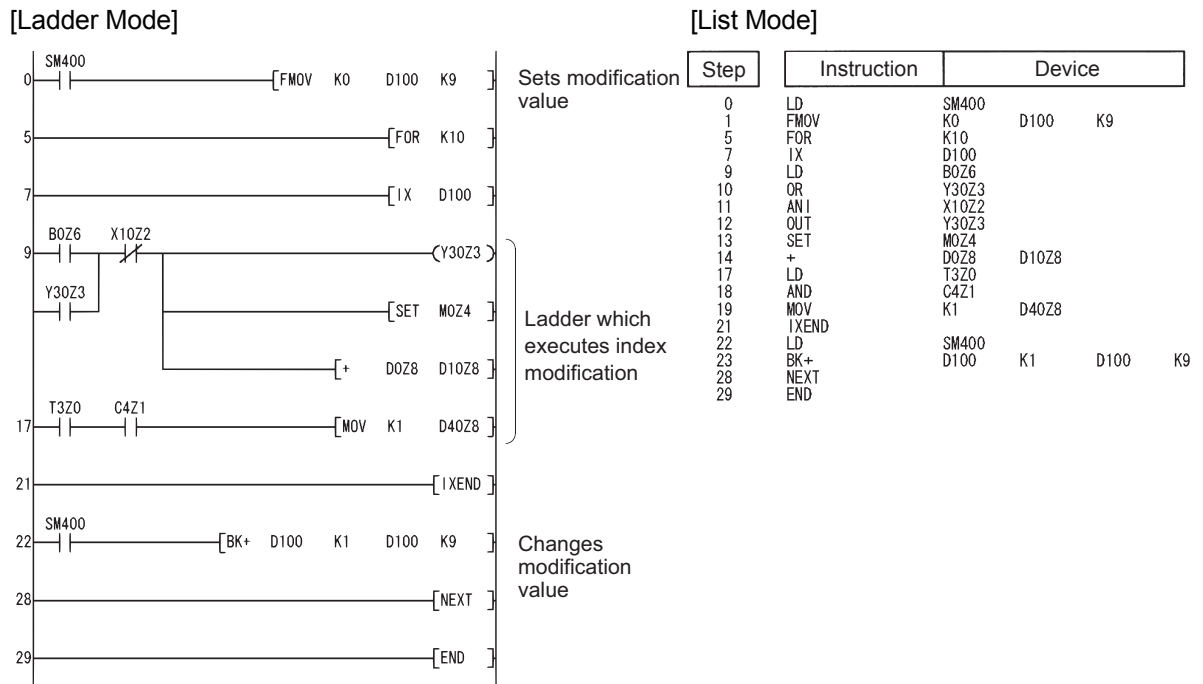
- When using the IX and IXEND instructions in both a normal sequence program and an interrupt sequence program, establish the interlock to avoid simultaneous execution. The interlock assumes the area between the IX and IXEND instructions in the normal sequence program as DI, disabling the interruption.
- The IXDEV and IXSET instructions can be used to specify modification values. Refer to 7.6.13 for details.

## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The IX and IXEND instructions are not used together. (Error code: 4231)
  - An END, FEND, GOEND, or STOP instruction is executed prior to the execution of the IXEND instruction, but after the execution of the IX instruction. (Error code: 4231)

## Program Example

- (1) The following program executes the same ladder 10 times, while changing device numbers.



### [Operation]

Modification value	1st time	2nd time	3rd time	10th time
D100	Modification value of T	B0 → B1	B2	-----> B9
D101	Modification value of C	X10 → X11	X12	-----> X19
D102	Modification value of X	Y30 → Y31	Y32	-----> Y39
D103	Modification value of Y	M0 → M1	M2	-----> M9
D104	Modification value of M	D0 → D1	D2	-----> D9
D105	Modification value of L	D10 → D11	D12	-----> D19
D106	Modification value of B	T3 → T4	T5	-----> T12
D107	Modification value of V	C4 → C5	C6	-----> C13
D108	Modification value of D	D40 → D41	D42	-----> D49

## 7.6.13 Designation of modification values in index modification of entire ladders (IXDEV,IXSET)

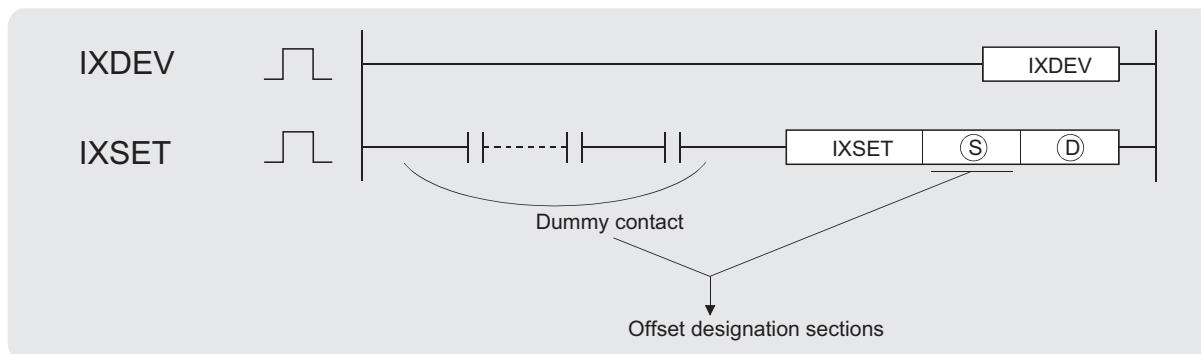
Basic

High performance

Process

Redundant

Universal



Ⓢ : Head number of the devices where index modification data is stored (pointer only) P□ (Pointer)

Ⓣ : Head number of the devices where index modification data will be stored (except a pointer) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J□□		U□\G□	Zn	Constants	Other P
	Bit	Word		Bit	Word				
Ⓢ	—	—				—			○
Ⓣ	—	○				—			—

### ★ Function

- (1) The IXDEV and IXSET instructions are used to configure an index modification table used in the IX and IXEND instructions.
- (2) The device offset value designated at the offset designation area is set at the index modification table designated by Ⓣ.
- (3) The value 0 will be entered if no designation is made.
- (4) Word devices are also indicated by contact (word device bit designation). Data register 10 (D10) is designated with D10.0.  
(Any value from 0 to F can be used for the bit number.)
- (5) Designation is made according to the method described below. \*1 (The symbol □ is where the offset value will be. The notation XX indicates random selection.)

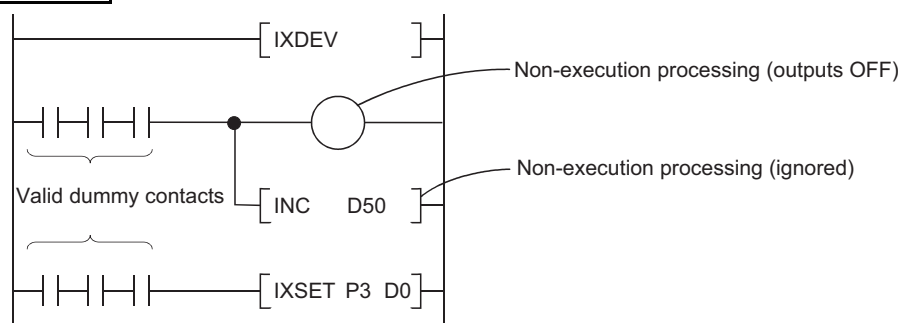
Device	T	C	X	Y	M	L	V	B
Designation method	T□ —	C□ —	X□ —	Y□ —	M□ —	L□ —	V□ —	B□ —
Device	D	W	R	U/G		J		ZR
Designation method	D□.XX —	W□.XX —	R□.XX —	U□\G□.XX —		J□\B□ <sup>*2</sup> —		ZR□.XX —
Device	P							
Designation method	— IXSET (Ⓢ) (Ⓣ) — <sup>*3</sup>							

\*1: When using a basic model QCPU, the devices R, U/G, J, ZR and P cannot be used.

\*2: Devices following J□\ designate B, W, X, or Y, and the offset value is also set in correspondence with this.

\*3: When using a basic model QCPU, specify a dummy device number. Ⓢ is P□.

- (6) If two offsets for two identical types of device have been set in the offset designation area, the last value set will be valid.
- (7) The IXDEV and IXSET instructions should be treated as a pair.
- (8) Any value from 0 to 32767 is valid for ZR. (The offset value will be the remainder of the quotient of the designated device number divided by 32768.)
- (9) The dummy contacts in the offset specifying part are valid for only LD and AND located within the range of the IXDEV-IXSET instructions. The IXDEV-IXSET instructions will not be executed if other instructions are described.

**Example**

## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The IXDEV and IXSET instructions have not been used as a pair. (Error code: 4231)

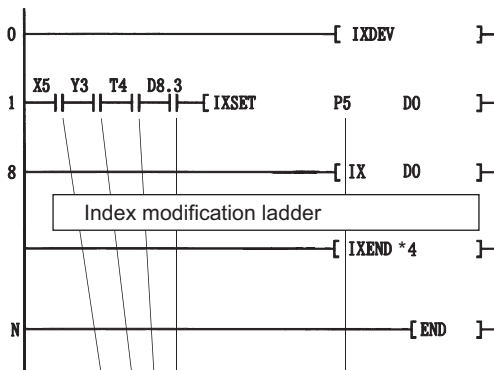
## Program Example

- (1) The following program changes the modification values for input (X), output (Y), data register (D) and pointer (P).

When using a basic model QCPU, the devices R, U/G, J, ZR and P cannot be used.

[Ladder Mode]

[List Mode]



Step	Instruction	Device
0	IXDEV	
1	LD	X5
2	AND	Y3
3	AND	T4
4	AND	D8.3
5	IXSET	P5
		DO
8	IX	DO
:	IXEND	
N	END	

Index modification table

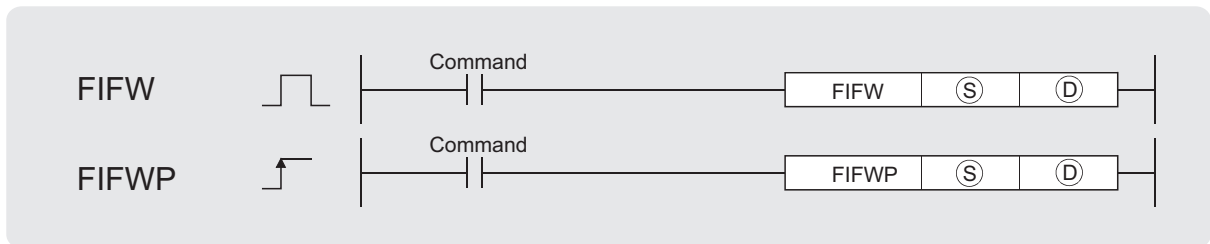
D0	4	T
	0	C
	5	X
	3	Y
	0	M
	0	L
	0	B
	0	V
	8	D
	0	W
	⋮	
	0	
D15	0	P

\*4: Refer to 7.6.12 for index modification using the IX to IXEND instructions.

# 7.7 Data Table Operation Instructions

## 7.7.1 Writing data to the data table (FIFW(P))

Basic High performance Process Redundant Universal



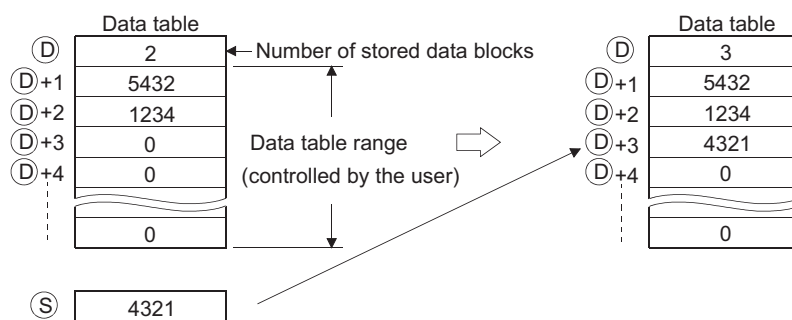
Ⓢ : Data to be written into the table or the number of the device where the data is stored (BIN 16 bits)  
 Ⓣ : Head number of the table (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○		○	—
Ⓣ	—	○				—		—	—

7

### ★ Function

- (1) Stores the 16-bit data designated by Ⓢ in the data table designated by Ⓣ. The number of data blocks stored in the table is stored at Ⓣ, and the data designated by Ⓢ is stored in sequence from Ⓣ+1.



- (2) The first time the FIFW instruction is executed, any values in the designated by Ⓣ device should be cleared.
- (3) The number of data blocks to be written in the data table and the data table range should be controlled by the user.  
 [See Program Example (2)]

7.7 Data Table Operation Instructions  
7.7.1 Writing data to the data table (FIFW(P))

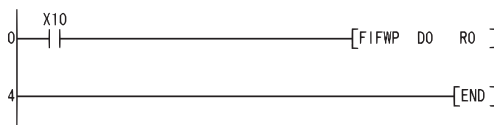
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data table range exceeds the relevant device range when the FIFW instruction is executed. (Error code: 4101)

## Program Example

- (1) The following program stores the data at D0 to the data table following R0 when X10 is turned ON.

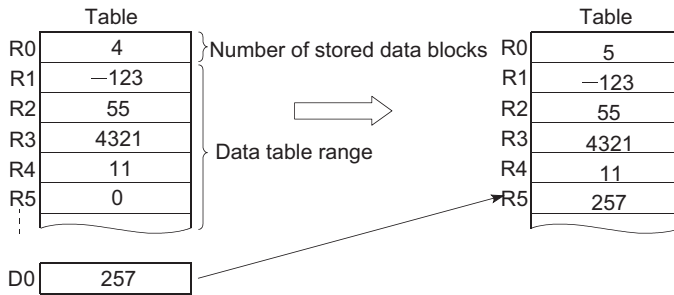
[Ladder Mode]



[List Mode]

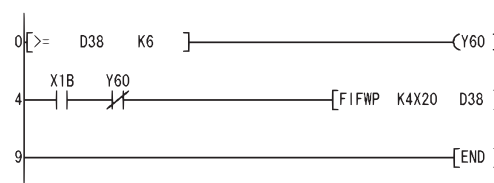
Step	Instruction	Device
0	LD	X10
1	FIFW	D0 R0
4	END	

[Operation]



- (2) The following program stores the data at X20 to X2F to data table of D38 to D44 table when X1B is turned ON, and, if there are more than 6 data blocks to be stored, turns Y60 ON and disables the FIFW instruction.

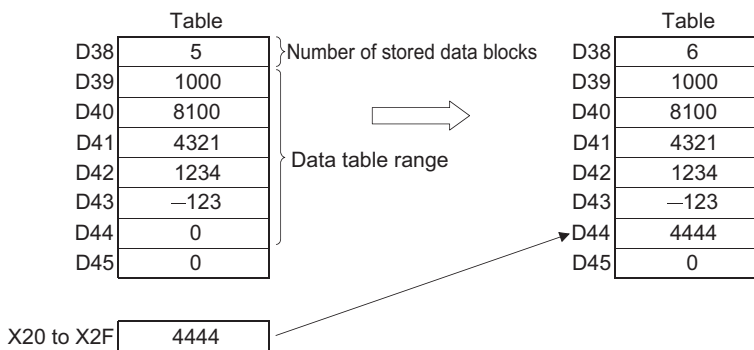
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD>=	D38 K6
3	OUT	Y60
4	LD	X1B
5	ANI	Y60
6	FIFW	K4X20 D38
9	END	

[Operation]





## 7.7.2 Reading oldest data from tables (FIFR(P))

Basic High performance Process Redundant Universal



Ⓢ : Head number of the devices where the data read from the table will be stored (BIN 16 bits)  
 ⓓ : Head number of the table (BIN 16 bits)

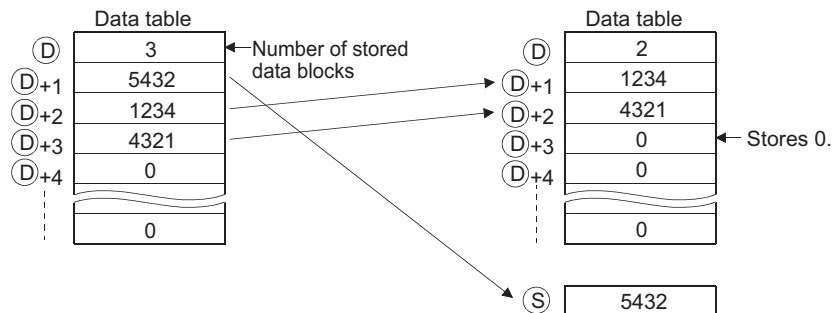
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○		—	
ⓓ	—	○				—		—	

7

### ★ Function

- (1) Stores the oldest data (ⓓ + 1) input to the table designated by ⓓ at the device designated by Ⓢ.

After the execution of the FIFR instruction, the data in the table is all compressed up by one block.



- (2) Users should attempt to avoid executing the FIFR instruction if the value stored at ⓓ is 0. [See Program Example (1)]

### ! Operation Error

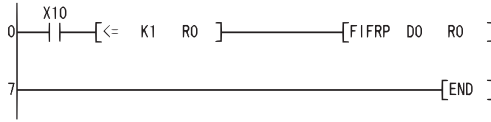
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The FIFR instruction was executed when the value at ⓓ was 0. (Error code: 4100)
  - The data table range exceeded the corresponding device range at execution of the FIFR instruction. (Error code: 4101)

7.7 Data Table Operation Instructions  
 7.7.2 Reading oldest data from tables (FIFR(P))

# Program Example

- (1) The following program stores the R1 data from the table R0 to R7 at D0 when X10 is turned ON.

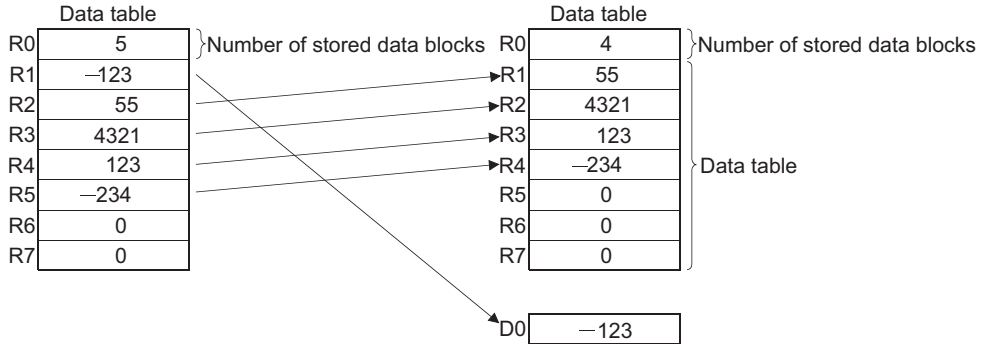
[Ladder Mode]



[List Mode]

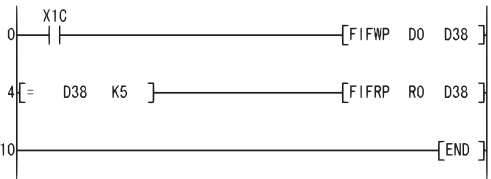
Step	Instruction	Device
0	LD	X10
1	AND<=	K1 R0
4	FIFRP	D0 R0
7	END	

[Operation]



- (2) The following program stores the data at D0 in the data table D38 to D43, and, when the table stores 5 data, stores the data at D39 of the data table in R0, when X1C is turned ON.

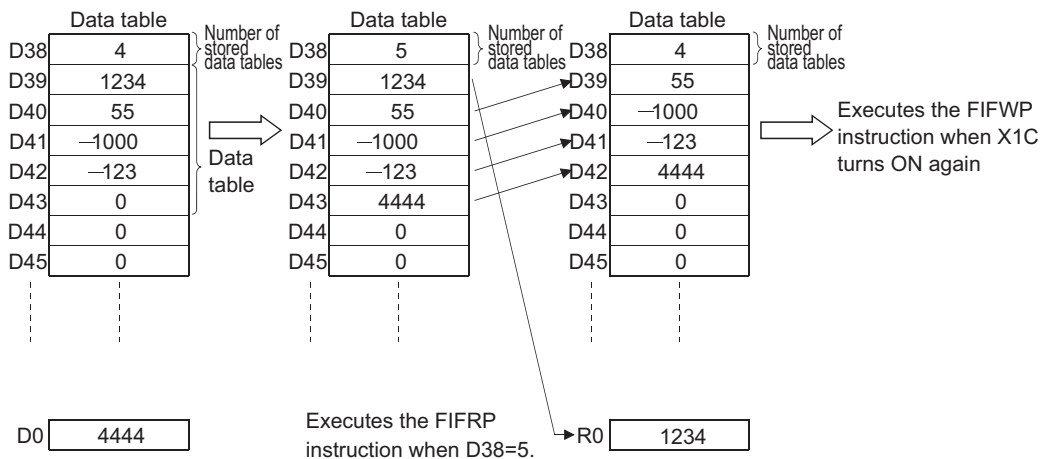
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	FIFWP	D0 D38
4	LD=	D38 K5
7	FIFRP	R0 D38
10	END	

[Operation]



### 7.7.3 Reading newest data from data tables (FPOP(P))

Basic High performance Process Redundant Universal



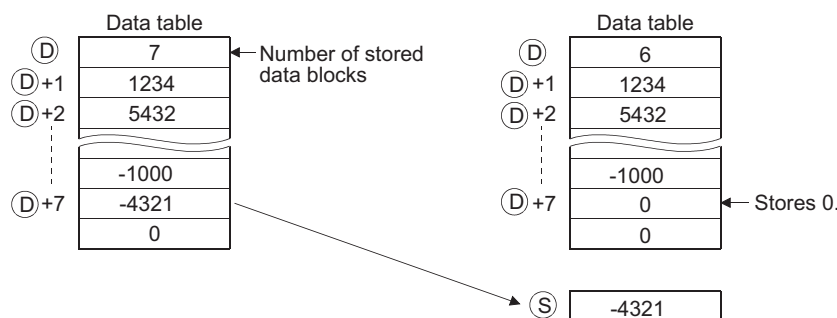
Ⓢ : Head number of the devices where the data read from the table will be stored (BIN 16 bits)

Ⓣ : Head number of the table (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○			—
Ⓣ	—		○			—			—

#### ★ Function

- (1) Stores the newest data input to the table designated by Ⓣ at the device designated by Ⓢ. After the execution of the FPOP instruction, the device storing the data read by the FPOP instruction is reset to 0.



- (2) Perform interlock to avoid executing the FPOP instruction when the value stored at Ⓣ is 0. [See Program Example (1)]

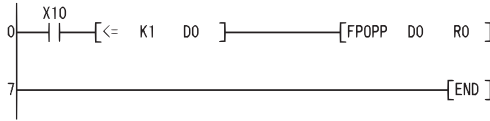
#### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The FPOP instruction was executed when the value of Ⓣ was 0. (Error code: 4100)
  - The data table range exceeded the corresponding device range at execution of the FPOP instruction. (Error code: 4101)

# Program Example

- (1) The following program stores the data stored last in the data table R0 to R7 at D0 when X10 is turned ON.

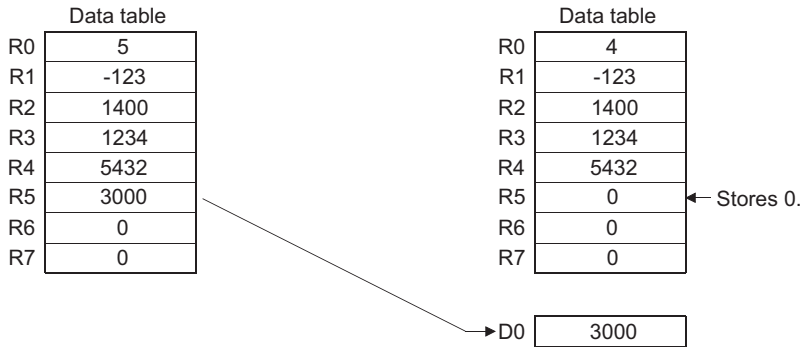
[Ladder Mode]



[List Mode]

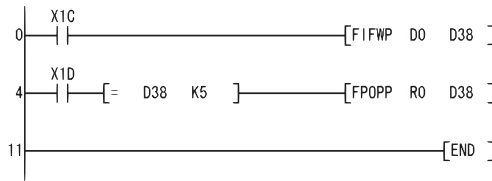
Step	Instruction	Device
0	LD	X10
1	AND<=	K1 D0
4	FPOPP	D0 R0
7	END	

[Operation]



- (2) The following program stores the data at D0 in the data table D38 to D43 when X1C is turned ON, and when the number of data stores in the table reaches 5, turns X1D ON, and stores the data stored last in the data table to R0.

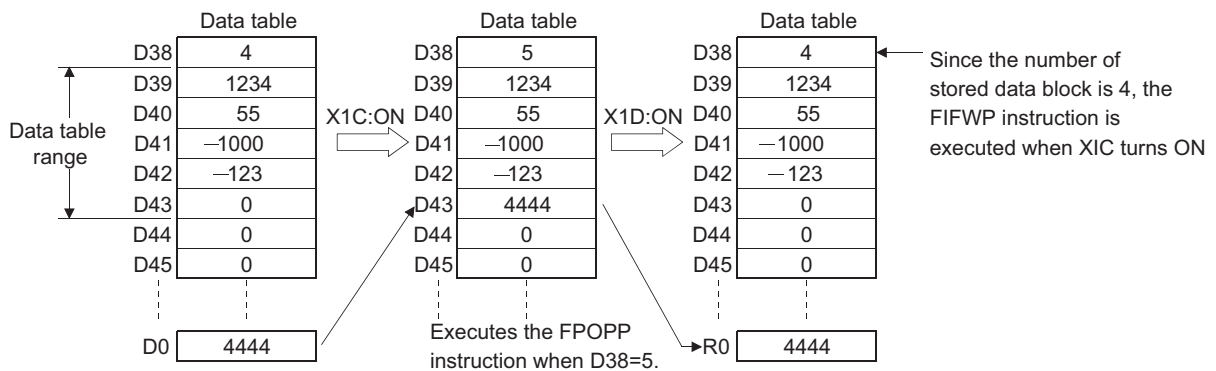
[Ladder Mode]



[List Mode]

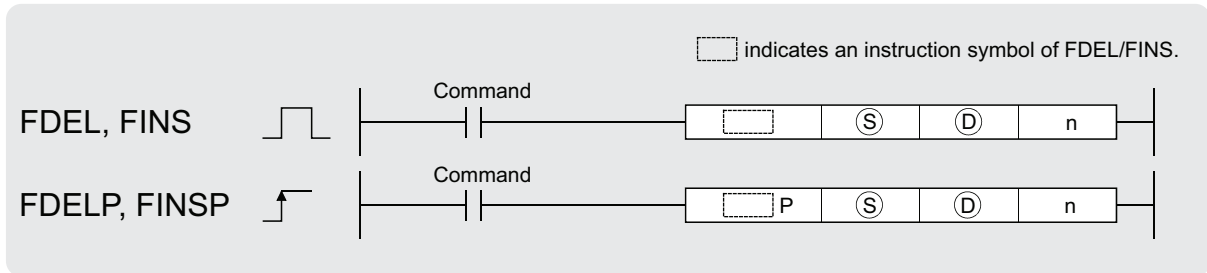
Step	Instruction	Device
0	LD	X1C
1	FIFWP	D0 D38
4	LD	X1D
5	AND=	D38 K5
8	FPOPP	R0 D38
11	END	

[Operation]



## 7.7.4 Deleting and inserting data from and in data tables (FDEL(P),FINS(P))

Basic High performance Process Redundant Universal



- Ⓢ : Head number of the devices where data to be inserted is stored (BIN 16 bits)
- Ⓣ : Head number of the devices where the data to be deleted will be stored (BIN 16 bits)
- Ⓧ : Head number of the table (BIN 16 bits)
- n : Location on the table where data is inserted/deleted (BIN 16 bits)

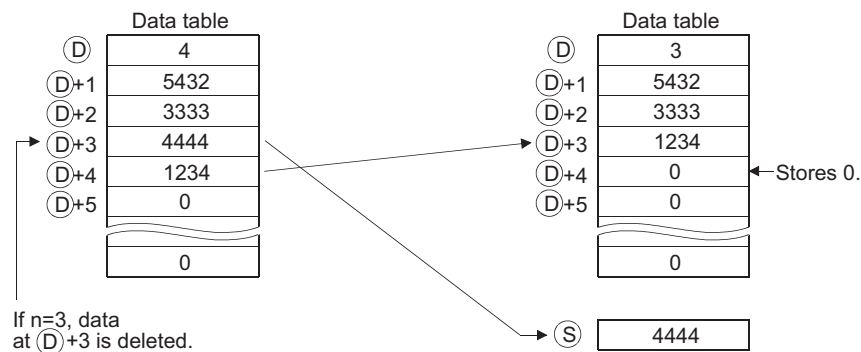
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○		—	—
Ⓧ	—	○				—		—	—
n	○	○				○		○	—

### ★ Function

#### FDEL

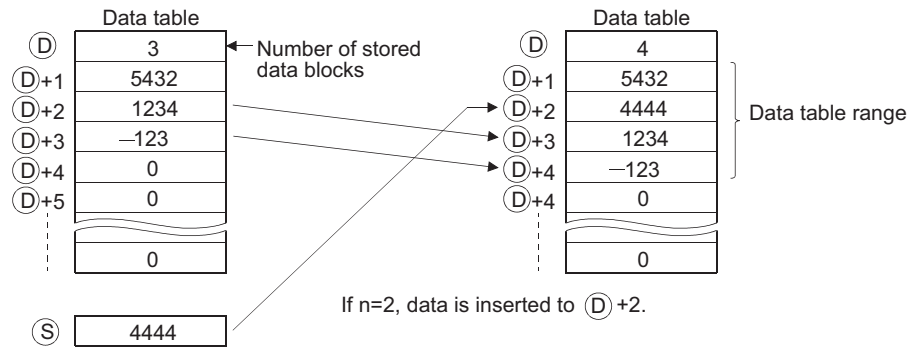
- (1) Deletes the nth block of data from the data table designated by Ⓧ, and stores it at the device designated by Ⓢ.

After the execution of the FDEL instruction, the data in the table following the deleted block is compressed forward by one block.



## FINS

- (1) Inserts the 16-bit data designated by  $\textcircled{S}$  at the  $n$ th block of the data table designated by  $\textcircled{D}$ . After the execution of the FINS instruction, the data in the table following the inserted block is all dropped one position.



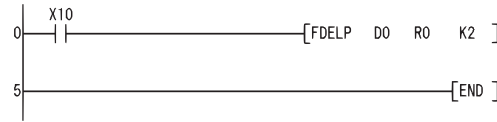
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $N$ th position from  $\textcircled{D}$  is larger than the data storage number at the execution of the FDEL instruction. (Error code: 4101)
  - The  $N$ th position from  $\textcircled{D}$  is larger than the "data storage number + 1" at the execution of the FINS instruction. (Error code: 4101)
  - The value of  $n$  in the case of the FDEL, FINS instruction exceeds the device range of the table  $\textcircled{D}$ . (Error code: 4101)
  - The FDEL or FINS instruction was executed when  $n = 0$ . (Error code: 4100)
  - The FDEL instruction was executed when the value of  $\textcircled{D}$  was 0. (Error code: 4100)
  - The data table range exceeded the corresponding device range at execution of the FDEL or FINS instruction. (Error code: 4101)

# Program Example

- (1) The following program deletes the second data from the table R0 to R7 and stores the deleted data at D0 when X10 is turned ON.

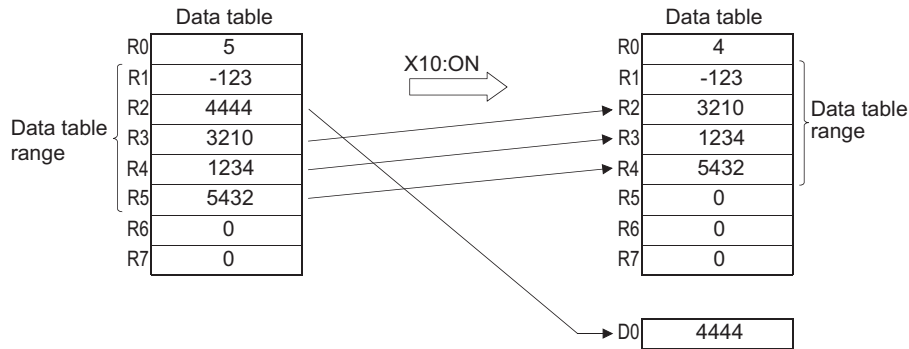
[Ladder Mode]



[List Mode]

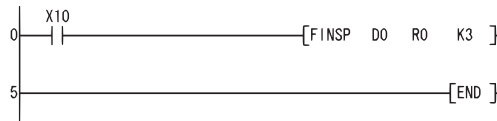
Step	Instruction	Device
0	LD	X10
1	FDEL P	D0 R0 K2
5	END	

[Operation]



- (2) The following program inserts the data at D0 into the third position at the table R0 to R7 when X10 is turned ON.

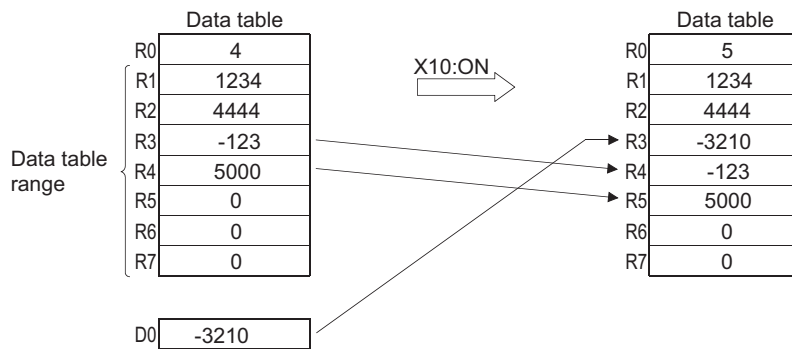
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	FINS P	D0 R0 K3
5	END	

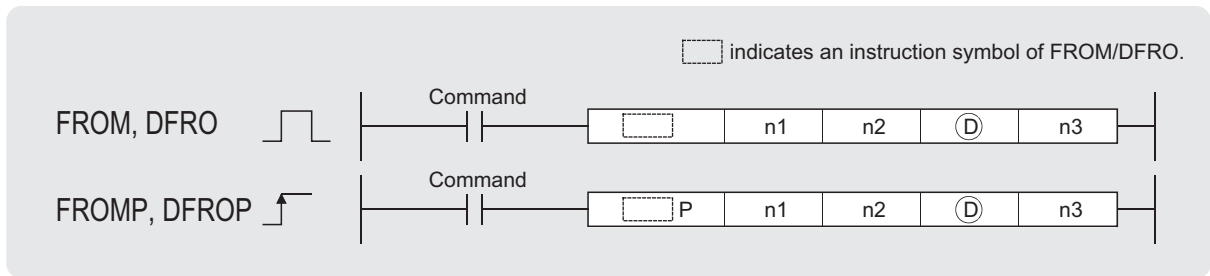
[Operation]



## 7.8 Buffer memory access instruction

### 7.8.1 Reading 1-/2-word data from the intelligent function module (FROM(P),DFRO(P))

Basic High performance Process Redundant Universal



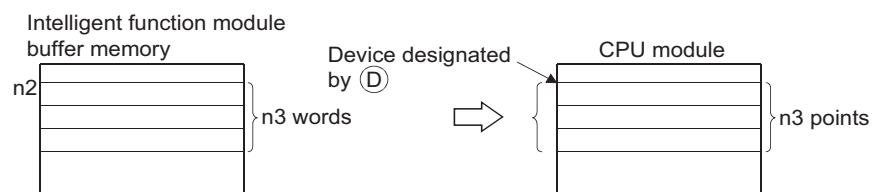
n1 : Head I/O number of an intelligent function module (BIN 16 bits)  
 n2 : Head address of data to be read (BIN 16 bits)  
 (D) : Head number of the devices where the read data will be stored (BIN 16/32 bits)  
 n3 : Number of data blocks to be read (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants K, H	Other U
	Bit	Word		Bit	Word				
n1		○				○			○
n2		○				○			—
(D)		○				—			—
n3		○				○			—

### ★ Function

#### FROM

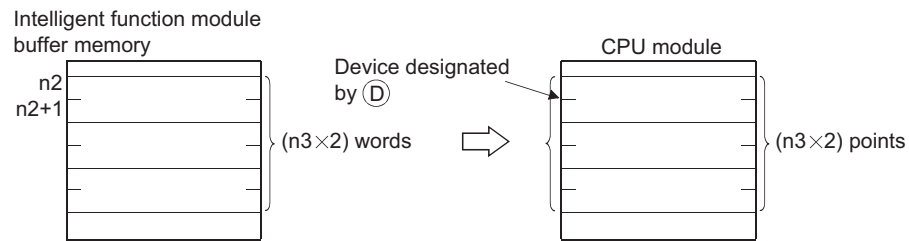
- (1) Reads the data in n3 words from the buffer memory address designated by n2 of the intelligent function module designated by n1, and stores the data into the area starting from the device designated by (D).





## DFRO

- (1) Reads the data in  $(n3 \times 2)$  words from the buffer memory address designated by  $n2$  of the the intelligent function module designated by  $n1$ , and stores the data into the area starting from the device designated by  $\textcircled{D}$ .



### POINT

Data read from intelligent function modules is also possible with the use of an intelligent function module device.

For the intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

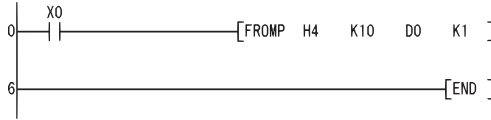
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- There has been no exchange of signals with an intelligent function module at the execution of the instruction. (Error code: 1412)
  - An error has been detected in an intelligent function module at the execution of the instruction. (Error code: 1402)
  - The I/O number designated by  $n1$  is not for the intelligent function module. (Error code: 2110)
  - The range of  $n3$  points ( $2 \times n3$  points for DFRO) from the device designated by  $\textcircled{D}$  exceeds the designated device range. (Error code: 4101)
  - The address designated by  $n2$  is outside the buffer memory range. (Error code: 4101)

## Program Example

- (1) The following program reads the digital value of CH1 of the A68AD mounted at I/O numbers 040 to 05F into D0 when X0 is turned ON. (Reads 1 word of data from address 10 of the buffer memory.)

[Ladder Mode]

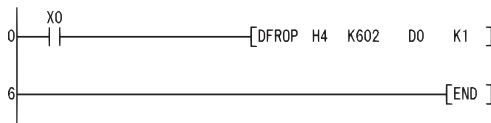


[List Mode]

Step	Instruction	Device
0	LD	X0
1	FROMP	H4 K10 D0 K1
6	END	

- (2) The following program reads the X-axis present value of the AD71 mounted at the I/O numbers 040 to 05F into D0 and D1, when X0 is turned ON. (Reads data in 2 words from the address 602 and 603 of the buffer memory.)

[Ladder Mode]

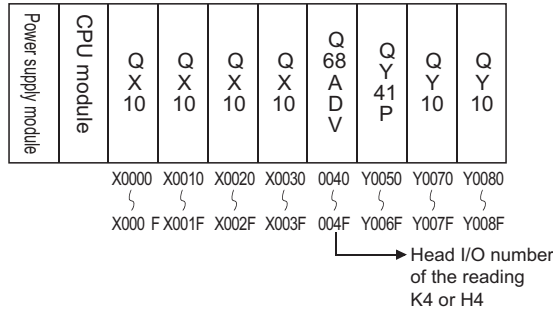


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DFROP	H4 K602 D0 K1
6	END	

### Remark

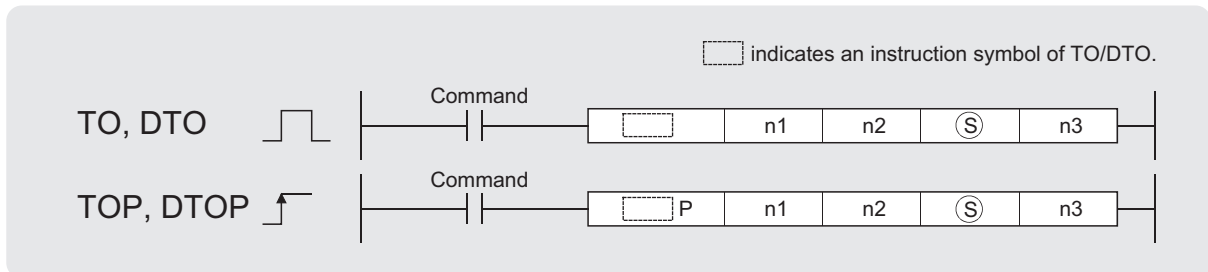
- The value of n1 is specified by the upper 3 digits of hexadecimal 4-digit representation of the head I/O number of the slot in which an intelligent function module is mounted.



- QCPU establishes the automatic interlock of the FROM/DFRO instructions.

## 7.8.2 Writing 1-/2-word data to intelligent function module (TO(P), DTO(P))

Basic High performance Process Redundant Universal



n1 : Head I/O number of an intelligent function module (BIN 16 bits)

n2 : Head address of the area where data is written (BIN 16 bits)

S : Data to be written or head number of the devices where the data to be written is stored (BIN 16/32 bits)

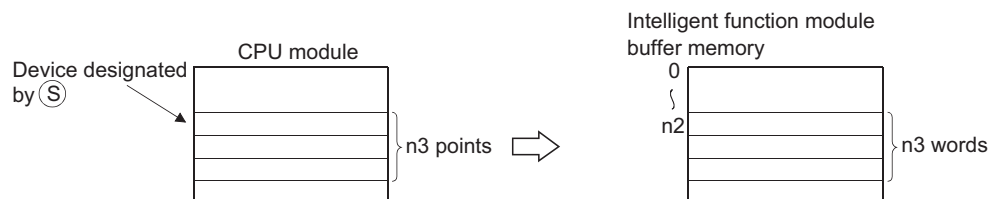
n3 : Number of data blocks to be written (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other U
	Bit	Word		Bit	Word				
n1		○				○		○	○
n2		○				○		○	—
S		○				—		○	—
n3		○				○		○	—

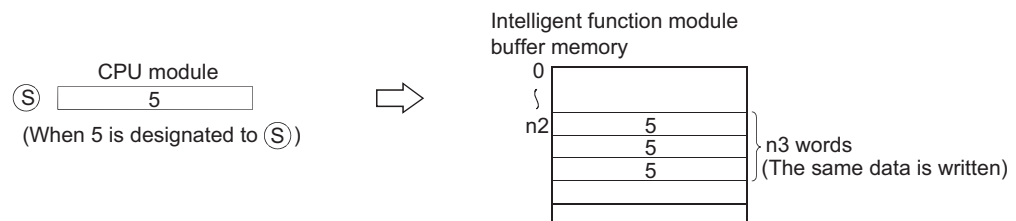
## ★ Function

## TO

Writes the data stored in n3 points starting from the device designated by S into the area starting from buffer memory address designated by n2 of the intelligent function module

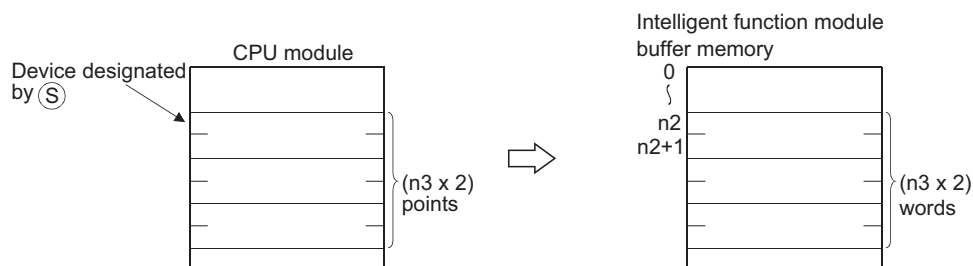


When a constant is designated to S, writes the same data (value designated to S) to the area of n3 points starting from the specified buffer memory. (S can be designated in the following range: -32768 to 32767 or 0H to FFFFH.)

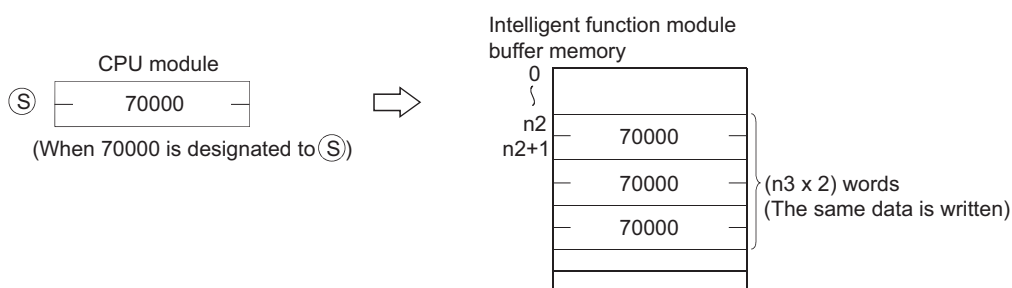


## DTO

Writes the data stored in  $n3 \times 2$  points starting from the device designated by  $\textcircled{S}$  into the area starting from buffer memory address designated by  $n2$  of the intelligent function module designated by  $n1$ .



When a constant is designated to  $\textcircled{S}$ , writes the same data (value designated to  $\textcircled{S}$ ) to the area of  $n3 \times 2$  points starting from the specified buffer memory. ( $\textcircled{S}$  can be designated in the following range: -2147483648 to 2147483647 or 0H to FFFFFFFFH.)



## POINT

Data write to intelligent function modules is also possible with the use of an intelligent function module device.

For the intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

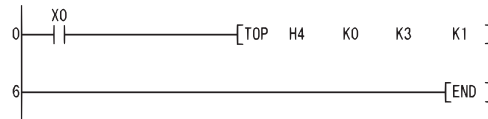
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - There has been no exchange of signals with an intelligent function module at the execution of the instruction. (Error code: 1412)
  - An error has been detected in an intelligent function module at the execution of the instruction. (Error code: 1402)
  - The I/O number designated by  $n1$  is not for the intelligent function module. (Error code: 2110)
  - The  $n3$  points ( $2 \times n3$  points for DTO) of the device designated by  $\textcircled{S}$  exceed the designated device range. (Error code: 4101)
  - The address designated by  $n2$  is outside the buffer memory range. (Error code: 4101)
  - The address designated by  $n2$  is an odd numbered address. (AJ71QC24(N)) (Error code: 4100)

## Program Example

- (1) The following program sets the CH1 and CH2 of the Q68ADV mounted at the I/O numbers 040 to 04F to the "A/D conversion" mode, when X0 is turned ON.  
(Writes 3 into the buffer memory address 0.)

[Ladder Mode]

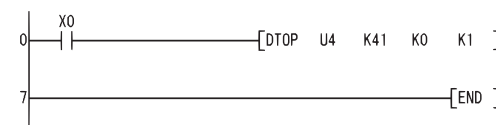


[List Mode]

Step	Instruction	Device
0	LD	X0
1	TOP	H4 K0 K3 K1
6	END	

- (2) The following program sets the X-axis current value of the AD71 mounted at I/O numbers 040 to 05F to 0 when X0 is turned ON. (Writes 0 to addresses 41, 42 of the buffer memory.)

[Ladder Mode]

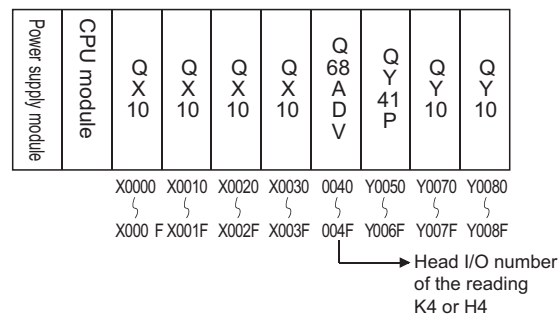


[List Mode]

Step	Instruction	Device
0	LD	X0
1	DTOP	U4 K41 K0 K1
7	END	

### Remark

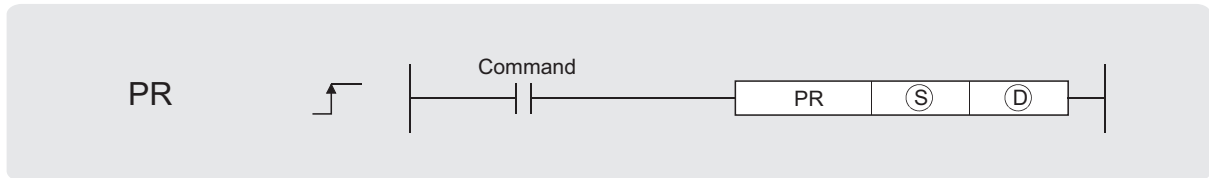
- The value of n1 is specified by the upper 3 digits of hexadecimal 4-digit representation of the head I/O number of the slot in which an intelligent function module is mounted.



- QCPU establishes the automatic interlock of the TO/DTO instructions.

# 7.9 Display instructions

## 7.9.1 Print ASCII code instruction (PR)



Ⓢ : ASCII code or head number of the devices where the ASCII code is stored (character string)

Ⓣ : Head number of the output module to which the ASCII code will be output (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	△*1			—		○	○	—
Ⓣ	○ (Only Y)	—			—		○	—	—

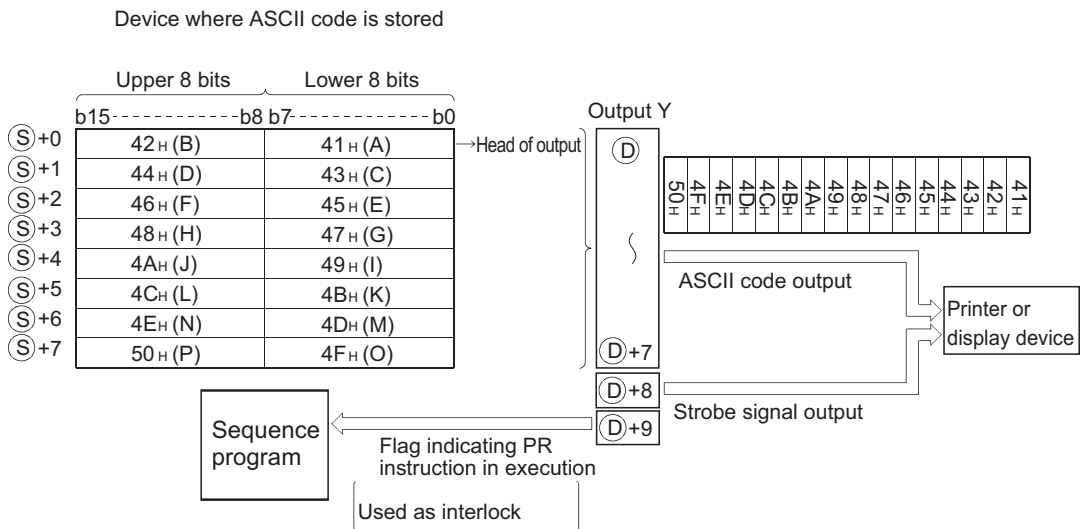
\*1: Local devices and the file registers set for individual programs cannot be used.

### ★ Function

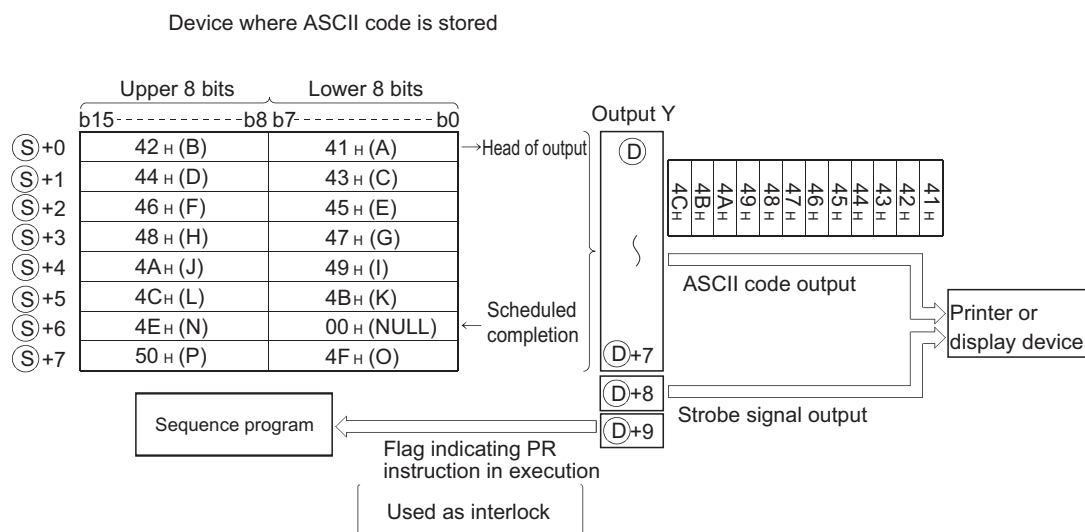
- Outputs ASCII code stored in the device specified by Ⓢ or ASCII code stored in the area starting from the device number to an output module specified by Ⓣ.

The number of characters output differs according to the ON/OFF status of SM701 (number of output characters selection).

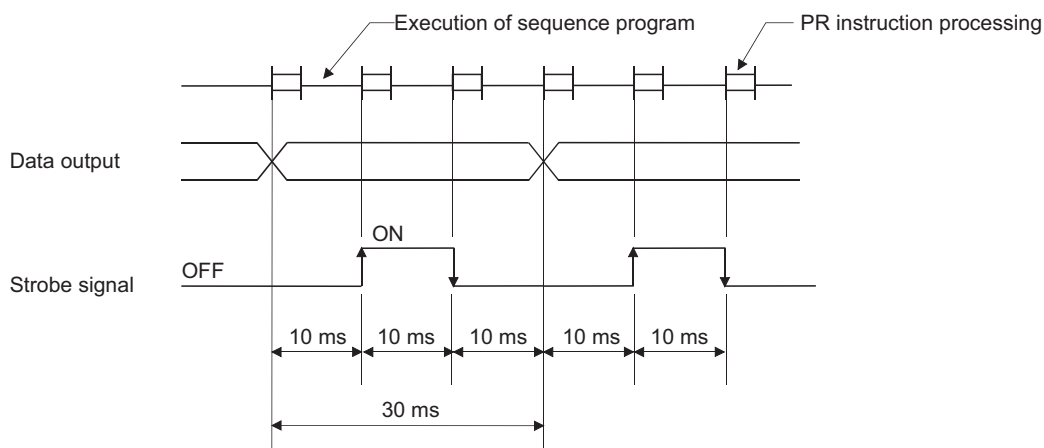
- If SM701 is ON, characters 8 points (16 characters) from the device designated by Ⓢ will be the target of the operation.



- (b) If SM701 is OFF, everything from the device designated by ③ to the 00H code will be the target of the operation.



- (2) The number of points used by the output module is 10 points from the Y address designated by ①.
- (3) Output signals from the output module are transmitted at the rate of 30 ms per character. For this reason, the time required to the completion of the transmission of the designated number of characters (n) will be  $30 \text{ ms} \times n$  (ms). At 10 ms interrupt intervals, the PR instruction executes data output, strobe signal ON, and strobe signal OFF. The other instructions are executed continuously during a period between the above processings.



- (4) In addition to the ASCII code, the output module also outputs a strobe signal (10 ms ON, 20 ms OFF) from the ① + 8 device.
- (5) Following the execution of the PR instruction, the PR instruction execution flag (① + 9 device) remains ON until the completion of the transmission of the designated number of characters.
- (6) The PR and PRC instructions can be used multiple times, but it is preferable to establish an interlock with the PR instruction execution flag (① + 9 device) so that they will not be ON simultaneously.
- (7) If the contents of the device in which ASCII codes are stored changes during the ASCII code output, the modified data after change will be output.

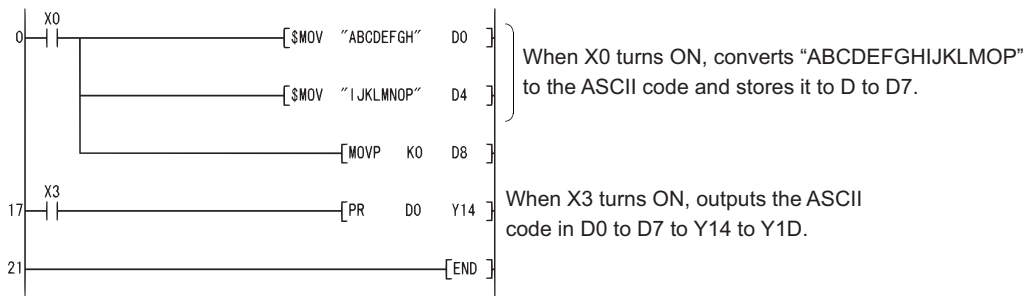
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- There is no 00H code within the range of the device specified by ③ when SM701 is OFF. (Error code: 4101)

## Program Example

- (1) The following program converts the string "ABCDEFGHJKLMNOP" to ASCII code when X0 is turned ON and stores it from D0 to D7, and then outputs the ASCII code at D0 to D7 to Y14 to Y1D when X3 is turned ON.

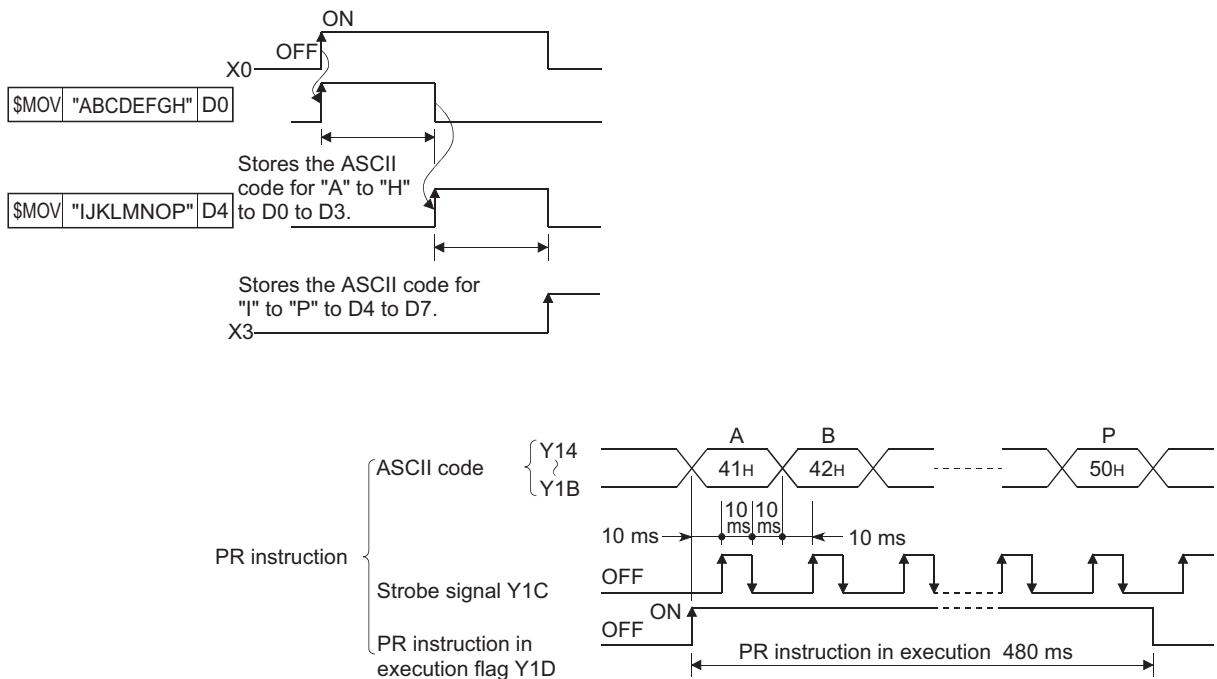
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	\$MOV	"ABCDEFGH" D0
8	\$MOV	"JKLMNOP" D4
15	MOV	K0 D8
17	LD	X3
18	PR	D0 Y14
21	END	

[Timing Chart]

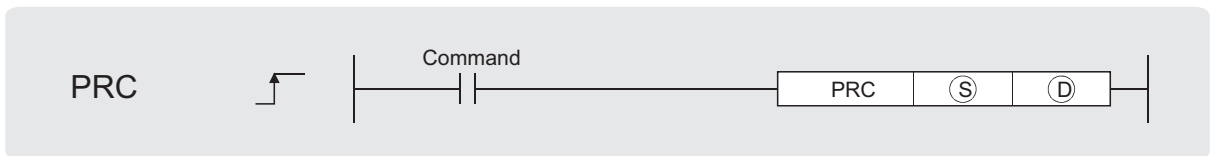




## 7.9.2 Print comment instruction (PRC)



① When High Performance model QCPU/Process CPU is used



Ⓢ : Head number of the device which prints the comment (Device name)

ⓓ : Head number of the output module which outputs the comment (bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other P, I, J, U,
	Bit	Word		Bit	Word				
Ⓢ	○	○			○		—	—	○
ⓓ	○ (Only Y)	—			—		—	—	—

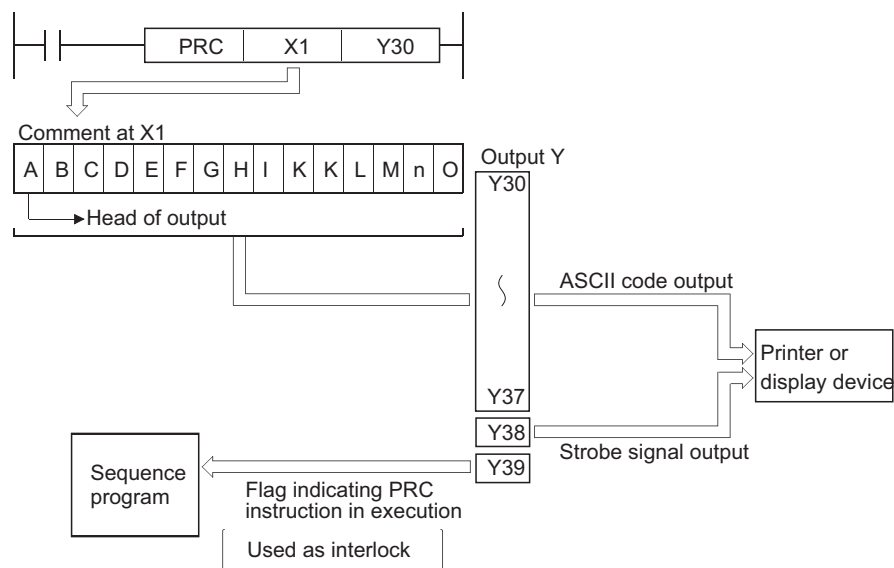
### ★ Function

- (1) Outputs comment (ASCII code) at device designated by Ⓢ to output module designated by ⓓ.

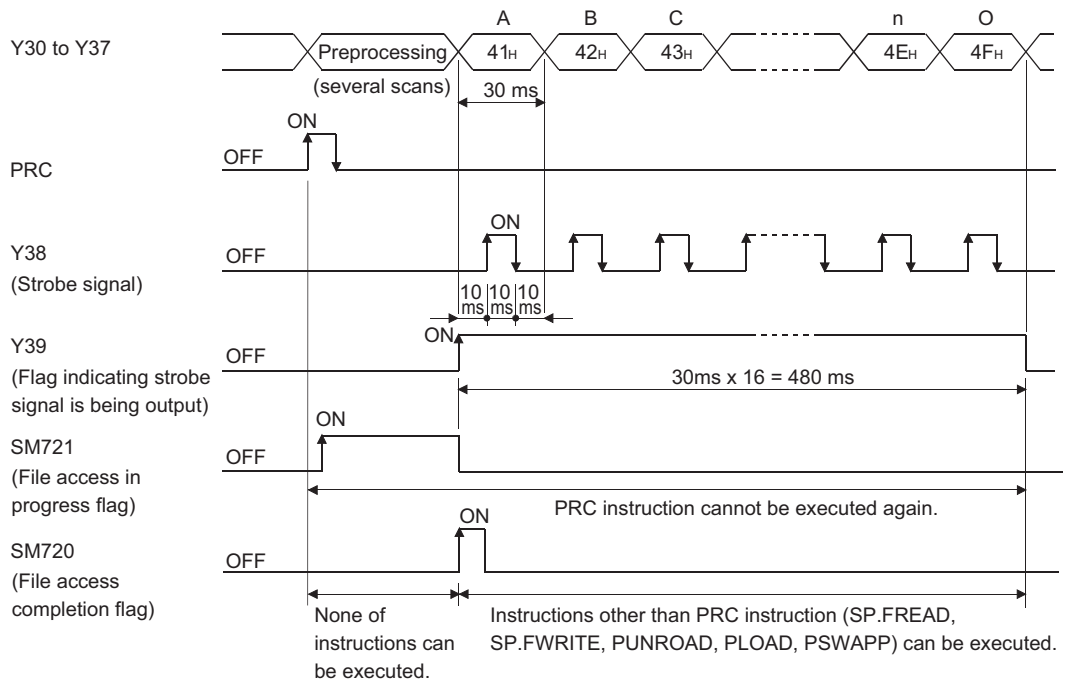
The number of characters output differs according to the ON/OFF status of SM701.

- When SM701 is OFF: Comment is 32 characters
- When SM701 is ON : Comment is the upper 16 characters

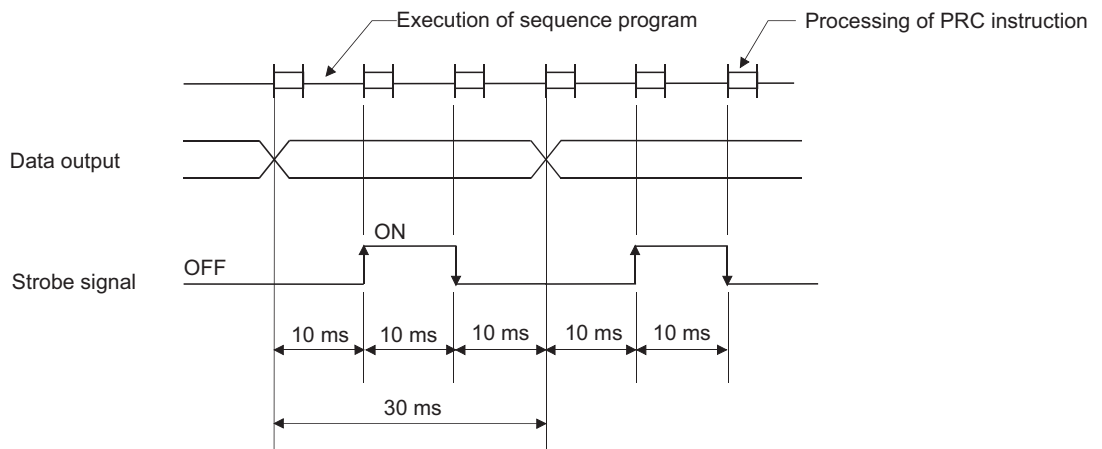
The number of points used by the output module is 10 points from the Y address designated by ⓓ.



[Timing Chart]



- (2) Output signals from the output module are transmitted at the rate of 30 ms per character. For this reason, the time required to the completion of the transmission of the designated number of characters will be  $30 \text{ ms} \times n$  (ms). At 10ms interrupt intervals, the PRC instruction executes data output, strobe signal ON, and strobe signal OFF. The other instructions are executed continuously during a period between the above processings.



- (3) In addition to the ASCII code, the output module also outputs a strobe signal (10 ms ON, 20 ms OFF) from the  $\text{Ⓔ} + 8$  device.
- (4) Following the execution of the PRC instruction, the PRC instruction execution flag ( $\text{Ⓔ} + 9$  device) remains ON until the completion of the transmission of the designated number of characters.
- (5) The PRC instruction can be used multiple times, but it is preferable to establish an interlock with the PRC instruction execution flag ( $\text{Ⓔ} + 9$  device) so that they will not be ON simultaneously.
- (6) If no comments have been registered at the device designated by  $\text{Ⓔ}$ , processing will not be performed.
- (7) When a comment is read, SM720 turns ON for one scan after the instruction is completed. SM721 turns ON during the execution of the instruction. The PRC instruction cannot be executed while SM721 is ON. If the attempt is made, no processing is performed.

**POINT**

1. For device comments used with the PRC instruction, use comment files stored in the memory card Standard Rom.  
Comment files stored in the program memory cannot be used.
2. The comment file used by the PRC instruction is set at the "PLC File Setting" option in the PLC parameter dialog box.  
If no comment file has been set for use by the PLC file setting, it will not be possible to output device comments with the PRC instruction.
3. Do not execute the PRC instruction during an interrupt program.  
Otherwise, malfunction may occur.

**! Operation Error**

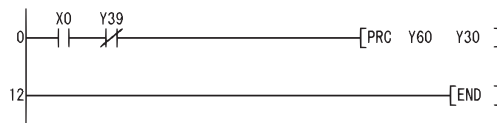
- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The PRC instruction is executed while a comment is written during RUN.

(Error code: 4100)

**Program Example**

- (1) Program which outputs the comment of Y60 to Y30 to Y39 when X0 is turned ON.

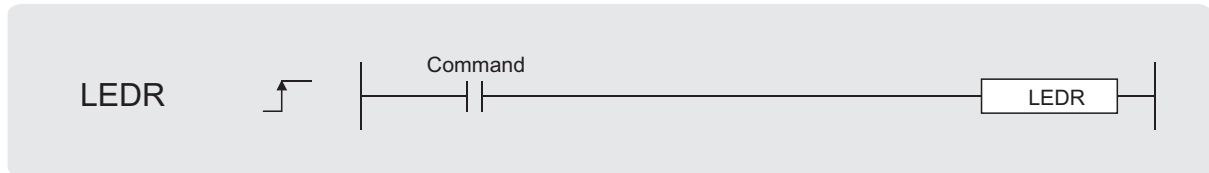
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ANI	Y39
2	PRC	Y60 Y30
5		<:y60="aa">
12	END	

## 7.9.3 Error display and annunciator reset instruction (LEDR)



Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

Resets the self-diagnosis error display so that annunciator display or operation can be continued. With one execution of this instruction, either error display or annunciator is reset.

#### (1) Operation when self-diagnosis error is generated

##### (a) If the self-diagnosis error is one which allows continued operation.

If the self-diagnosis error being displayed is one that will allow continued operation of the CPU module, the "ERROR/ERR." LED or error indication is reset. It will be necessary to reset SM0, SM1, and SD0 at the user program, because they are not reset automatically.

Since the cause of the error displayed at this time has a higher priority over annunciator, no action for resetting the annunciator is taken.

##### (b) When a battery error is generated.

If the LEDR instruction is executed after the battery has been replaced, the "BAT. ARM/ BAT." LED at the front of the CPU module and the error display will be reset. SM51 is also turned OFF at this time.

## (2) Operations when an annunciator (F) is ON.

## (a) When the CPU module has no LED display

The following operations will be conducted when the LEDR instruction is executed:

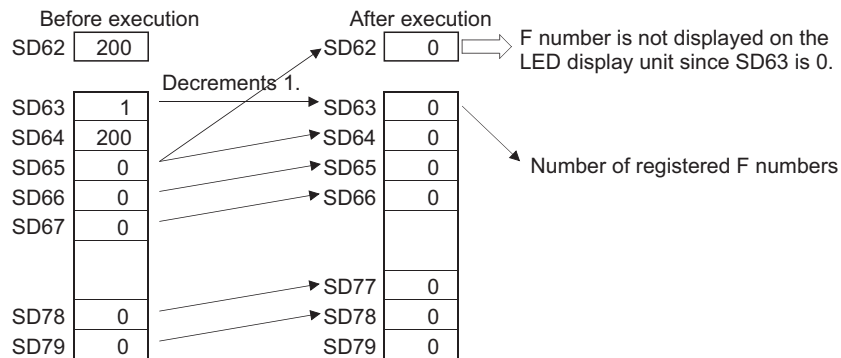
- 1) "USER" LED flickers, and is turned OFF
- 2) The annunciators (F) stored in SD62 and SD64 are reset, and the F numbers for SD65 to SD79 are moved up.
- 3) The data newly stored at SD64 is transmitted to SD62.
- 4) The data at SD63 is decremented by  $-1$ . However, if SD63 is 0, it remains 0.



## (b) For CPUs with an LED display at the front

The following operations will be conducted when the LEDR instruction is executed:

- 1) The F number being displayed at the front of the CPU module will be reset.
- 2) "USER" LED flickers, and is turned off.
- 3) The annunciators (F) stored in SD62 and SD64 are reset, and the F numbers for SD65 to SD79 are compressed forwards.
- 4) The data newly stored at SD64 is transmitted to SD62.
- 5) The data at SD63 is decremented by  $-1$ . However, if SD63 is 0, it remains 0.
- 6) The F number being stored at SD62 is displayed at the LED display. However, if the value of SD63 is 0, nothing will be displayed.



**Remark**

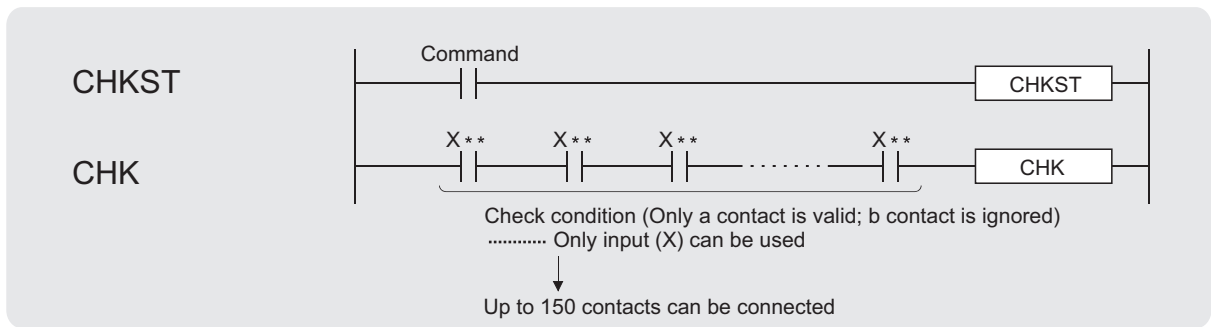
1. The defaults for the error item numbers set in special registers SD207 to SD209 and order of priority are given in the table below:

Priority	Factor number (Hexadecimal)	Meaning	Remarks
1	1	AC DOWN SINGLE PS.DOWN SINGLE PS.ERROR	Power supply cut Redundant base unit power supply voltage drop Redundant power supply module fault
2	2	UNIT VERIFY ERR. FUSE BREAK OFF SP. UNIT ERROR	I/O module verify error Blown fuse Special function module verify error
3	3	OPERATION ERROR LINK PARA.ERROR SFCP OPE. ERROR SFCP EXE. ERROR	[Operation Errors] Link parameter error SFC instruction operation error SFC program execution error
4	4	ICM.OPE ERROR FILE OPE. ERROR EXTEND INST. ERROR OPE. MODE DIFF. CAN'T EXE.MODE TRK.TRANS.ERR. TRK.SIZE ERROR TRK.DISCONNECT	Memory card operation error File access error Extend instruction error Operation status, switch mismatch Current mode-time function execution disabled Tracking data transmission error Tracking capacity excess error Tracking cable not connected, failure
5	5	PRG.TIME OVER	Constant scan setting time over error Low speed execution monitoring time over error
6	6	CHK instruction	—
7	7	Annunciators	—
8	8	LED instruction	—
9	9	BATTERY ERR.	—
10	A	Clock data	—
11	B	CAN'T SWITCH STANDBY SYS.DOWN MEM.COPY EXE.	System switching error Standby system not started/stop error Memory copy function executed

2. If the highest priority is given to the annunciator, it can be reset with priority by the LEDR instruction.

# 7.10 Debugging and failure diagnosis instructions

## 7.10.1 Special format failure checks (CHKST,CHK)



Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

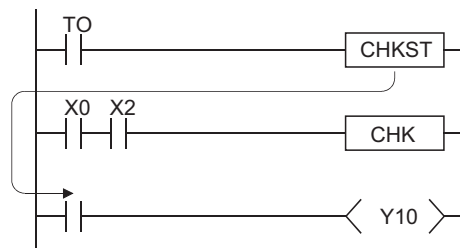
7

### ★ Function

#### CHKST

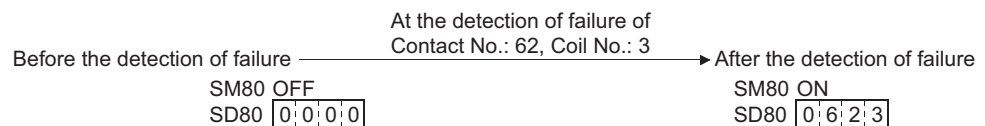
- (1) The CHKST instruction is the instruction that starts the CHK instruction.  
If the command for the CHKST instruction is OFF, execution jumps from the CHK instruction to the next instruction.  
If the command for the CHKST instruction is ON, the CHK instruction is executed.

When T0 is OFF, program jumps to the instruction next to the CHK instruction.



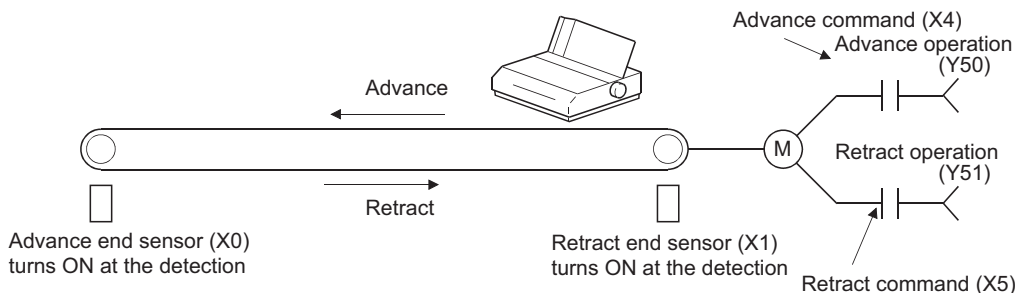
#### CHK

- (1) The CHK instruction is the instruction used for the bidirectional operation as shown on the following page to confirm the nature of the system failure.
  - (a) When the CHK instruction is executed, a failure diagnosis check is conducted with the designated check conditions, and if a failure is detected, SM80 is turned ON, and the failure number is stored at SD80 as a BCD value.  
The error code "9010" will be returned if a failure is detected.  
The contact number where the failure was discovered is stored at the upper 3 digits of SD80 (see (3)), and the coil number where the failure was detected (see (2)) is stored at the lower 1 digit of SD80.

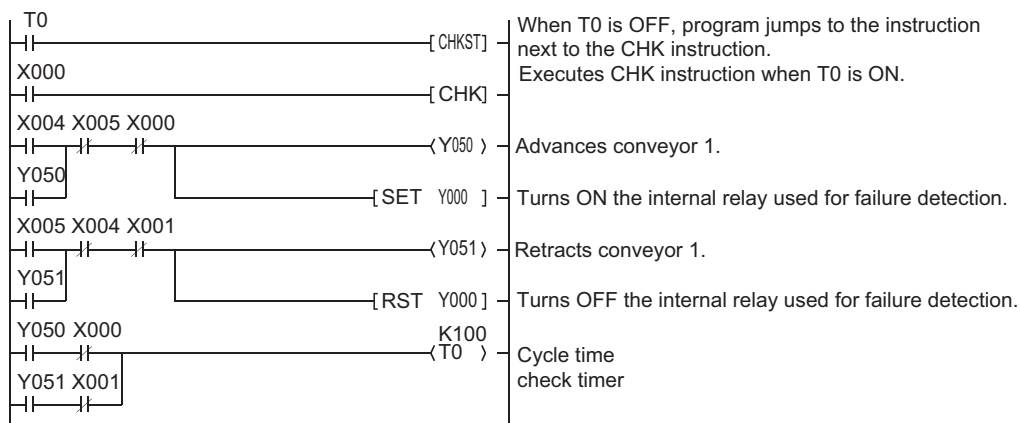


7.10 Debugging and failure diagnosis instructions  
7.10.1 Special format failure checks (CHKST,CHK)

- (b) The contact instruction prior to the CHK instruction does not control the execution of the CHK instruction, but rather sets the check conditions.



- (c) A ladder such as the one shown below can be created to perform a cycle time over check for the system shown above:



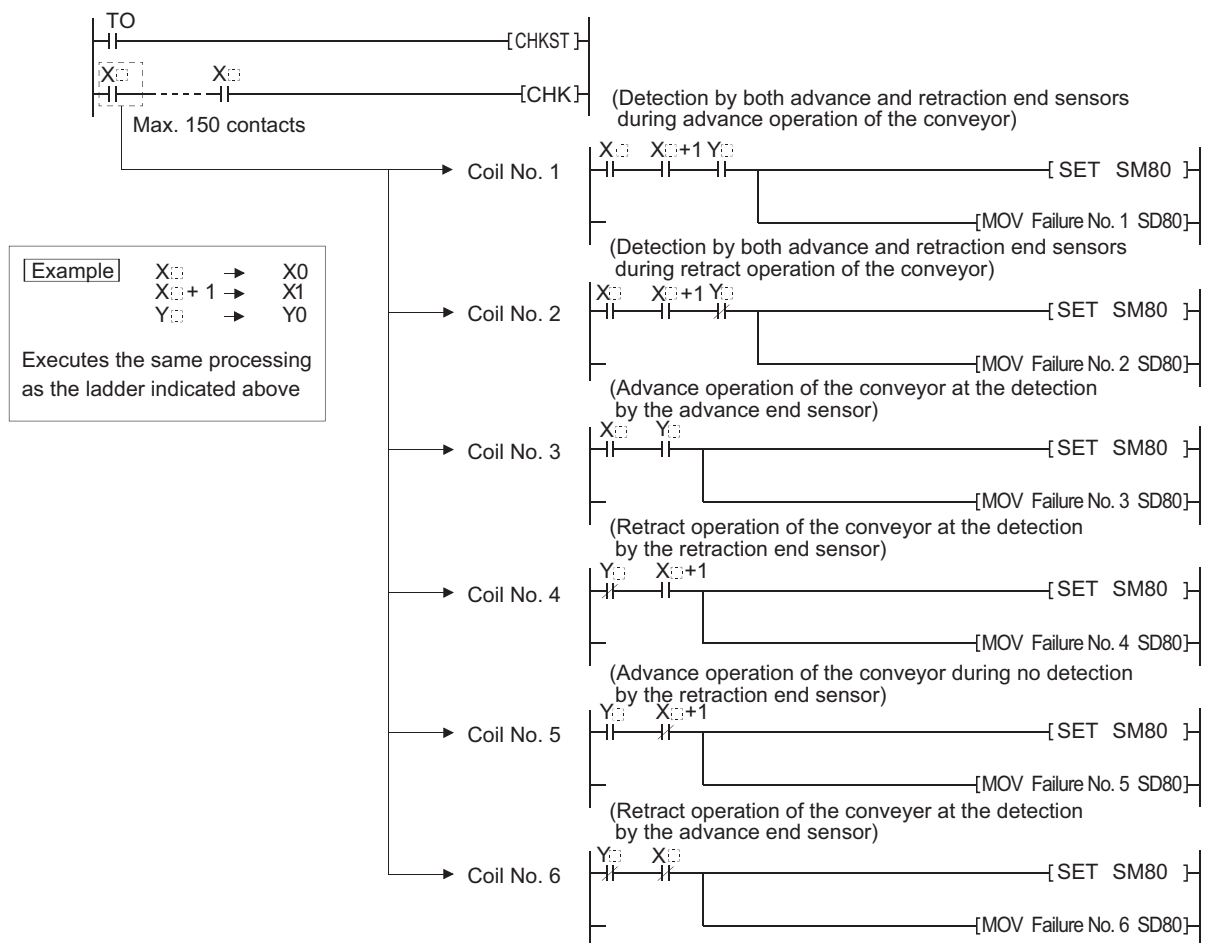
- (d) The following points should be taken into consideration when creating a ladder for use with the CHK instruction:

- 1) The contact numbers for the advance edge detection sensor and the retract edge detection sensor (X□) must always be continuous. Further, the contact number (X□) for the advance edge detection sensor should be lower than that for the retract edge.
- 2) Controls for the advance edge detection sensor contact number (X□) and output with the identical number (Y□) \*1 are as follows:  
 When advance operation is in progress .... turn ON  
 When retract operation is in progress ..... turn OFF

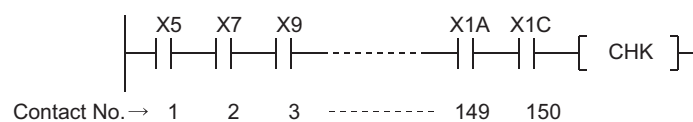
\*1: Output (Y□) is treated as an internal relay, and cannot be output to an external device.



- (2) Depending on the designated contact, the CHK instruction undergoes processing identical to that shown for the ladder below:

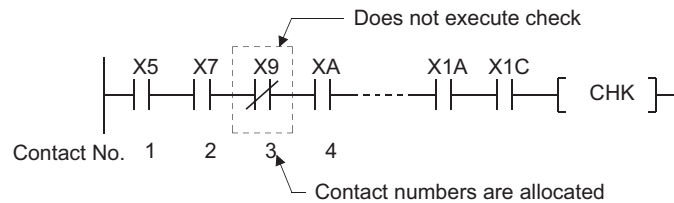


- (3) Numbers 1 to 150 from the vertical bus on the left side have been allocated as contact numbers during failure detection.



- (4) Reset SM80 and SD80 prior to forcing the execution of the CHK instruction.  
After the execution of the CHK instruction, it cannot be performed once again until SM80 and SD80 have been reset.  
(The contents of SM80 and SD80 will be preserved until reset by user.)
- (5) A CHKST instruction must be placed before the CHK instruction.  
An error will be returned if an instruction other than the LD, LDI, AND or ANI instruction is used between the CHK instruction and the CHKST instruction. (Error code: 4235)
- (6) The CHK instruction can be written at any step of the program.  
However, there is a limit in the number of uses of the CHK instruction.
- Can be used up to two places in all program files being executed.
  - Can be used only one place in a single program file.
- An error will be returned if the CHK instruction is used exceeding the number of uses specified above. (Error code: 4235)

- (7) Place LD and AND instructions prior to the CHK instruction to establish a check condition. Check conditions cannot be set using other contact instructions. If a check condition has been set with LDI or ANI, the processing for the check condition they specify will not be conducted. However, contact numbers during failure detection can also be allocated to the LDI and ANI instructions.



- (8) The failure detection method differs according to whether SM710 is ON or OFF.
- (a) If SM710 is OFF, checks will be conducted of coil numbers 1 to 6 for each contact successively. When the CHK instruction is executed, checks will be in order from coil No. 1 of contact No. 1, through coil No. 6, then move on to contact No. 2 and check the coils in order from No. 1. The CHK instruction will be completed when coil No. 6 from contact No. n has been checked.
- (b) If SM710 is ON, checks will be conducted of contact numbers 1 through n, in coil number order. When the CHK instruction is executed, checks will begin with the ladder for coil No. 1, in order from contact No. 1 until contact No. n, then move on to the coil No. 2 ladder and begin from contact No. 1. The CHK instruction will be completed when a check has been made through contact No. n of coil No. 6.
- (9) If more than one failure is detected, the number of the first failure detected will be stored. Failure numbers detected after this will be ignored.
- (10) The CHK instruction cannot be used by a low speed execution type program. If a low speed execution type program has been set in a program file containing the CHK instruction, an operation error will be returned, and the CPU module operation will be suspended.



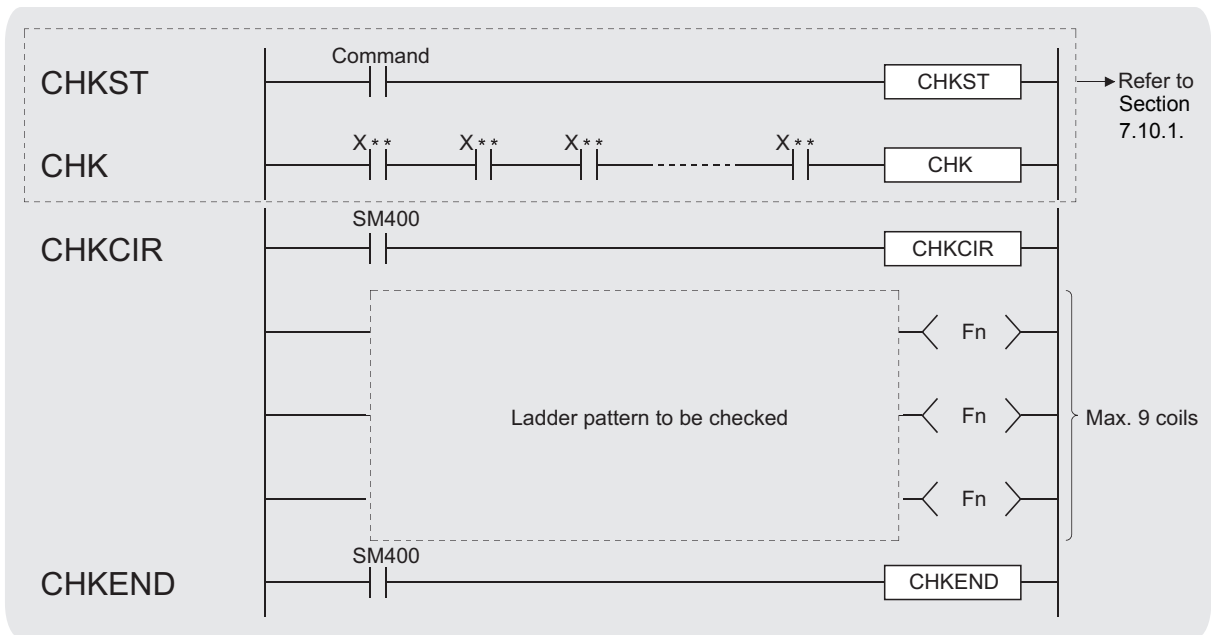
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- There is a parallel ladder. (Error code: 4235)
  - There is an NOP instruction. (Error code: 4235)
  - There are more than 150 contact instructions. (Error code: 4235)
  - A CHK instruction is not executed following the CHKST instruction. (Error code: 4235)
  - The CHK instruction is executed when no CHKST instruction has been executed. (Error code: 4235)
  - The CHKST and CHK instruction are used in a low speed execution type program. (Error code: 4235)
  - There is an instruction other than the LD, LDI, AND or ANI instruction between the CHK instruction and the CHKST instruction. (Error code: 4235)
  - The CHK instruction is used at three places or more in all of programs being executed. (Error code: 4235)
  - The CHK instruction is used at two places or more in a single program. (Error code: 4235)

## 7.10.2 Changing check format of CHK instruction (CHKCIR,CHKEND)



1 When the GX Developer is used (High Performance model QCPU/Process CPU/Redundant CPU)



Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

#### CHKCIR, CHKEND

- (1) The check ladder pattern that will be used in the CHK instruction can be updated to any format desired.  
The actual failure checks are conducted with the CHKST and CHK instructions.
- (2) Failure checks are conducted according to the check conditions designated by the CHK instruction and the ladder pattern described between the CHKCIR and CHKEND instructions.

#### Remark

Refer to 7.10.1 for more information on the CHKST and CHK instructions.

#### ☒ POINT

To change the check format of the CHK instruction using the CHKCIR to CHKEND instructions, the user should create a ladder with index modification (Z0).

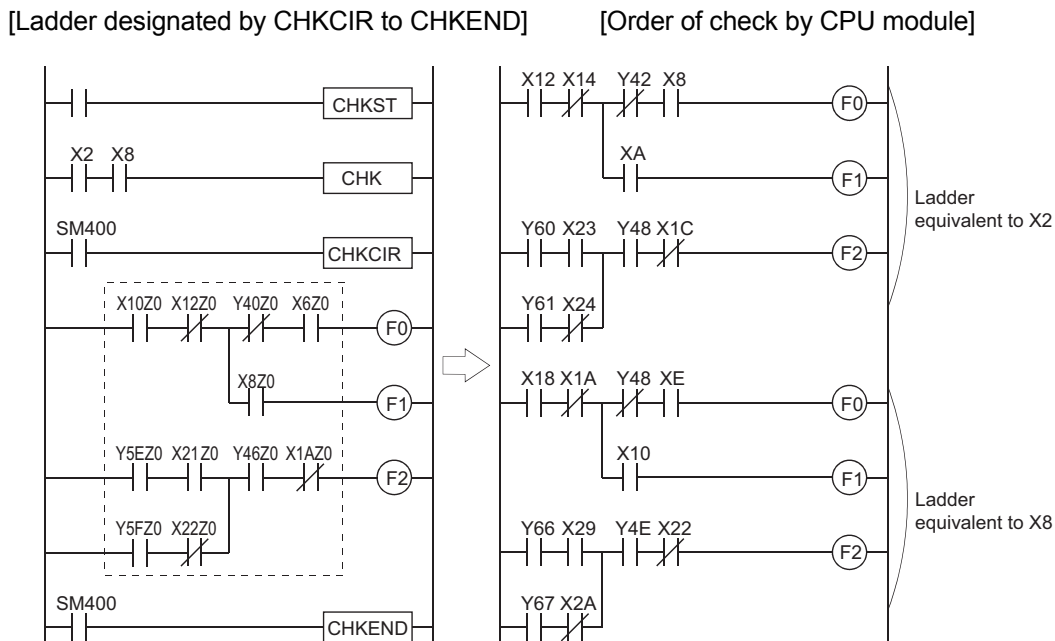
- (a) The device numbers indicated at check conditions (X2 and X8 in the figure below) will assume index modification values for the individual device numbers (with the exception of annunciators (F)) described in the ladder patterns.

**Example** X10 in the □ in the figure below would be as follows:

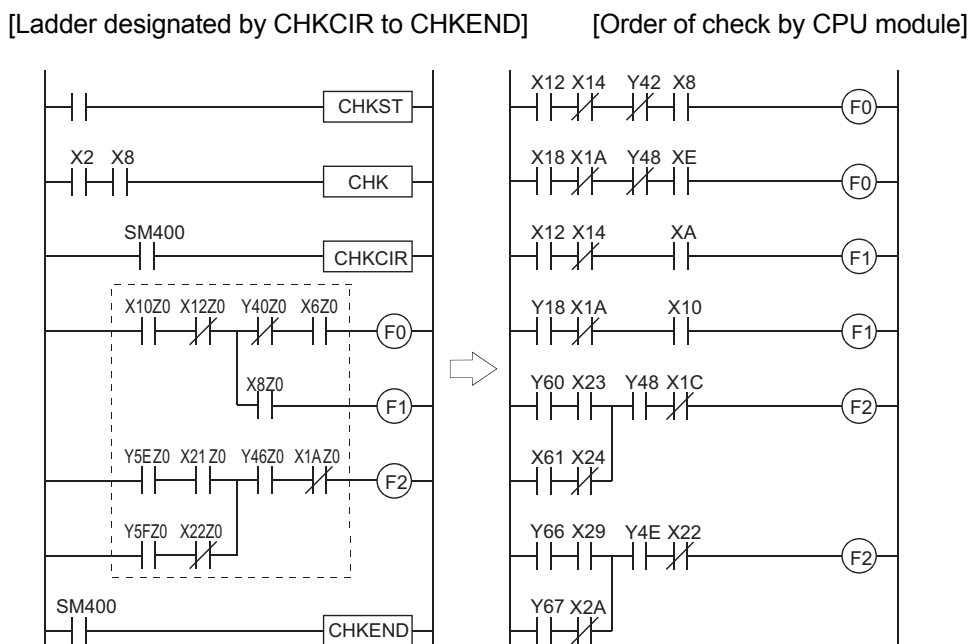
When corresponding to check condition X2 Processing performed by...X12  
 When corresponding to check condition X8 Processing performed by...X18

However, the order in which failure detection is executed differs depending on whether SM710 is ON or OFF.

- 1) If SM710 is OFF, checks will be conducted of coil numbers 1 through the end for each contact successively.



- 2) If SM710 is ON, checks will be conducted of contact numbers 1 through the end, in coil number order.

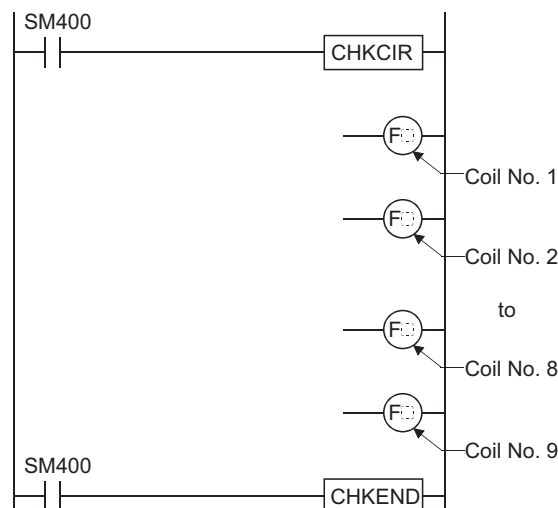


- (b) Failure checks check the ON/OFF status of OUT F $\square$  by using the ladder pattern in the various check conditions.

In all check conditions, SM80 will be turned ON if even one of the OUT F $\square$  is ON in a ladder pattern.

Further, the error numbers (contact numbers and coil numbers) corresponding to the OUT F $\square$  which were found to be ON will be stored from SD80 in BCD order.

- (c) The instructions that can be used in ladder patterns are as follows:  
 Contacts ... LD, LDI, AND, ANI, OR, ORI, ANB, ORB, MPS, MPP,MRD, and comparative operation instructions  
 Coil ..... OUT F $\square$
- (d) The following devices can be used for ladder pattern contacts:  
 Input (X), Output (Y)
- (e) Only annunciators (F) can be used in ladder pattern coils.  
 However, since annunciators (F) are used as a dummy, any value can be set for an annunciator (F).  
 Further, they can overlap with no difficulties.
- (f) ON/OFF controls can be performed without error if an annunciator (F) used during the execution of the CHK instruction has the same number as an annunciator (F) used in some other context than the CHK instruction. They will be treated differently during the CHK instruction than they are in the different context.
- (g) Since the annunciators (F) used in the CHK instruction do not turn ON/OFF actually, they will not turn ON/OFF if monitored by a peripheral device.
- (h) A ladder pattern can be created up to 256 steps.  
 Further, OUT F $\square$  can use up to 9 coils.
- (3) Coil numbers for ladders designated with the CHKCIR through CHKEND instructions are allocated coil numbers from 1 to 9, from top to bottom.



- (4) The CHKCIR and CHKEND instructions can be written at any step in the program desired. It can be used in up to two locations in all program files being executed. However, the CHKCIR and CHKEND instructions cannot be used in more than 1 location in a single program file.
- (5) The CHKCIR and CHKEND instructions cannot be used in low speed execution type programs. If a program file in which the CHKCIR or CHKEND instruction is described is set as a low speed execution type program, an operation error will occur, and the High Performance model QCPU/Process CPU/Redundant CPU operation will be suspended.



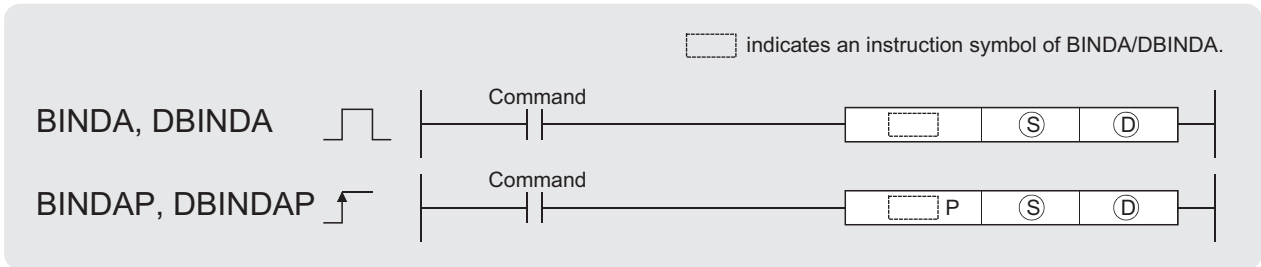
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The CHKCIR or CHKEND instruction appears three or more times in all program files. (Error code: 4235)
  - The CHKCIR or CHKEND instruction appears two or more times in a single program file. (Error code: 4235)
  - The CHKEND instruction is not executed following the execution of the CHKCIR instruction. (Error code: 4230)
  - The CHKEND instruction is executed although no CHKCIR instruction has been executed. (Error code: 4230)
  - The CHKST and CHK instruction are used in a low speed execution type program. (Error code: 4235)
  - There are 10 or more F instances in a ladder pattern. (Error code: 4235)
  - A ladder pattern has 257 or more steps. (Error code: 4235)
  - A device has been encountered which cannot be used in a ladder pattern. (Error code: 4235)
  - Index modification has been conducted on a ladder pattern device. (Error code: 4235)

# 7.11 Character string processing instructions

## 7.11.1 Conversion from BIN 16-bit or 32-bit to decimal ASCII (BINDA(P),DBINDA(P))

Basic
High performance
Process
Redundant
Universal



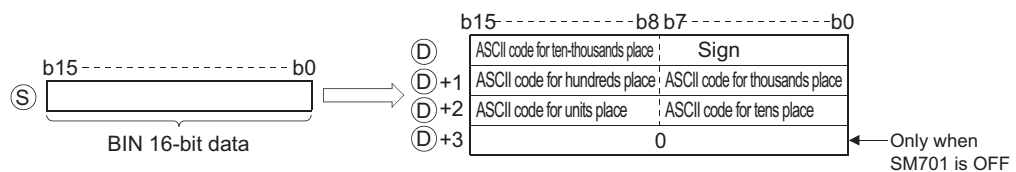
- Ⓢ : BIN data to be converted to ASCII (BIN 16/32 bits)
- Ⓣ : Head number of the devices where the conversion result will be stored (character string)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○			—
Ⓣ	—	○				—			—

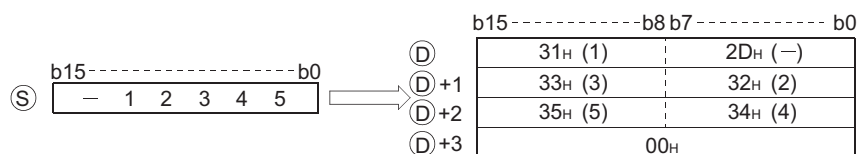
### ★ Function

#### BINDA

- Converts the individual digit numbers of decimal notation of the BIN 16-bit data designated by Ⓢ into ASCII codes, and stores the results into the area starting from the device designated by Ⓣ.



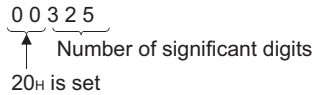
For example, if -12345 has been designated at Ⓢ, the following will be stored from Ⓣ onward:



- The BIN data designated at Ⓢ can be in the range from -32768 to 32767.

(3) The operation results stored at  $\textcircled{D}$  are as follows:

- (a) The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative.
- (b) The sign "20H" will be stored for the leading zeros of effective digits. (Zero suppression is conducted.)

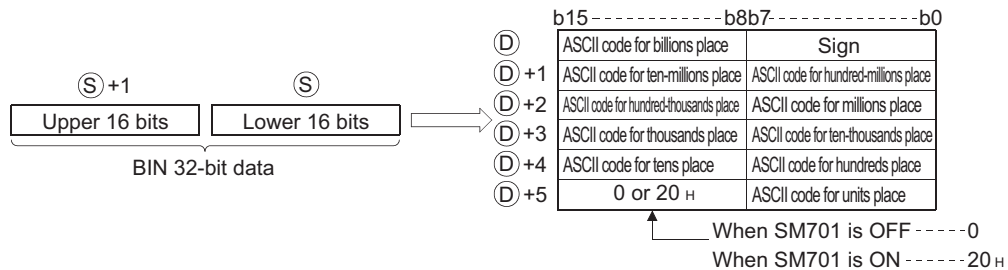


(c) The storage of data at devices specified by  $\textcircled{D} + 3$  differs depending on the ON/OFF status of SM701 (output number of characters conversion signal).

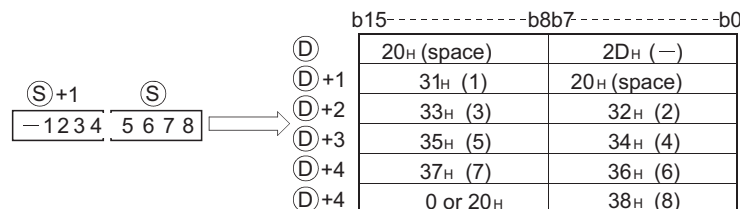
When SM701 is OFF .....Stores "0"  
 When SM701 is ON .....Does not change

**DBINDA**

(1) Converts the individual digit numbers of decimal notation of the BIN 32-bit data designated by  $\textcircled{S}$  into ASCII codes, and stores the results into the area starting from the device designated by  $\textcircled{D}$ .



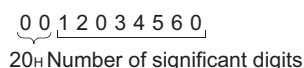
For example, if the value -12345678 has been designated by  $\textcircled{S}$ , the following would be stored into the area starting from  $\textcircled{D}$ :



(2) BIN data designated by  $\textcircled{S}$  can be between -2147483648 to 2147483647.

(3) The operations results stored at  $\textcircled{D}$  will be stored in the following way:

- (a) The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative.
- (b) The sign "20H" will be stored for the leading zeros of effective digits. (Zero suppression is conducted.)



(c) The data stored at the upper 8 bits of the device designated by  $\textcircled{D} + 5$  differs depending on the ON/OFF status of SM701 (number of characters to output select signal).

When SM701 is OFF .....Stores "0"  
 When SM701 is ON .....Stores "20H"



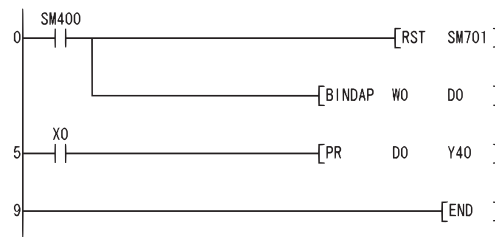
# ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by Ⓣ exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)

# Program Example

- (1) The following example program uses the PR instruction to output the 16-bit BIN data W0 value by decimal to Y40 to Y48 as ASCII.

[Ladder Mode]

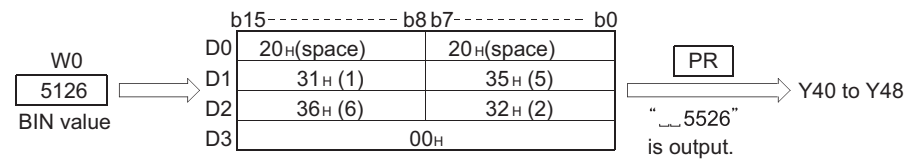


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	RST	SM701
2	BINDAP	W0 DO
5	LD	X0
6	PR	DO Y40
9	END	

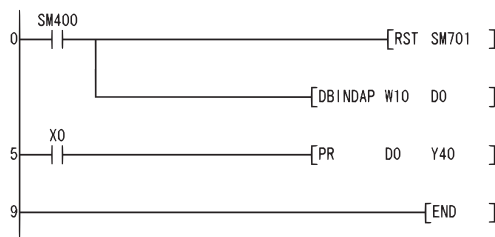
[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON. Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



- (2) The following program uses the PR instruction to output the decimal value of the 32-bit BIN data at W10 and W11 in ASCII code to Y40 to Y48.

[Ladder Mode]

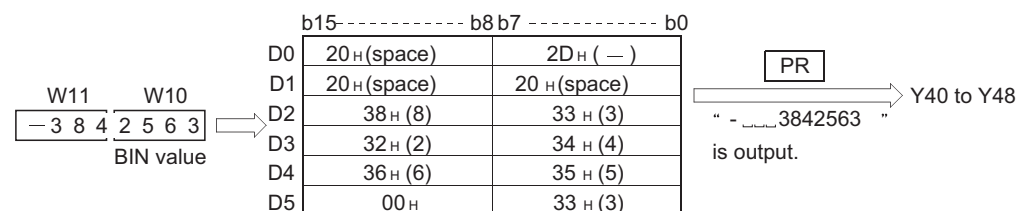


[List Mode]

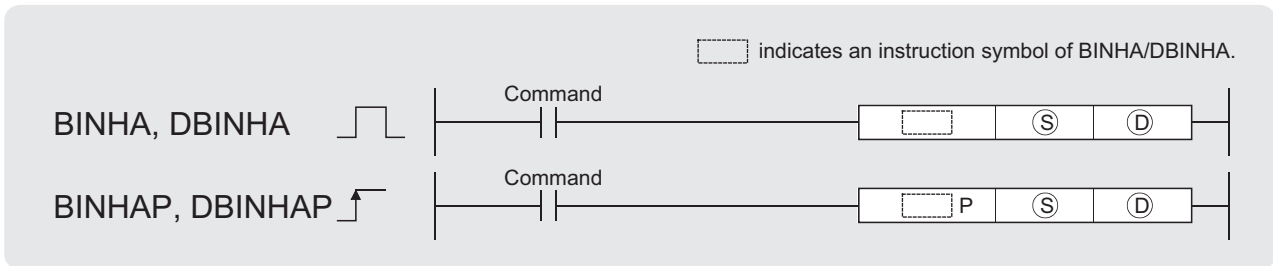
Step	Instruction	Device
0	LD	SM400
1	RST	SM701
2	DBINDAP	W10 DO
5	LD	X0
6	PR	DO Y40
9	END	

[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON. Because SM701 is OFF, the PR instruction will output ASCII code until 00H is encountered.



## 7.11.2 Conversion from BIN 16-bit or 32-bit data to hexadecimal ASCII (BINHA(P),DBINHA(P))



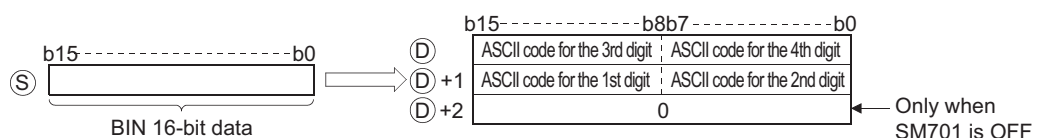
- Ⓢ : BIN data to be converted to ASCII (BIN 16/32 bits)
- Ⓣ : Head number of the devices where the conversion result will be stored (character string)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○			—
Ⓣ	—	○				—			—

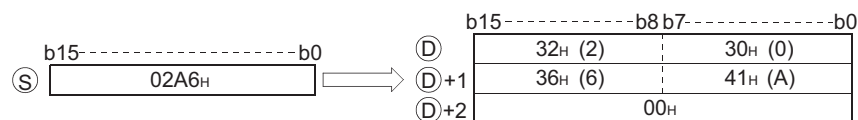
### ★ Function

#### BINHA

- (1) Converts the individual digit numbers of hexadecimal notation of the BIN 16-bit data designated by Ⓢ into ASCII codes, and stores the results into the area starting from the device designated by Ⓣ.



For example, if 02A6H has been designated by Ⓢ, it will be stored as follows:Ⓣ

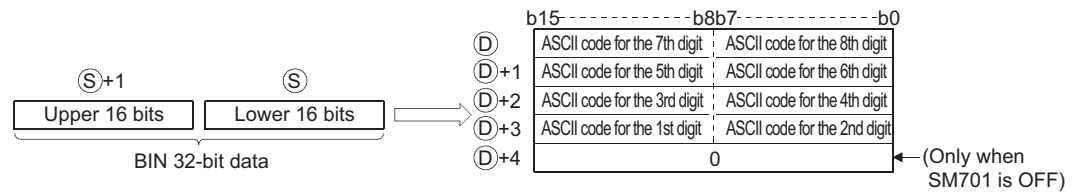


- (2) The BIN data designated by Ⓢ can be in the range from 0H to FFFFH.
- (3) The operation results stored at Ⓣ are processed as 4-digit hexadecimal values. For this reason, zeros which are significant digits on the left side of the value are processed as "0". (No zero suppression is conducted.)
- (4) The data to be stored at the device designated by Ⓣ +2 differs depending on the ON/OFF status of SM701 (number of characters to output select signal).

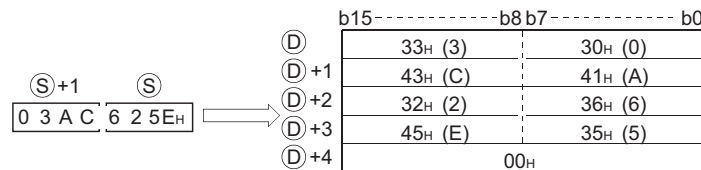
When SM701 is OFF .....Stores "0"  
 When SM701 is ON .....Does not change

## DBINHA

- (1) Converts the individual digit numbers of hexadecimal notation of the BIN 32-bit data designated by  $\textcircled{S}$  into ASCII codes, and stores the results into the area starting from the device designated by  $\textcircled{D}$ .



For example, if the value 03AC625EH has been designated by  $\textcircled{S}$ , it would be stored following  $\textcircled{D}$  in the following manner:



- (2) The BIN data designated by  $\textcircled{S}$  can be in the range from 0H to FFFFFFFFH.
- (3) The operation results stored at  $\textcircled{D}$  are processed as 8-digit hexadecimal values. For this reason, zeros which are significant digits on the left side of the value are processed as "0". (No zero suppression is conducted.)
- (4) The data to be stored at the device designated by  $\textcircled{D}+2$  differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
- When SM701 is OFF.....Stores "0"
- When SM701 is ON .....Does not change



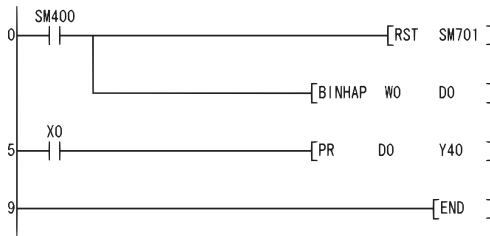
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device specified by  $\textcircled{D}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program uses the PR instruction to output the hexadecimal value of the 16-bit BIN data at W0 in ASCII code to Y40 to Y48.

[Ladder Mode]

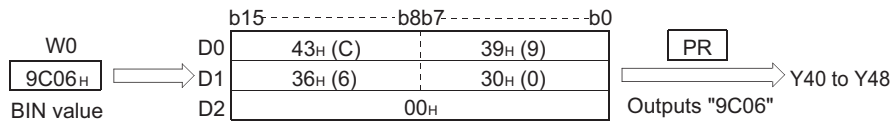


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	RST	SM701
2	BINHAP	W0 D0
5	LD	X0
6	PR	D0 Y40
9	END	

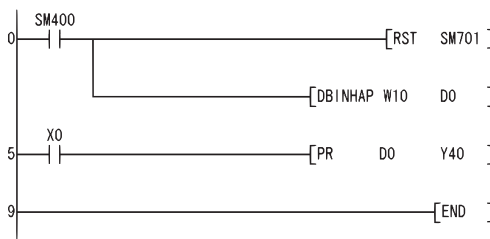
[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON. Because SM701 is OFF, The PR instruction will output ASCII code until 00H is encountered.



- (2) The following program uses the PR instruction to output the hexadecimal value of the 32-bit BIN data at W10 and W11 to Y40 to Y48.

[Ladder Mode]

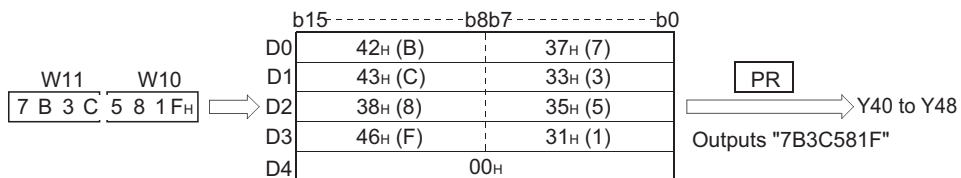


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	RST	SM701
2	DBINHAP	W10 D0
5	LD	X0
6	PR	D0 Y40
9	END	

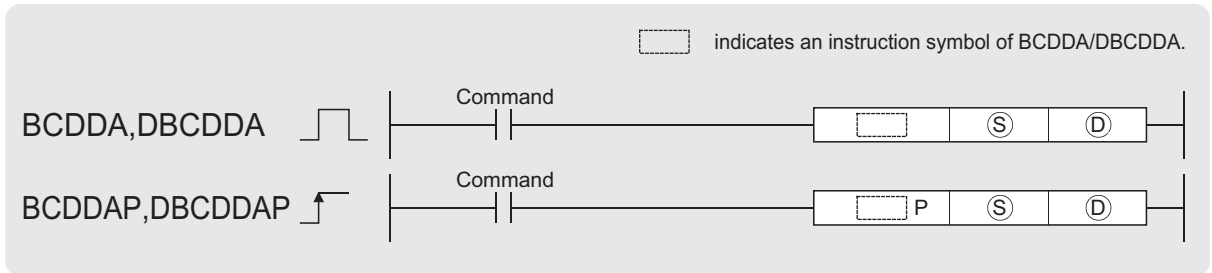
[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON. Because SM701 is OFF, The PR instruction will output ASCII code until 00H is encountered.



# 7.11.3 Conversion from BCD 4-digit and 8-digit to decimal ASCII data (BCDDA(P),DBCDDA(P))

Basic
High performance
Process
Redundant
Universal



(S) : BCD data to be converted to ASCII (BCD 4 digits/8 digits)  
 (D) : Head number of the devices where the conversion result will be stored (character string)

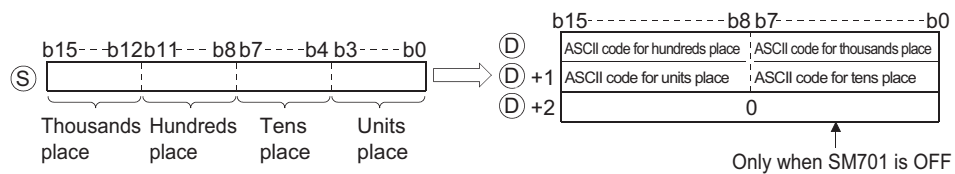
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
(S)	○	○				○			—
(D)	—	○				—			—

7

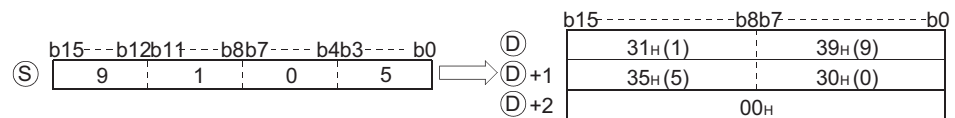
## ★ Function

### BCDDA

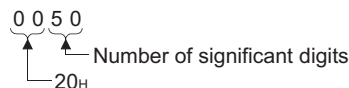
- Converts the individual digit numbers of hexadecimal notation of the BCD 4-digit data designated by (S) into ASCII codes, and stores the results into the area starting from the device designated by (D).



For example, when "9105" is designated for (S), the results of the operation are stored into the area starting from (D) in the following manner:



- The BCD data designated by (S) can be in the range of from 0 to 9999.
- The results of calculation stored in the device (D). All zeros on the left side of the "Number of significant digits" are zero-suppressed.

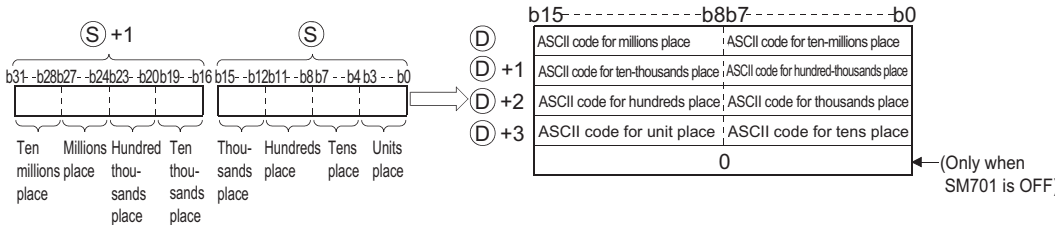


7.11 Character string processing instructions  
7.11.3 Conversion from BCD 4-digit and 8-digit to decimal ASCII data (BCDDA(P),DBCDDA(P))

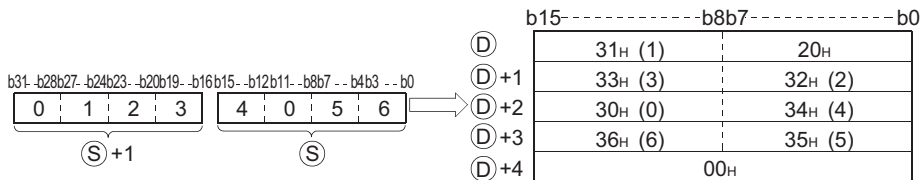
- (4) The data to be stored at the device designated by  $\textcircled{D} + 2$  differs depending on the ON/OFF status of SM701 (number of characters to output select signal).  
 When SM701 is OFF .....Stores "0"  
 When SM701 is ON .....Does not change

**DBCDDA**

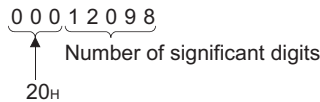
- (1) Converts the individual digit numbers of hexadecimal notation of the BCD 8-digit data designated by  $\textcircled{S}$  into ASCII codes, and stores the results into the area starting from the device designated by  $\textcircled{D}$ .



For example, if the value 01234056 is designated by  $\textcircled{S}$ , the operation result would be stored following  $\textcircled{D}$  in the following manner:



- (2) The BCD data designated by  $\textcircled{S}$  can be in the range of 0 to 99999999.  
 (3) The results of calculation stored in the device  $\textcircled{D}$ . All zeros on the left side of the "Number of significant digits" are zero-suppressed.



- (4) The data to be stored at the device designated by  $\textcircled{D} + 4$  differs depending on the ON/OFF status of SM701 (number of characters to output select signal).  
 When SM701 is OFF .....Stores "0"  
 When SM701 is ON .....Does not change

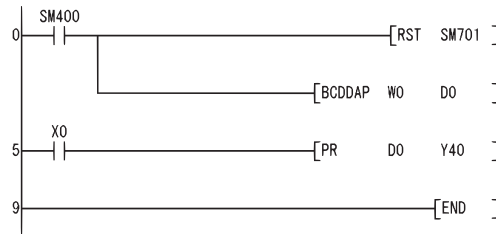
**Operation Error**

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data at  $\textcircled{S}$  during the operation of the BCDDA instruction is outside the range of from 0 to 9999. (Error code: 4100)
  - The data at  $\textcircled{S}$  during the operation of the DBCDDA instruction is outside the range of 0 to 99999999. (Error code: 4100)
  - The device specified by  $\textcircled{D}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program uses the PR instruction to convert BCD 4-digit data (the value at W0) to decimal, and outputs it in ASCII format to Y40 to Y48.

[Ladder Mode]

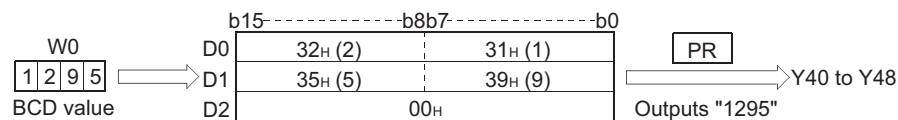


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	RST	SM701
2	BCDDAP	W0 D0
5	LD	X0
6	PR	D0 Y40
9	END	

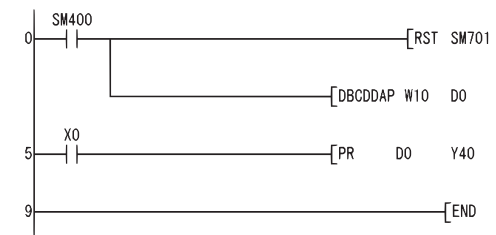
[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON. Because SM701 is OFF, The PR instruction will output ASCII code until 00H is encountered.



- (2) The following program uses the PR instruction to convert BCD 8-digit data (the values at W10 and W11) to decimal, and outputs it in ASCII format to Y40 to Y48.

[Ladder Mode]

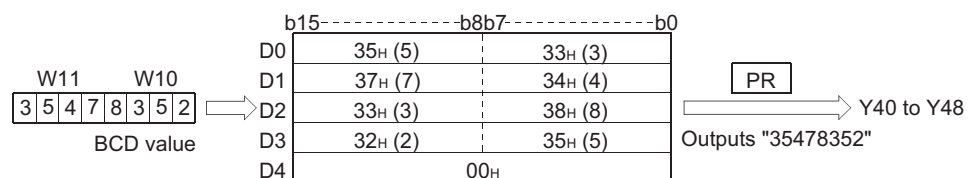


[List Mode]

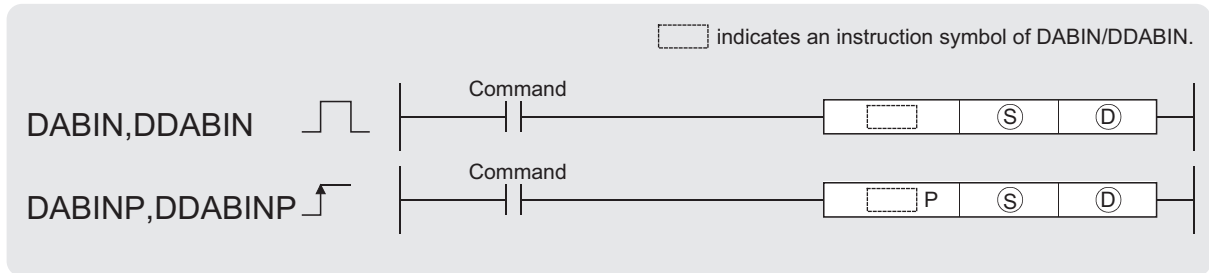
Step	Instruction	Device
0	LD	SM400
1	RST	SM701
2	DBCDDAP	W10 D0
5	LD	X0
6	PR	D0 Y40
9	END	

[Operation]

Conducts ASCII output of Y40 to Y48 by using the PR instruction when X0 goes ON. Because SM701 is OFF, The PR instruction will output ASCII code until 00H is encountered.



## 7.11.4 Conversion from decimal ASCII to BIN 16-bit and 32-bit data (DABIN(P),DDABIN(P))



Ⓢ : ASCII data to be converted to BIN value or head number of the devices where the ASCII data is stored (character string)

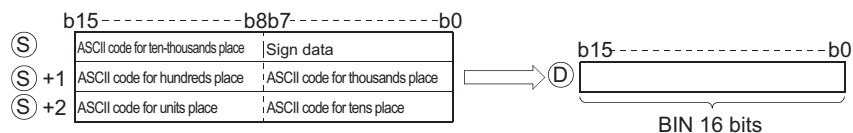
Ⓣ : Head number of the devices where the conversion result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	○	○				○		—	—

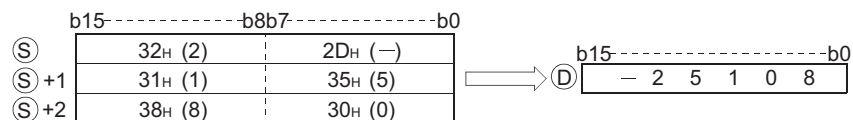
### ★ Function

#### DABIN

- Converts decimal ASCII data stored into the area starting from the device number designated by Ⓢ into BIN 16-bit data, and stores it in the device number designated by Ⓣ.



For example, if the ASCII code "– 25108H" is specified for the area starting from Ⓢ, the conversion result is stored at Ⓣ as shown below:

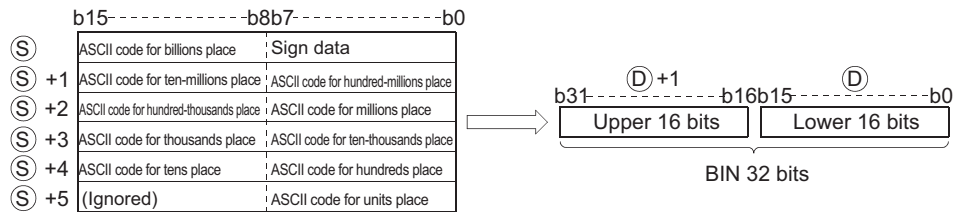


- The ASCII data designated by from Ⓢ to Ⓢ +2 can be in the range of from – 32768 to 32767
- The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative.  
(If other than "20H" and "2DH" is set, it will be processed as positive data.)
- ASCII code can be set for each position within the range from "30H" to "39H".
- If the ASCII code set for individual positions is "20H" or "00H," it will be processed as "30H".

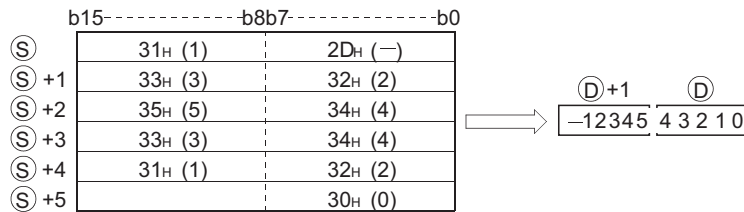


## DDABIN

- (1) Converts decimal ASCII data stored into the area starting from the device number designated by  $\textcircled{S}$  into BIN 32-bit data, and stores it in the device number designated by  $\textcircled{D}$ .



For example, if the ASCII code of -1234543210H is designated for the area starting from  $\textcircled{S}$ , the operation result would be stored at  $\textcircled{D} + 1$  and  $\textcircled{D}$  in the following manner:



- (2) The ASCII data designated by  $\textcircled{S}$  to  $\textcircled{S} + 5$  can be in the range of from -2147483648 to 2147483647.
- Further, data stored at the upper bytes of  $\textcircled{S} + 5$  will be ignored.
- (3) The sign "20H" will be stored if the BIN data is positive, and the sign "2DH" will be stored if it is negative.  
(If other than "20H" and "2DH" is set, it will be processed as positive data.)
- (4) ASCII code can be set for each position within the range from "30H" to "39H".
- (5) If the ASCII code set for individual positions is "20H" or "00H," it will be processed as "30H".

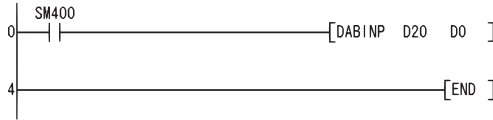
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The ASCII code designated by  $\textcircled{S}$  to  $\textcircled{S} + 5$  for the individual numbers is something other than "30H" to "39H", "20H", or "00H". (Error code: 4100)
  - The ASCII data designated by  $\textcircled{S}$  to  $\textcircled{S} + 5$  is outside the ranges shown below: (Error code: 4100)
    - When DABIN instruction is used ..... - 32768 to 32767
    - When DDABIN instruction is used..... - 2147483648 to 2147483647
  - The device specified by  $\textcircled{S}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program converts the decimal, 5-digit ASCII data and sign set at D20 through D22 to BIN values, and stores the result at D0.

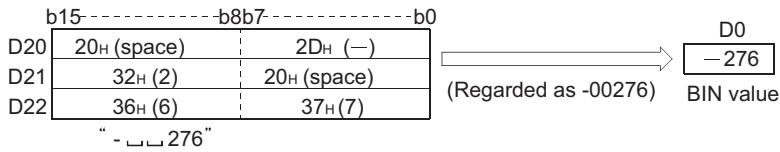
[Ladder Mode]



[List Mode]

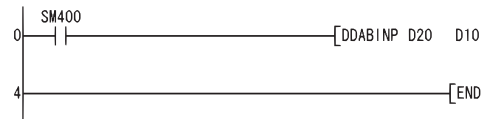
Step	Instruction	Device
0	LD	SM400
1	DABINP	D20 D0
4	END	

[Operation]



- (2) The following program converts the decimal, 10-digit ASCII data and sign set at D20 through D25 to BIN values and stores the result at D10 and D11.

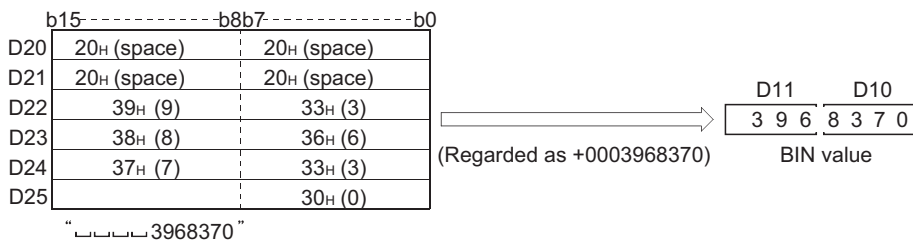
[Ladder Mode]



[List Mode]

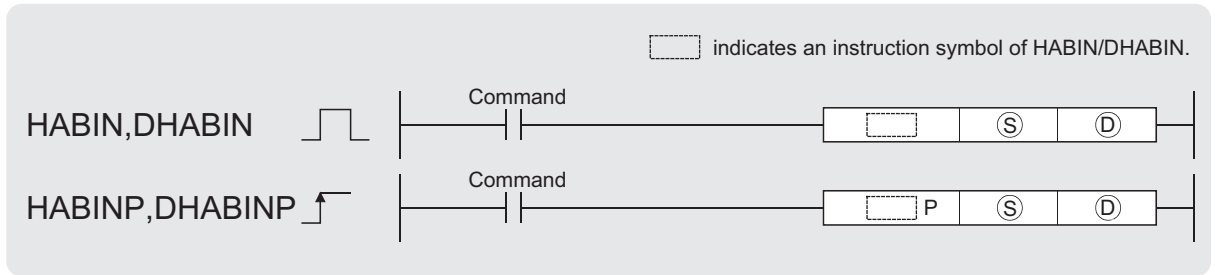
Step	Instruction	Device
0	LD	SM400
1	DDABINP	D20 D10
4	END	

[Operation]



# 7.11.5 Conversion from hexadecimal ASCII to BIN 16-bit and 32-bit data (HABIN(P),DHABIN(P))

Basic
High performance
Process
Redundant
Universal



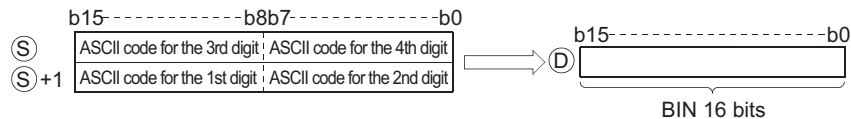
- Ⓢ : ASCII data to be converted to BIN value or head number of the devices where the ASCII data is stored (character string)
- Ⓣ : Head number of the devices where the conversion result will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	○	○				○		—	—

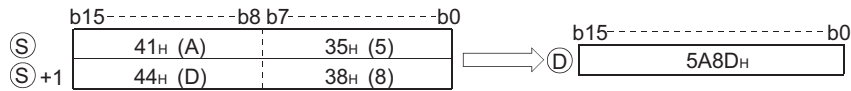
## ★ Function

### HABIN

- (1) Converts hexadecimal ASCII data stored in the area starting from the device number designated by Ⓢ into BIN 16-bit data, and stores it in the device number designated by Ⓣ.



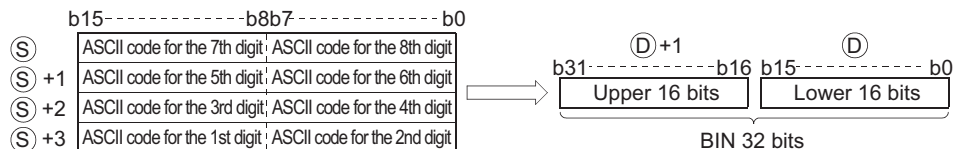
For example, if the ASCII code of 5A8DH is designated for the area starting from Ⓢ, the operation result would be stored at Ⓣ in the following manner:



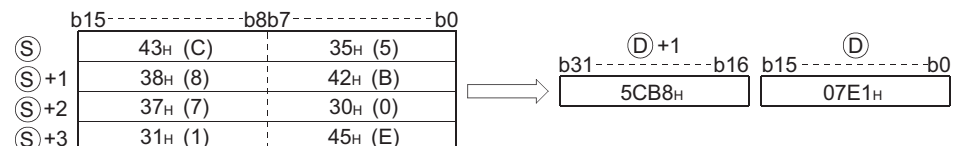
- (2) The ASCII data designated by Ⓢ to Ⓢ+1 can be in the range of from 0000H to FFFFH.
- (3) The ASCII codes can be in the range of "30H" to "39H" and from "41H" to "46H".

## DHABIN

- (1) Converts hexadecimal ASCII data stored in the area starting from the device number designated by  $\textcircled{S}$  into BIN 32-bit data, and stores it in the device number designated by  $\textcircled{D}$ .



For example, if the ASCII code of 5CB807E1H is designated for the area starting from  $\textcircled{S}$ , the operation result would be stored at  $\textcircled{D}+1$  and  $\textcircled{D}$  in the following manner:



- (2) The ASCII data designated by  $\textcircled{S}$  to  $\textcircled{S}+3$  can be in the range of from 00000000H to FFFFFFFFH.
- (3) The ASCII codes can be in the range of "30H" to "39H" and from "41H" to "46H".



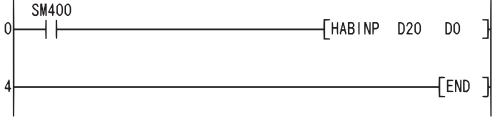
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The ASCII codes for the individual numbers designated by  $\textcircled{S}$  to  $\textcircled{S}+3$  are outside the range of from "30H" to "39H" and from "41H" to "46H". (Error code: 4100)
  - The device specified by  $\textcircled{S}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

# Program Example

- (1) The following program converts the hexadecimal, 4-digit ASCII data set at D20 and D21 to BIN data, and stores the result at D0.

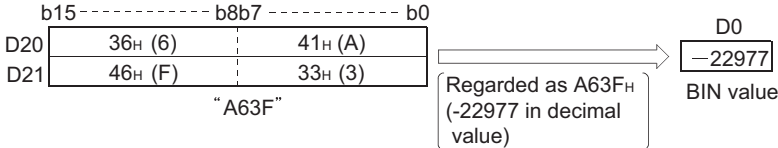
[Ladder Mode]



[List Mode]

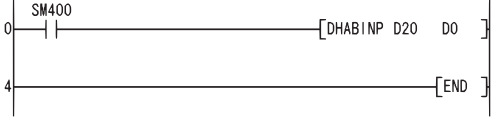
Step	Instruction	Device
0	LD	SM400
1	HABINP	D20 D0
4	END	

[Operation]



- (2) The following program converts the hexadecimal, 8-digit ASCII data set at D20 to D23 to BIN values, and stores the result at D10 and D11.

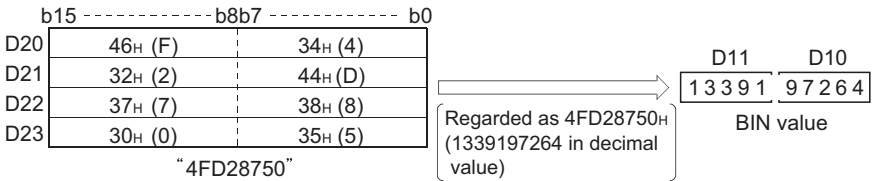
[Ladder Mode]



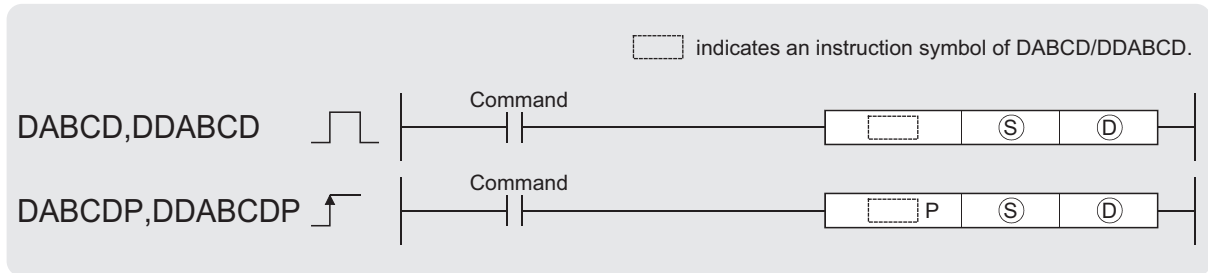
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DHABINP	D20 D0
4	END	

[Operation]



## 7.11.6 Conversion from decimal ASCII to BCD 4-digit or 8-digit data (DABCD(P),DDABCD(P))



Ⓢ : ASCII data to be converted to BCD value or head number of the devices where the ASCII data is stored (character string)

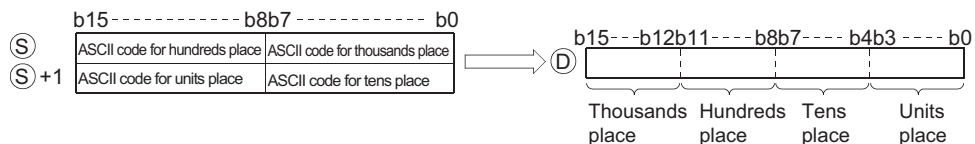
Ⓣ : Head number of the devices where the conversion result will be stored (BCD 4 digits/8 digits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	○	○				○		—	—

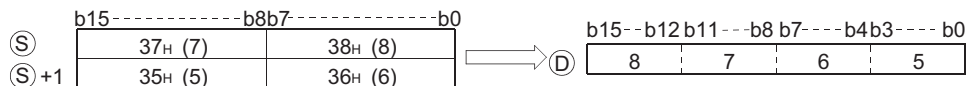
### ★ Function

#### DABCD

- Converts decimal ASCII data stored in the area starting from device number designated by Ⓢ into 4-digit BCD data, and stores at device number designated by Ⓣ.



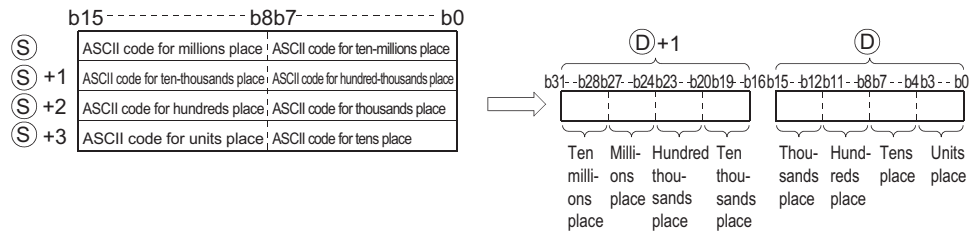
For example, if the ASCII code of 8765H is designated for the area starting from Ⓢ, the operation results would be stored at Ⓣ in the following manner:



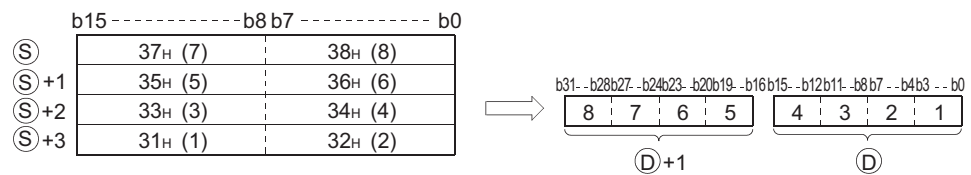
- The ASCII data designated by Ⓢ to Ⓢ+1 can be in the range of from 0 to 9999.
- The ASCII code set at each digit can be in the range of from "30H" to "39H".
- If ASCII code for individual digits is "20H" or "00H", it is processed as "30H".

## DDABCD

- (1) Converts decimal ASCII data stored in the area starting from the device designated by  $\textcircled{S}$  to 8-digit BCD data, and stores it into the area starting from the device designated by  $\textcircled{D}$ .



For example, if the ASCII code of 87654321H is designated for the area starting from  $\textcircled{S}$ , the operation results would be stored at  $\textcircled{D}+1$  and  $\textcircled{D}$  in the following manner:



- (2) The ASCII data designated at  $\textcircled{S}$  to  $\textcircled{S}+3$  can be in the range of from 0 to 99999999.
- (3) The ASCII code set at each digit can be in the range of from "30H" to "39H".
- (4) If ASCII code for individual digits is from "20H" to "00H", it is processed as "30H".



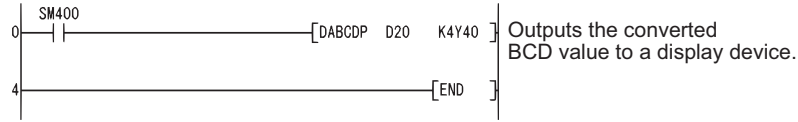
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - There are characters within the data at  $\textcircled{S}$  that are outside the 0 to 9 range. (Error code: 4100)
  - The device specified by  $\textcircled{S}$  exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program converts the decimal ASCII data set from D20 to D22 to BCD 4-digit data, and outputs the results to Y40 to Y4F.

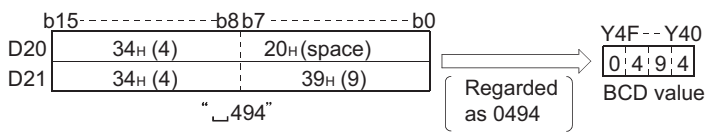
[Ladder Mode]



[List Mode]

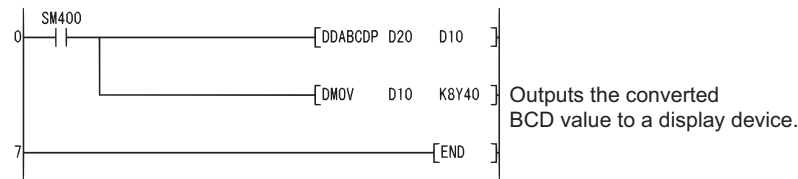
Step	Instruction	Device
0	LD	SM400
1	DABCDP	D20 K4Y40
4	END	

[Operation]



- (2) The following program converts the decimal ASCII data set at D20 to D23 into 8-digit BCD data, stores the result at D10 and D11, and also outputs it to from Y40 to Y5F.

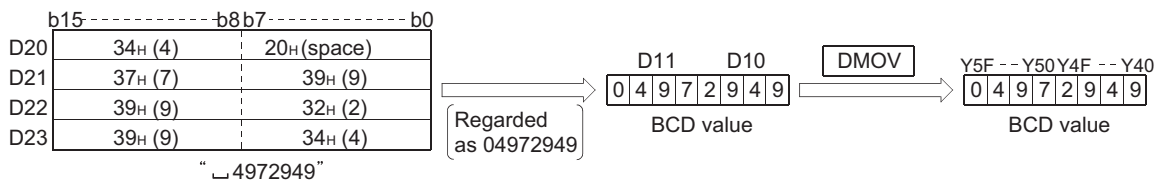
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DDABCDP	D20 D10
4	DMOV	D10 K8Y40
7	END	

[Operation]





## 7.11.7 Reading device comment data (COMRD(P))

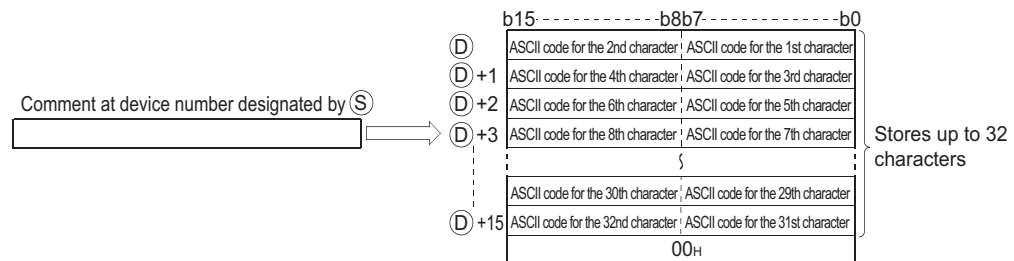


- Ⓢ : Head number of the devices where a comment to be read is stored (Device name)  
 Ⓣ : Head number of the devices where the read comment will be stored (character string)

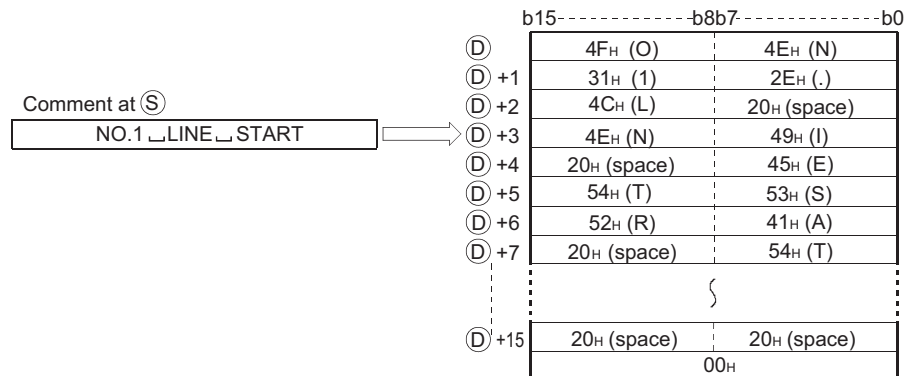
Setting Data	Internal Devices		R, ZR	J <small>DATA</small>		U <small>DATA</small> G <small>DATA</small>	Zn	Constants	Other BL,S,BL VTR,BL, P,I,J,U
	Bit	Word		Bit	Word				
Ⓢ	○	○			○		—		○
Ⓣ	—	○			—		—		—

### ★ Function

- (1) Reads the comment at the device number designated by Ⓢ, and stores it as ASCII code in the area starting from the device number designated by Ⓣ.



For example, if the comment for the device designated by Ⓢ were "NO. 1 LINE START," the operation results would be stored following Ⓣ as follows:



- (2) If no comment has been registered for the device specified by ⑤ despite the fact that the comment range setting is made, all of the characters for the comment are processed as "20H" (space).
- (3) The device number plus 1 where the final character of ⑥ is stored differs depending on the ON/OFF status of SM701 (number of characters to output select signal).
- |                   |                   |
|-------------------|-------------------|
| When SM701 is OFF | : Does not change |
| When SM701 is ON  | : Stores "0"      |
- (4) When a comment is read, SM720 turns ON for one scan after the instruction is completed. SM721 turns ON during the execution of the instruction. While SM721 is ON, the COMRD(P) instruction cannot be executed. If the attempt is made, no processing is performed.

### POINT

---

1. The device comment used in the COMRD(P) instruction uses a comment file stored in a memory card and the standard ROM. Comment files stored in the program memory cannot be used.
  2. Set the comment file used for the COMRD(P) instruction in "PLC file setting" in the PLC parameter dialog box. If the comment file to be used is not set in the PLC file setting, device comments cannot be output with the COMRD(P) instruction.
  3. The COMRD(P) instruction cannot be executed during the interrupt program. No operation if executed.
- 

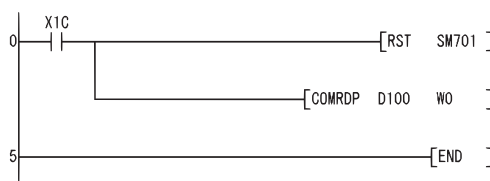
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The comment is not registered to the device number specified by ⑤. (Error code: 4100)
  - The device number specified by ⑥ is not a word device. (Error code: 4101)
  - The device specified by ⑥ exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)

## Program Example

- (1) The following program stores the comments set at D100 into the area starting from W0 as ASCII when X1C is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	RST	SM701
2	COMRDP	D100 WO
5	END	

[Operation]

	b15-----b8	b7-----b0
W0	49H (I)	4CH (L)
W1	45H (E)	4EH (N)
W2	41H (A)	20H (space)
W3	54H (T)	20H (space)
W4	52H (R)	41H (A)
W5	55H (E)	57H (G)
W6	20H (space)	54H (T)
W7	20H (space)	20H (space)
	}	
W15	20H (space)	20H (space)
W16	00H	

Comment at D100  
 LINE TARGET

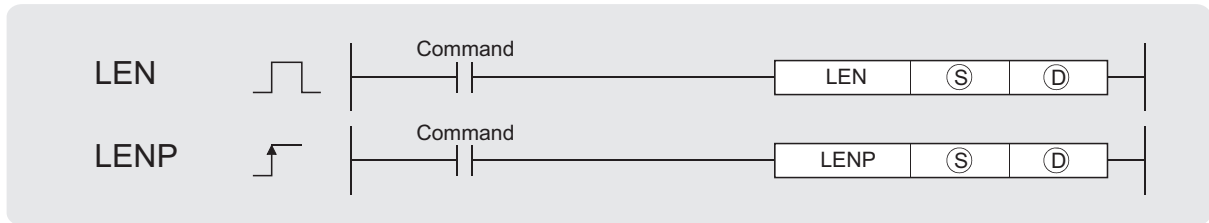
## Caution

- QCPU completes the processing after several scans.
- The COMRD(P)/PRC instruction is not executed if the start signal (execution command) of the COMRD(P)/PRC instruction is turned ON before completion of the instruction (while SM721 is ON). Execute the COMRD(P)/PRC instruction when SM721 is OFF.
- Two or more file comments cannot be accessed simultaneously.
- The following instructions cannot be executed simultaneously because they use SM721 in common.

Instruction Name	ON During Execution	ON for One Scan After Completion	ON after Abnormal Completion
SP. FREAD SP. FWRITE	SM721	Designated by instruction.	(Device designated by instruction) + 1
PRC COMRD		SM720	None

# 7.11.8 Character string length detection (LEN(P))

Basic
High performance
Process
Redundant
Universal



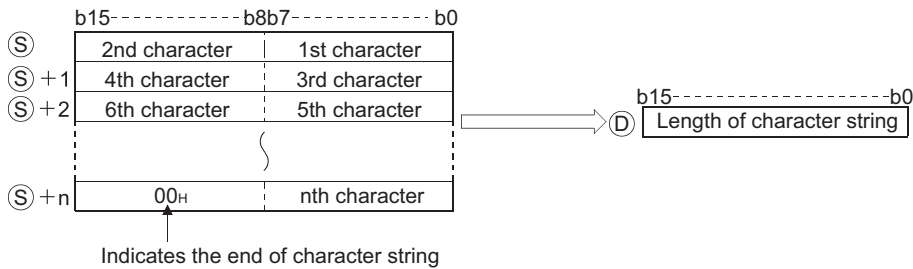
(S) : Character string or head number of the devices where the character string is stored (character string)  
 (D) : Number of the device where the length of detected character string will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JOG		UJOG	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
(S)	—	○				—		○	—
(D)	○	○				○		—	—

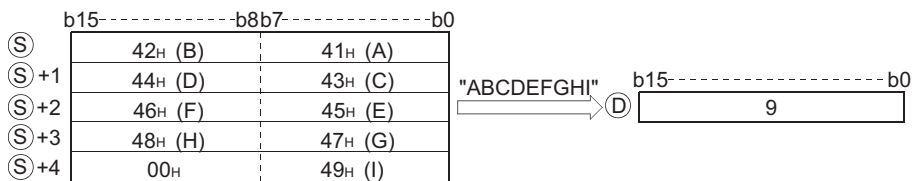
## ★ Function

- (1) Detects length of character string designated by (S) and stores in the area starting from the device number designated by (D).

Processes the data from the device number designated by (S) to the device number storing "00H" as a character string.



For example, when the value "ABCDEFGHI" is stored in the area starting from (S), the value 9 is stored at (D).



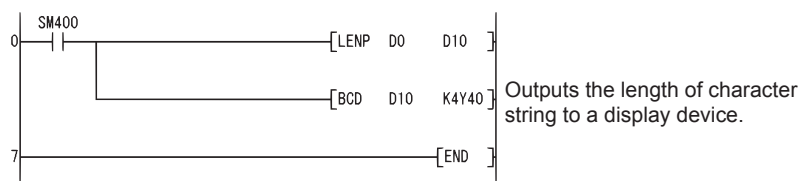
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- There is no "00H" set within the relevant device range following the device number designated by (S). (Error code: 4101)

## Program Example

- (1) The following program outputs the length of the character string from D0 to Y40 to Y4F as BCD 4-digit values.

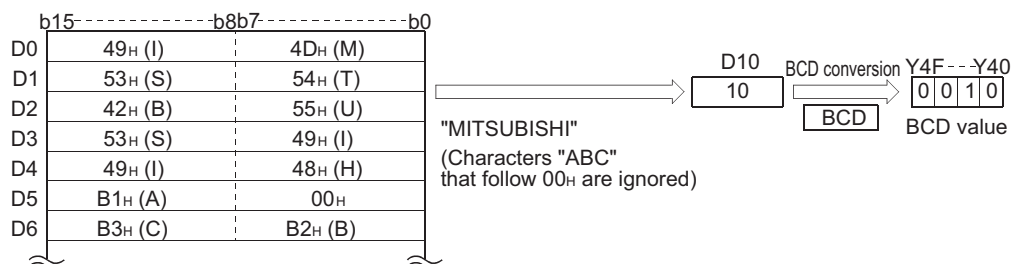
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	LENP	D0 D10
4	BCD	D10 K4Y40
7	END	

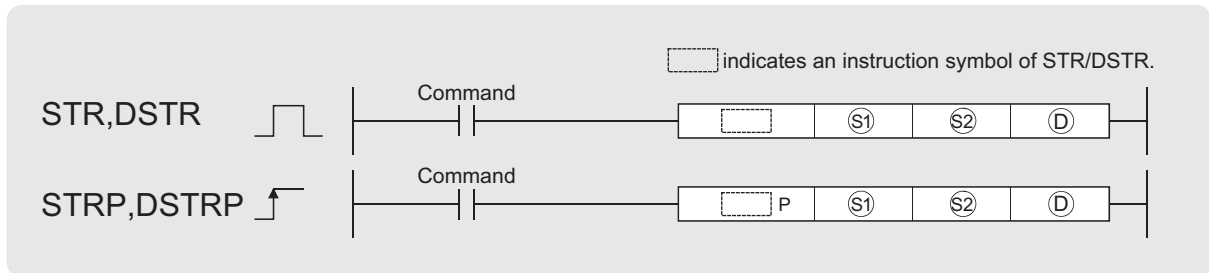
[Operation]



# 7.11.9 Conversion from BIN 16-bit or 32-bit to character string (STR(P),DSTR(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



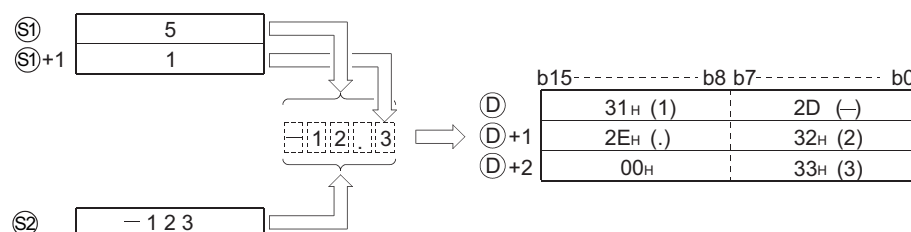
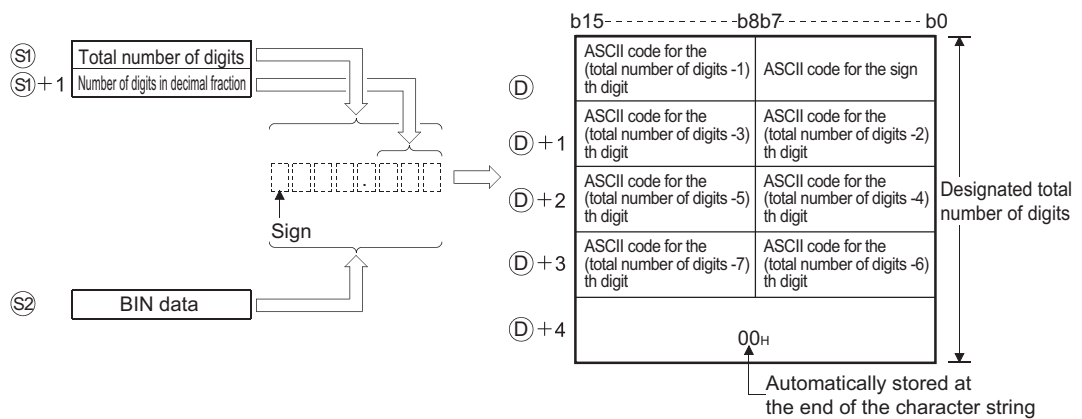
- Ⓢ1 : Head number of the devices where the digits numbers for the numerical value to be converted are stored (BIN 16 bits)
- Ⓢ2 : BIN data to be converted (BIN 16/32 bits)
- ⓈD : Head number of the devices where the converted character string will be stored (character string)

Setting Data	Internal Devices		R, ZR	J:DO		U:GO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1	○	○				○		—	—
Ⓢ2	○	○				○		○	—
ⓈD	—	○				—		—	—

## ★ Function

### STR

(1) Adds a decimal point to the BIN 16-bit data designated by Ⓢ2 at the location designated by Ⓢ1, converts the data to character string data, and stores it in the area starting from the device number designated by ⓈD.

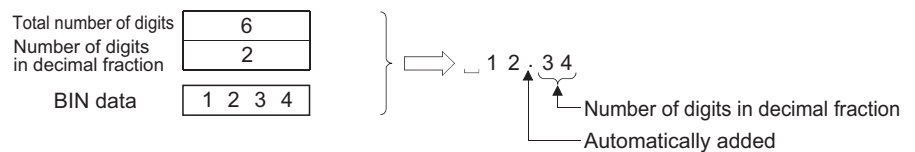


- (2) The total number of digits that can be designated by  $\text{S}_1$  is from 2 to 8.
- (3) The number of digits that can be designated by  $\text{S}_1 + 1$  as a part of the decimal fraction is from 0 to 5.
- However, the number of digits following the decimal point must be smaller than or equal to the total number of digits minus 3.

(4) BIN data in the range between  $-32768$  and  $32767$  can be designated at  $\text{S}_2$ .

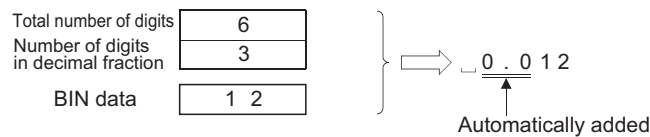
(5) After conversion, character string data is stored at the device number  $\text{D}$  or later device number as indicated below:

- (a) The sign "20H" (space) will be stored if the BIN data is positive, and the sign "2DH" (minus sign) will be stored if it is negative.
- (b) If the setting for the number of digits after the decimal fraction is anything other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits.

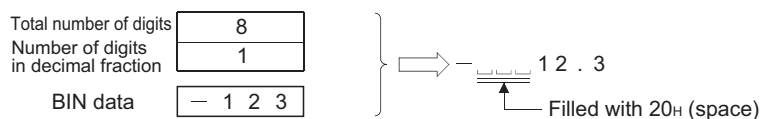


If the number of digits in the decimal fraction part of the number is "0", the ASCII code "2EH" (.) will not be stored.

- (c) If the total number of digits following the decimal fraction is greater than the number of BIN data digits, a zero will be added automatically and the number converted by shifting to the right, so that it would become "0.000000".



- (d) If the total number of digits excluding the sign and the decimal point is greater than the number of BIN data digits, "20H" (space) will be stored between the sign and the numeric value.

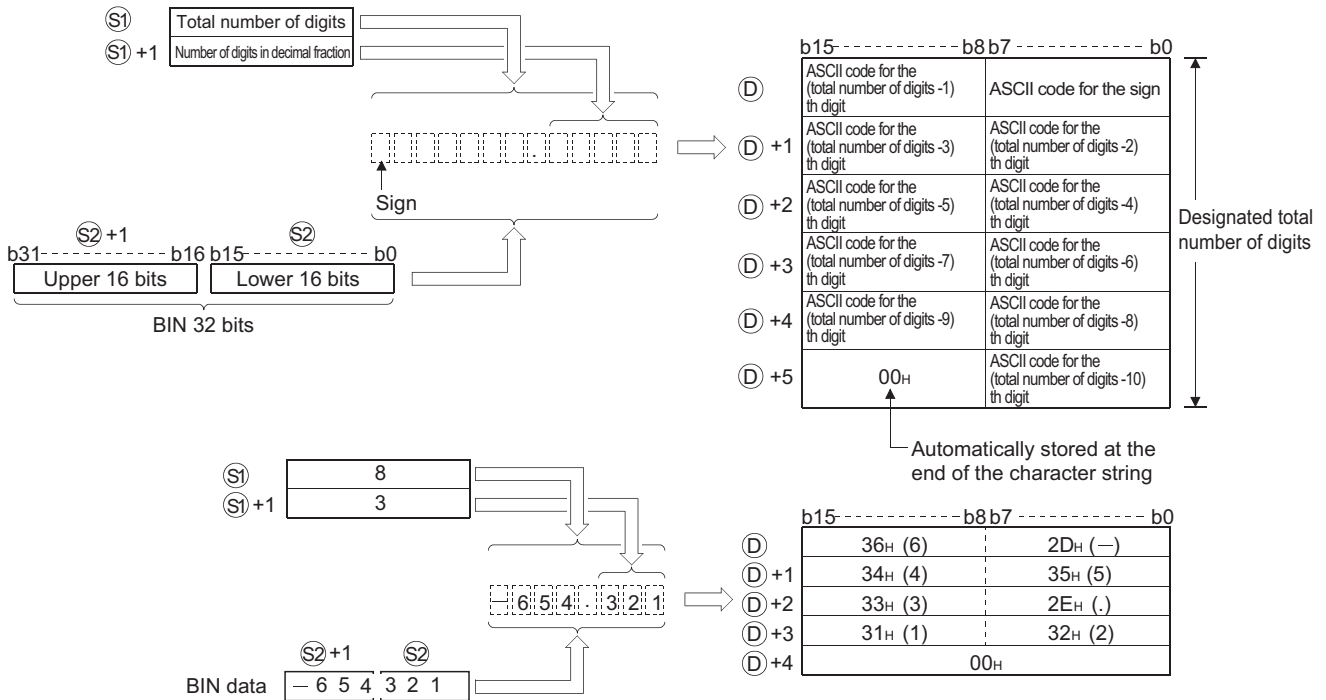


If the number of BIN digits is greater, an error will be returned.

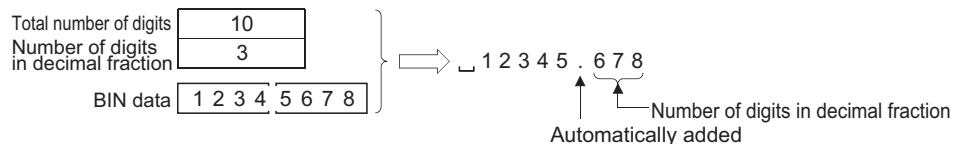
- (e) The value "00H" is automatically stored at the end of the converted character string.

### DSTR

- (1) Adds a decimal point to the BIN 32-bit data designated by  $\text{S2}$  at the location designated by  $\text{S1}$ , converts the data to character string data, and stores it following the device number designated by  $\text{D}$ .



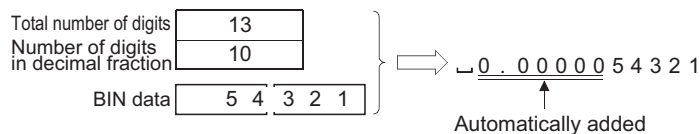
- (2) The total number of digits that can be designated by  $\text{S1}$  is from 2 to 13.
- (3) The number of digits that can be designated by  $\text{S1} + 1$  as a part of the decimal fraction is from 0 to 10.  
However, the number of digits following the decimal point must be smaller than or equal to the total number of digits minus 3.
- (4) The BIN data that can be designated by  $\text{S1}$  and  $\text{S2} + 1$  is within the range of from -2147483648 to 2147483647.
- (5) After conversion, character string data is stored at the device number following  $\text{D}$  as indicated below:
- The sign "20H" (space) will be stored if the BIN data is positive, and the sign "2DH" (minus sign) will be stored if it is negative.
  - If the setting for the number of digits after the decimal fraction is anything other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits.



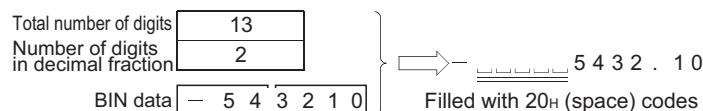
If the number of digits in the decimal fraction part of the number is "0", the ASCII code "2EH" (.) will not be stored.



- (c) If the total number of digits following the decimal fraction is greater than the number of BIN data digits, a zero will be added automatically and the number converted by shifting to the right, so that it would become "0.00000".



- (d) If the total number of digits excluding the sign and the decimal point is greater than the number of BIN data digits, "20H" (space) will be stored between the sign and the numeric value.



If the number of BIN digits is greater, an error will be returned.

- (e) The value "00H" is automatically stored at the end of the converted character string.

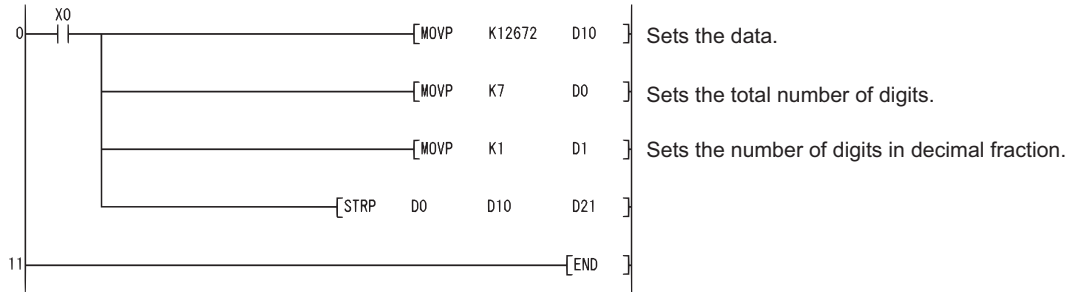
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The total number of digits designated by  $\textcircled{S1}$  is outside the ranges shown below. (Error code: 4100)
    - When the STR instruction is in use .....2 to 8
    - When the DSTR instruction is in use.....2 to 13
  - The number of digits designated as a part of the decimal fraction by  $\textcircled{S1}+1$  is outside the range shown below. (Error code: 4100)
    - When the STR instruction is in use .....0 to 5
    - When the DSTR instruction is in use.....0 to 10
  - The relationship between the total number of digits designated by  $\textcircled{S1}$  and the number of digits in the decimal fraction designated by  $\textcircled{S1}+1$  is not as shown below: (Error code: 4100)
    - Total number of digits minus 3 is equal to or larger than the number of digits in the decimal fraction.
  - The number of digits designated by  $\textcircled{S1}$  is smaller than "2 + number of digits of the BIN data, designated by  $\textcircled{S2}$ ". (Error code: 4100)
    - (Number of digits of  $\textcircled{S1}$  < Number of digits of the BIN data at  $\textcircled{S2}$  without a sign + Number of digits of a sign (+ or -) + Number of digits of decimal point (.))
  - The device range where the character string designated by  $\textcircled{D}$  will be stored exceeds the relevant device range. (Error code: 4101)

## Program Example

- (1) The following program converts the BIN 16-bit data stored at D10 when X0 is turned ON in accordance with the digit designation of D0 and D1, and stores the result from D20 to D23.

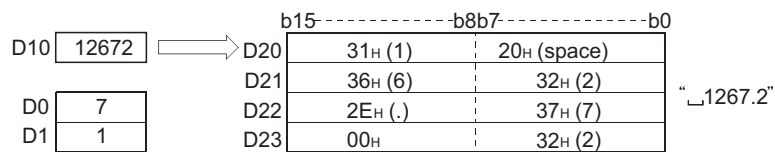
[Ladder Mode]



[List Mode]

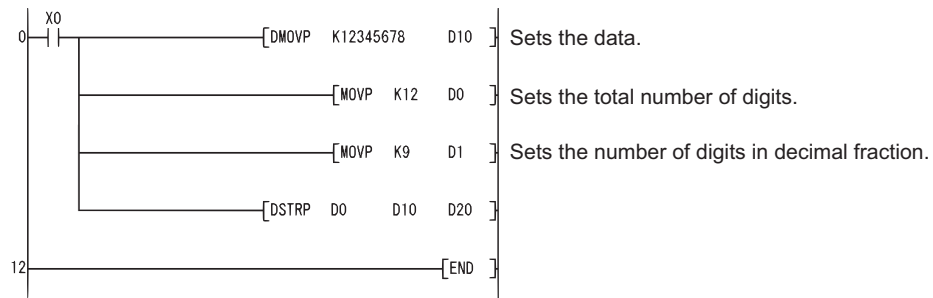
Step	Instruction	Device
0	LD	X0
1	MOV	K12672 D10
3	MOV	K7 D0
5	MOV	K1 D1
7	STR	D0 D10 D21
11	END	

[Operation]



- (2) The following program converts the BIN 32-bit data stored at D10 and D11 when X0 is turned ON in accordance with the digit designation of D0 and D1, and stores the result at from D20 to D26.

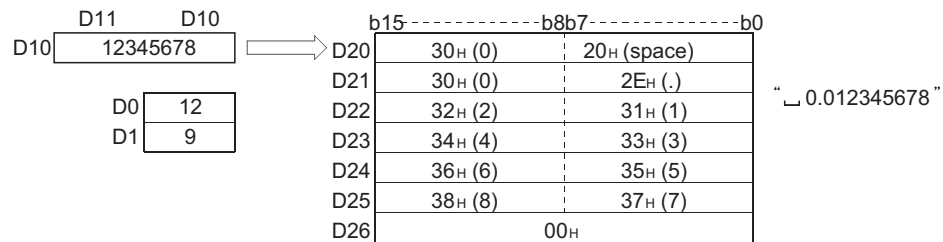
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DMOVP	K12345678 D10
4	MOVP	K12 D0
6	MOVP	K9 D1
8	DSTRP	D0 D10 D20
12	END	

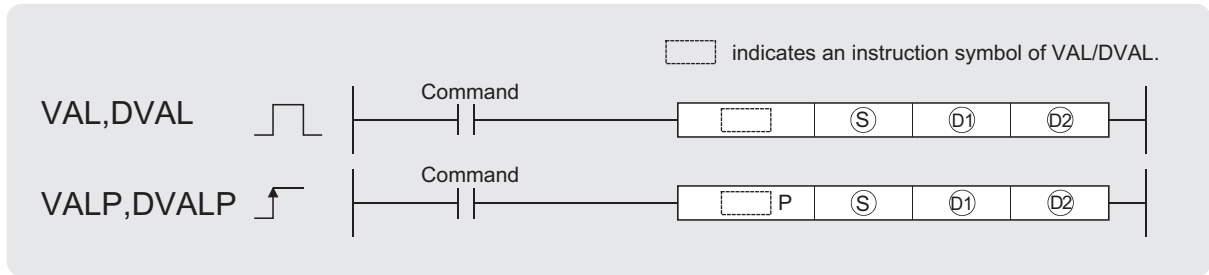
[Operation]



# 7.11.10 Conversion from character string to BIN 16-bit or 32-bit data (VAL(P),DVAL(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger  
(Corresponding GX Developer :Version 8.00 A or later).



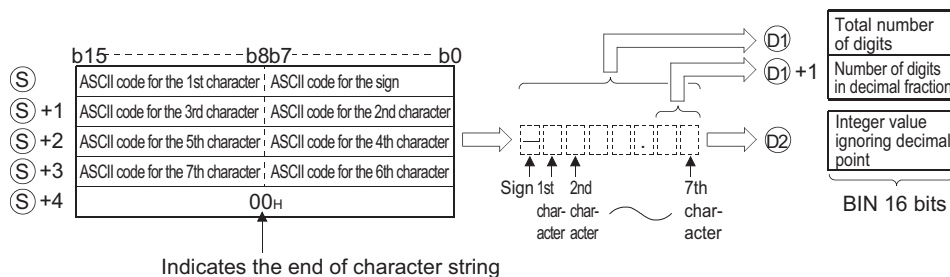
- Ⓢ : Character string to be converted to BIN data or head number of the devices where the character string is stored (character string)
- Ⓣ1 : Head number of the devices where the number of digits of the converted BIN data will be stored (BIN 16 bits)
- Ⓣ2 : Head number of the devices where the converted BIN data will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	JMO		UAGO	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ1	○	○				—		—	—
Ⓣ2	○	○				○		—	—

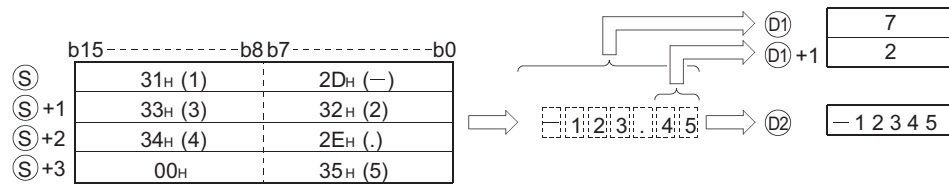
## ★ Function

### VAL

- (1) Converts character strings stored in the device numbers starting from that designated at Ⓢ to BIN 16-bit data, and stores the number of digits and BIN data in Ⓣ1 and Ⓣ2.  
For conversions from character strings to BIN, all data from the device number designated by Ⓢ to the device number where "00H" is stored will be processed as character strings.



For example, if the character string "-123.45" is designated for the area starting from (S), the operation result would be stored at (D1) and (D2) in the following manner:



- (2) The total number of characters that can be designated as a character string at (S) is from 2 to 8.

- (3) From 0 to 5 characters from the character string designated at (S) can become the decimal fraction part.

However, this number must not exceed the total number of digits minus 3.

- (4) The range of the numerical character string that can be converted to BIN value is from -32768 to 32767, ignoring a decimal point.

Numerical value character strings, excluding the sign and the decimal point, can be designated only within the range from "30H" to "39H".

The value ignoring a decimal point means:

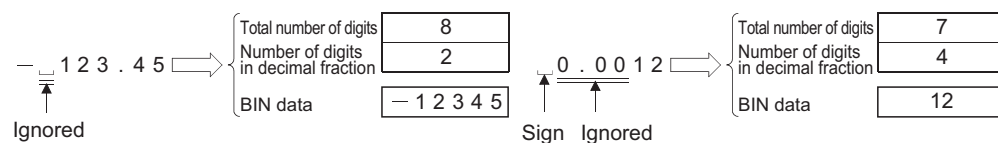
**Example**: "-12345.6" → "-123456"

- (5) The sign "20H" will be stored if the numerical value is positive, and the sign "2DH" will be stored if it is negative.
- (6) "2EH" is set for the decimal point.
- (7) The total number of digits stored at (D1) amounts to all characters expressing numerical values (including signs and decimal points).

The characters following the decimal point stored at (D1)+1 include the number of characters from "2EH" (.) onward.

The BIN data stored at (D2) is the character string ignoring the decimal point that has been converted to BIN data.

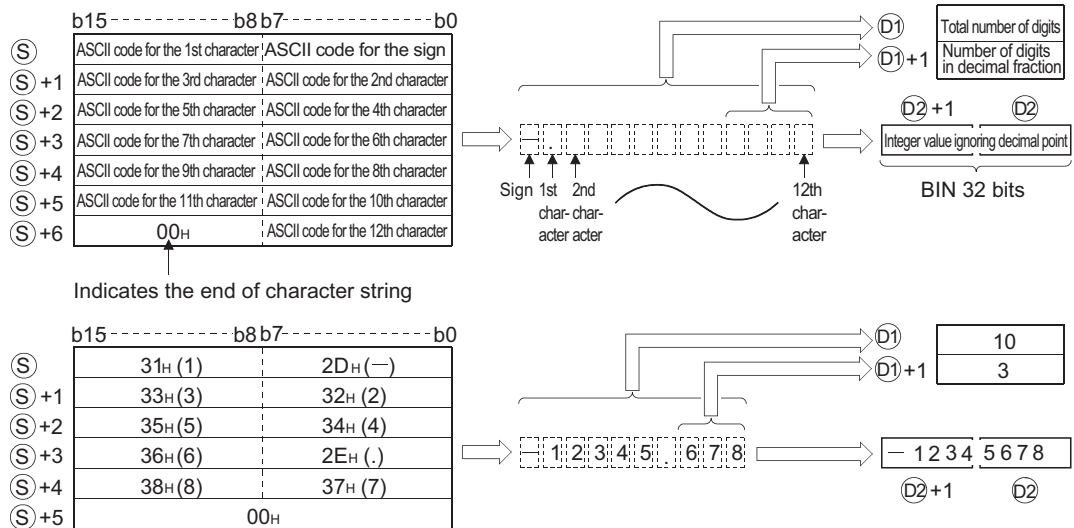
- (8) In cases where the character string designated by (S) contains "20H" (space) or "30H" (0) between the sign and the first numerical value other than "0", these "20H" and "30H" are ignored in the conversion into a BIN value.



## DVAL

- (1) Converts the character string stored in the area starting from the device designated by (S) to BIN 32-bit data, and stores the digits numbers and BIN data in (D1) and (D2).

For conversions from character strings to BIN, all data from the device number designated by (S) to the device number where "00H" is stored will be processed as character strings.

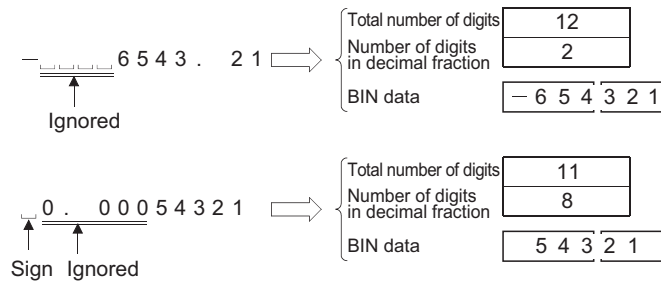


- (2) The total number of characters in the character string indicated by (S) is from 2 to 13.
- (3) From 0 to 10 characters in the character string indicated by (S) can be the decimal fraction part.  
However, this number must not exceed the total number of digits minus 3.
- (4) The range of the numerical character string that can be converted to BIN value is from -2147483648 to 2147483647, excluding the decimal point.  
Numerical value character strings, excluding the sign and the decimal point, can be designated only within the range from "30H" to "39H".
- (5) The sign "20H" will be stored if the numerical value is positive, and the sign "2DH" will be stored if it is negative.
- (6) "2EH" is set for the decimal point.
- (7) The total number of digits stored at D1 amounts to all characters expressing numerical values (including signs and decimal points).

The characters following the decimal point stored at (D1)+1 include the number of characters from "2EH" (.) onward.

The BIN data stored at (D2) is the character string ignoring the decimal point that has been converted to BIN data.

- (8) In cases where the character string designated by ⑤ contains "20H" (space) or "30H" (0) between the sign and the first numerical value other than "0", these "20H" and "30H" are ignored in the conversion into a BIN value.



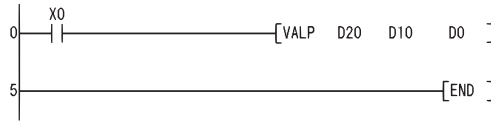
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The number of characters in the character string designated by ⑤ falls outside the ranges shown below: (Error code: 4100)
    - When VAL instruction is in use ..... From 2 to 8 characters
    - When DVAL instruction is in use ..... From 2 to 13 characters
  - The number of characters in the decimal fraction portion of the character string designated by ⑤ falls outside the ranges shown below: (Error code: 4100)
    - When VAL instruction is in use ..... 0 to 5
    - When DVAL instruction is in use ..... 0 to 10
  - The total number of characters in the character string designated by ⑤ and the number of characters in the decimal fraction part stand in a relationship that is outside the range indicated below: (Error code: 4100)
    - Total number of characters minus 3 is equal to or greater than the number of characters in the decimal fraction part.
  - An ASCII code other than "20H" or "2DH" has been set for the sign. (Error code: 4100)
  - An ASCII code other than from "30H" to "39H" or "2EH" (decimal point) has been set as a digit for one of the individual numbers. (Error code: 4100)
  - There has been more than one decimal point set in the value. (Error code: 4100)
  - The value of the BIN value when converted falls outside the following ranges: (Error code: 4100)
    - When VAL instruction is in use .....  $-32768$  to  $32767$
    - When DVAL instruction is in use .....  $-2147483648$  to  $2147483647$
  - No "00H" is set within the range from the device number designated by ⑤ to the last device number of the relevant device. (Error code: 4101)

## Program Example

- (1) The following program reads the character string data stored from D20 to D22 as an integer, converts it to a BIN value, and stores it at D0 when X0 is ON.

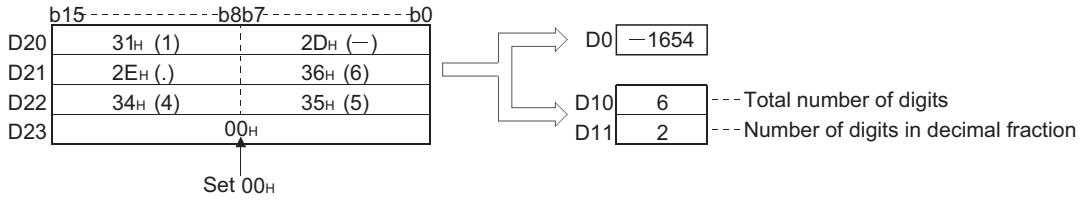
[Ladder Mode]



[List Mode]

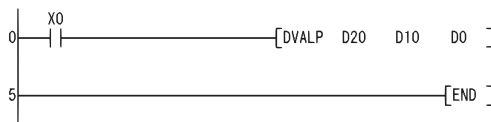
Step	Instruction	Device
0	LD	X0
1	VALP	D20 D10 D0
5	END	

[Operation]



- (2) The following program reads the character string data stored from D20 to D24 as an integer, converts it to a BIN value, and stores it at D0 when X0 is ON.

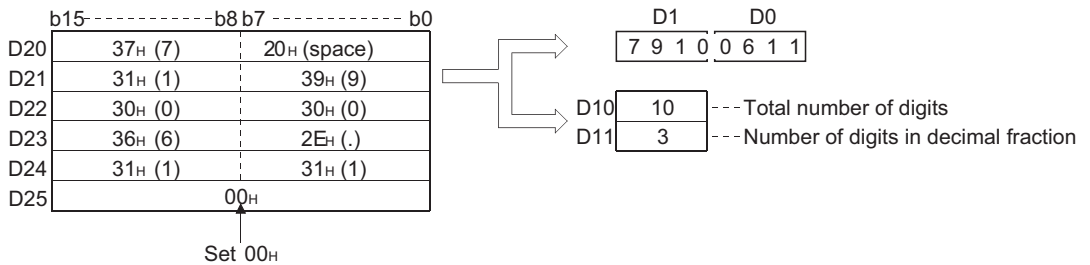
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DVALP	D20 D10 D0
5	END	

[Operation]

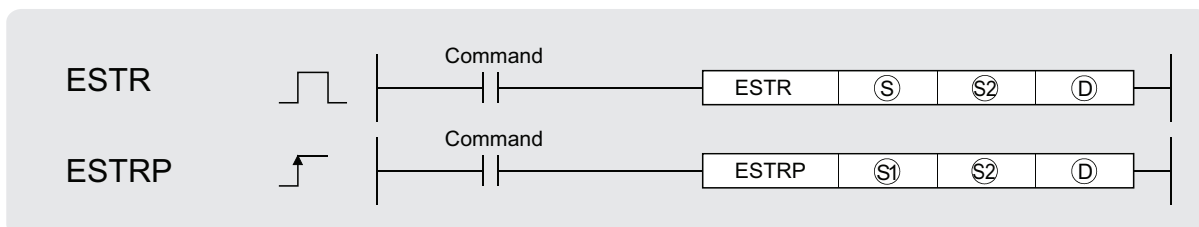




## 7.11.11 Conversion from floating decimal point to character string data (ESTR(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ<sub>1</sub>: 32-bit floating decimal point data to be converted or head number of the devices where the data is stored (real number)

Ⓢ<sub>2</sub>: Head number of the devices where display designation for the numerical value to be converted is stored (BIN 16 bits)

Ⓓ: Head number of the devices where the converted character string will be stored (character string)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○	—	—	○	○*1	○	—	
Ⓢ <sub>2</sub>	—	○	—	—	—	—	—	—	
Ⓓ	—	○	—	—	—	—	—	—	

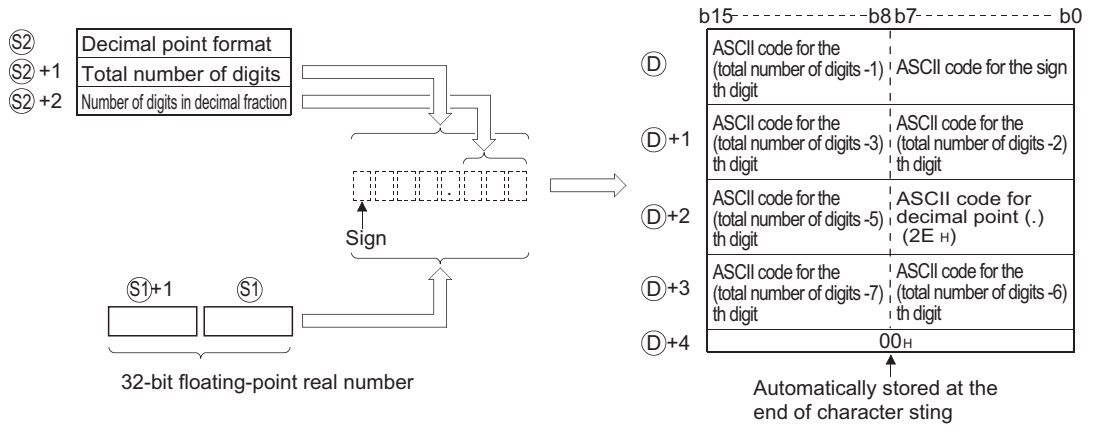
\*1: Available only in multiple Universal model QCPU

### ★ Function

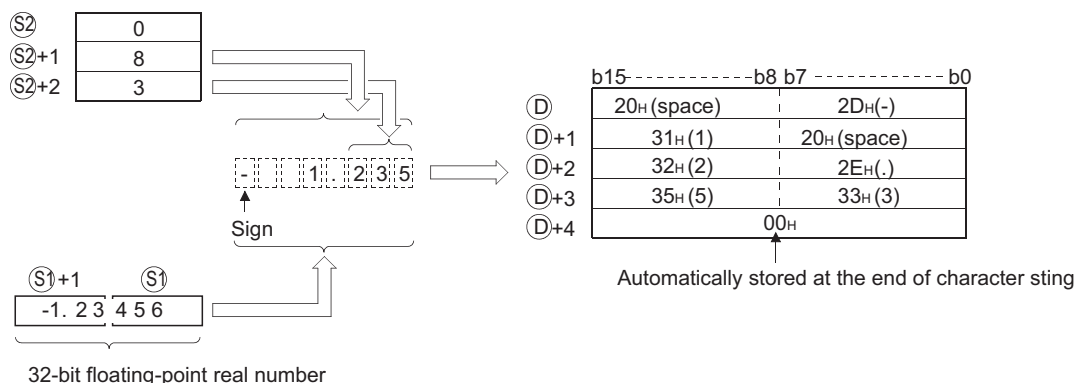
- Converts the 32-bit floating decimal point data designated by Ⓢ<sub>1</sub> to a character string according to the display designation specified by Ⓢ<sub>2</sub>, and stores the result into the area starting from the device number designated by Ⓓ.
- The post-conversion data differs depending on the display designation designated by Ⓢ<sub>2</sub>.

Ⓢ <sub>2</sub>	0: Decimal point format 1: Exponent format	} The converted data differs depending on the format selected at Ⓢ <sub>2</sub> . ... Setting is possible in the range from 2 to 24.
Ⓢ <sub>2</sub> + 1	Total number of digits	
Ⓢ <sub>2</sub> + 2	Number of digits in decimal fraction	

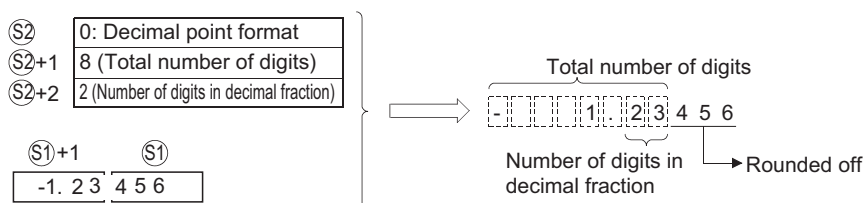
#### When using decimal point format



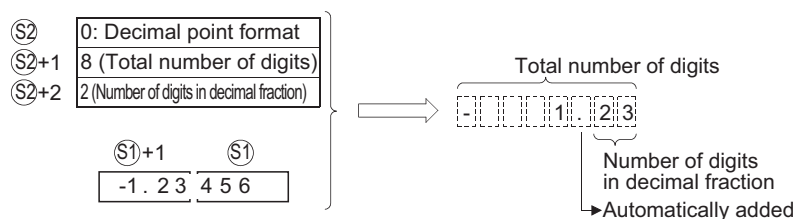
For example, in a case where there are 8 digits in total, with 3 digits in the decimal fraction part, and the value designated is -1.23456, the operation result would be stored in the area starting from  $\textcircled{D}$  in the following manner:



- (a) The total number of digits that can be designated by  $\textcircled{S2}+1$  is as shown below:  
 When the number of decimal fraction digits is "0"  
 ..... Number of digits (max.: 24)  $\cong$  2  
 When the number of decimal fraction digits is other than "0"  
 ..... Number of digits (max.: 24)  $\cong$  (Number of decimal fraction digits + 3)
- (b) The number of digits of decimal fraction part that can be designated by  $\textcircled{S2}+2$  is from 0 to 7.  
 However, the number of digits following the decimal point must be smaller than or equal to the total number of digits minus 3.
- (c) The converted character string data is stored at the area starting from the device number  $\textcircled{D}$  as indicated below:
  - 1) The sign "20H" (space) will be stored if the 32-bit floating decimal point type real number is positive, and the sign "2DH" (minus sign) will be stored if it is negative.
  - 2) If the decimal fraction part of a 32-bit floating point real number data is out of the range of the digits of decimal fraction part, the lower decimal values will be rounded off.

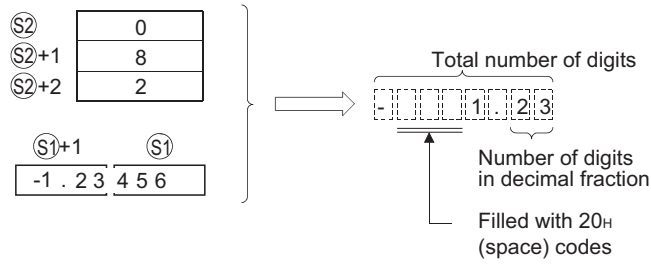


- 3) If the number of digits following the decimal point has been set at any value other than "0", "2EH" (.) will automatically be stored at the position before the first of the specified number of digits.



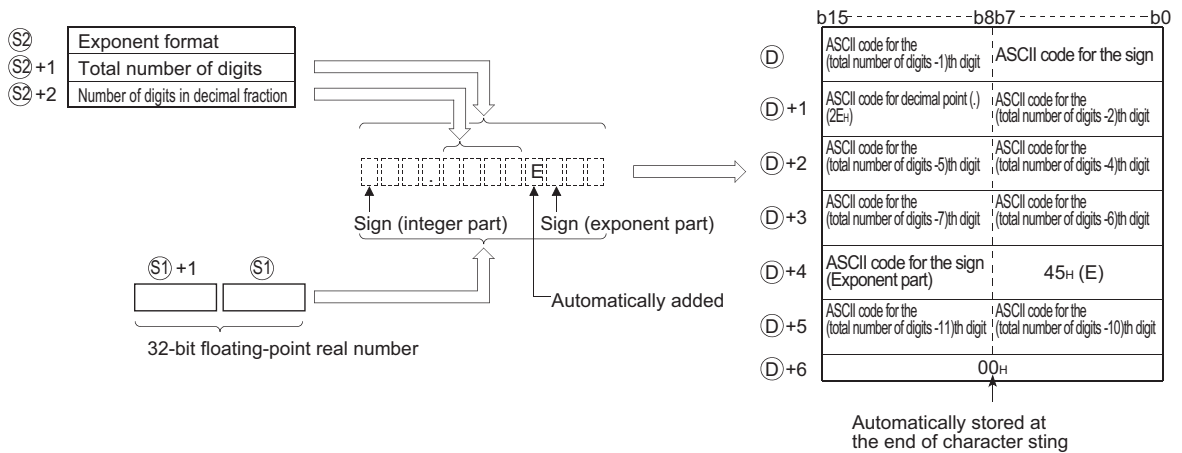
If the number of digits in the decimal fraction part is "0", the ASCII code "2EH" (.) will not be stored.

- 4) If the total number of digits, excluding the sign, the decimal point and the decimal fraction part, is greater than the integer part of the 32-bit floating point type real number data, "20H (space)" will be stored between the sign and the integer part.

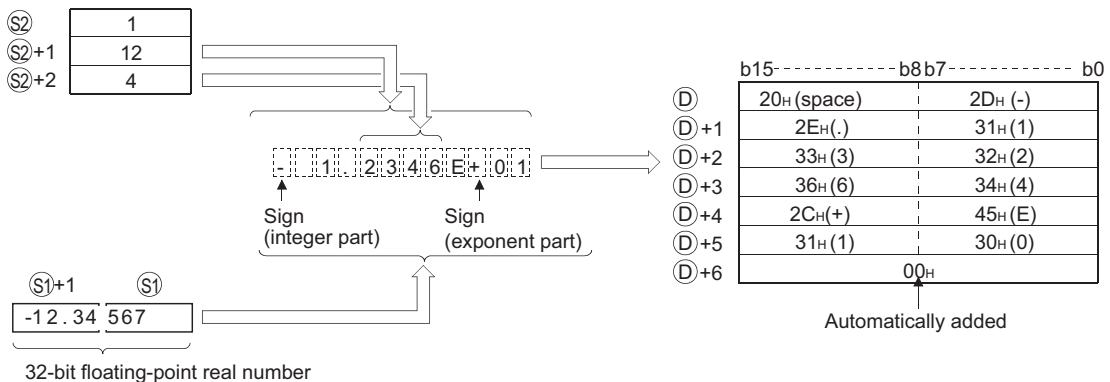


- 5) The value "00H" is automatically stored at the end of the converted character string.

**When using exponent format**

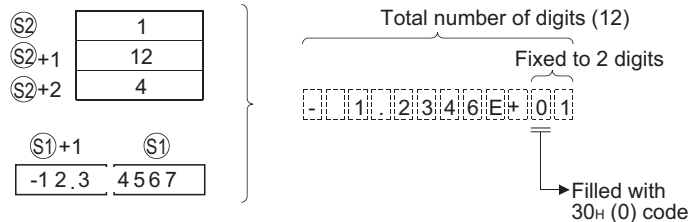


For example, in a case where there are 12 digits in total, with 4 digits in the decimal fraction portion, and the value designated is -12.34567, the operation results would be stored in the area starting from  $D$  in the following manner:





- 5) The ASCII code "2CH" (+) will be stored as the sign for the exponent portion of the value if the exponent is positive in value, and the code "2DH" (-) will be stored if the exponent is a negative value.
- 6) The exponent portion is fixed at 2 digits.  
If the exponent portion is only 1 digit, the ASCII code "30H" (0) will be stored between the sign and the exponent portion of the number.



- 7) The value "00H" is automatically stored at the end of the converted character string.

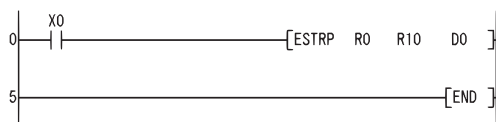
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The  $S1$  value is not within the range indicated below: (Error code: 4100)  
 $0, 2^{-126} \leq |S1| < 2^{128}$
  - The format designated by  $S2$  was neither 0 nor 1. (Error code: 4100)
  - The total number of digits designated by  $S2 + 1$  is outside the ranges shown below. (Error code: 4100)
    - When using decimal point format
      - When the number of decimal fraction digits is "0"
        - ..... Total number of digits  $\cong 2$
      - When the number of decimal fraction digits is other than "0"
        - ..... Total number of digits  $\cong$  (Number of decimal fraction digits + 3)
    - When using exponent format
      - When the number of decimal fraction digits is "0"
        - ..... Total number of digits  $\cong 6$
      - When the number of decimal fraction digits is other than "0"
        - ..... Total number of digits  $\cong$  (Number of decimal fraction digits + 7)
  - The number of digits designated for the decimal fraction portion of the value by  $S2+2$  was outside the ranges indicated below: (Error code: 4100)
    - When using the decimal point format
      - ..... Number of decimal fraction digits  $\cong$  (Total number of digits - 3)
    - When using the exponent format
      - ..... Number of decimal fraction digits  $\cong$  (Total number of digits - 7)
  - The value whose total digits exceeds "24" is specified. (Error code: 4100)
  - The device range to store the character string designated by  $D$  exceeds the relevant device range. (Error code: 4101)
  - The device specified by  $S2$  exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ . (Error code: 4140)  
(For the Universal model QCPU only.)

# Program Example

- (1) The following program converts the 32-bit floating point type real number data which had been stored at R0 and R1 in accordance with the conversion designation that is being stored at R10 to R12, and stores the result following D0 when X0 goes ON.

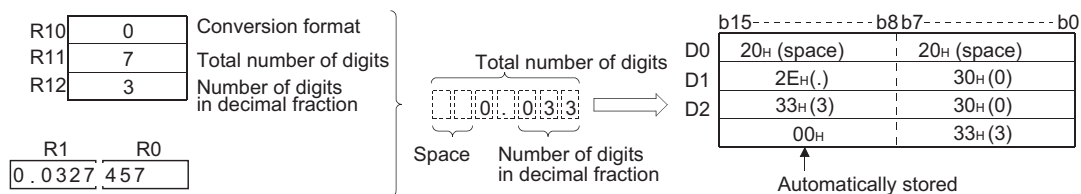
[Ladder Mode]



[List Mode]

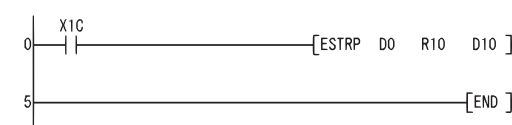
Step	Instruction	Device
0	LD	X0
1	ESTRP	R0 R10 D0
5	END	

[Operation]



- (2) The following program converts the 32-bit floating decimal point type real number data which had been stored at D0 and D1 in accordance with the conversion designation that is being stored at R10 to R12, and stores the result following D10 when X1C goes ON.

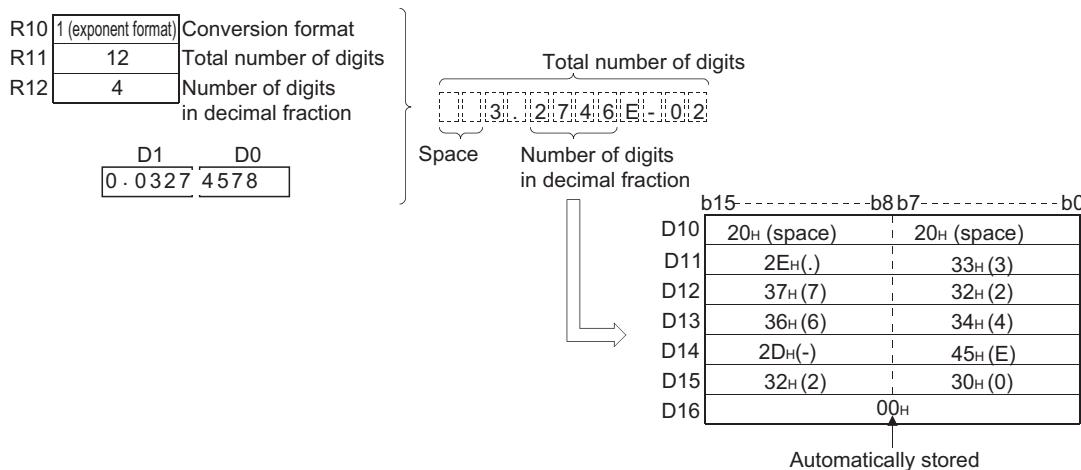
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	ESTRP	D0 R10 D10
5	END	

[Operation]



# 7.11.12 Conversion from character string to floating decimal point data (EVAL(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



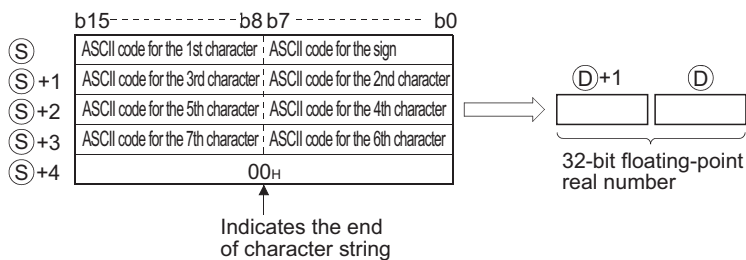
Ⓢ: Character string data to be converted to 32-bit floating decimal point real number data or head number of the devices where the character string data is stored (character string)

Ⓣ: Head number of the devices where the converted 32-bit floating decimal point real number data will be stored (real number)

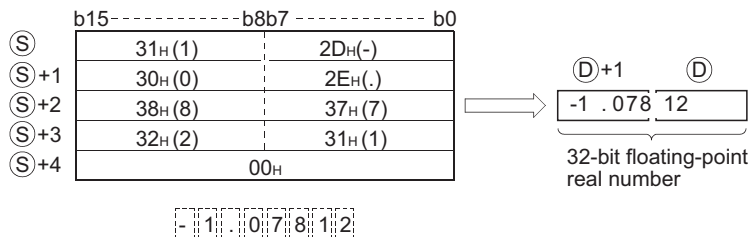
Setting Data	Internal Devices		R, ZR	J:G		U:G:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	—	—	—	○	—	
Ⓣ	—	○	—	○	—	—	—	—	

## ★ Function

- Converts character string stored in the area starting from the device number designated by Ⓢ to 32-bit floating point type real number, and stores result at device designated by Ⓣ.
- The designated character string can be converted to 32-bit floating point type real number data either in the decimal point format or the exponent format.

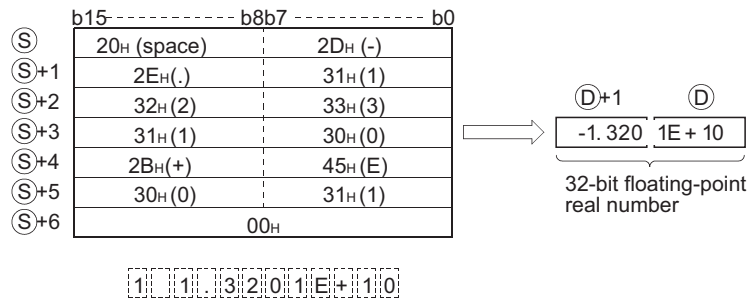


(a) When using decimal point format



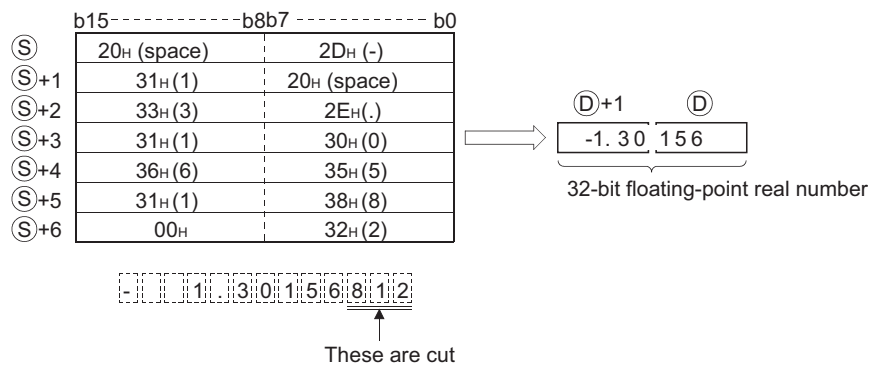


(b) When using exponent format

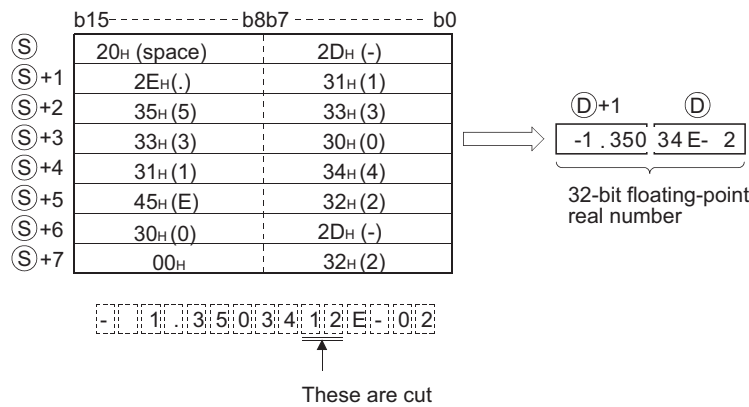


(3) Excluding the sign, decimal point, and exponent portion of the result, 6 digits of the character string designated by Ⓢ to be converted to a 32-bit floating decimal point type real number will be effective; the 7th digit on later digit will be cut from the result.

(a) When using decimal point format

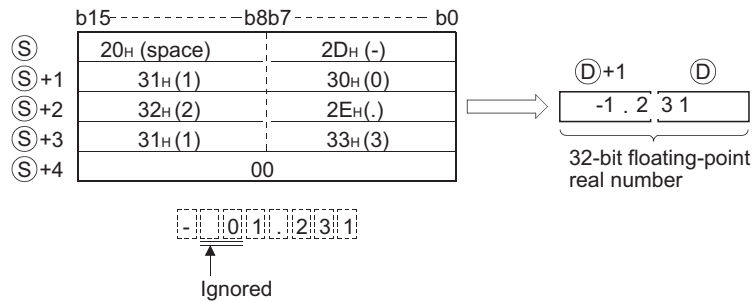


(b) When using exponent format

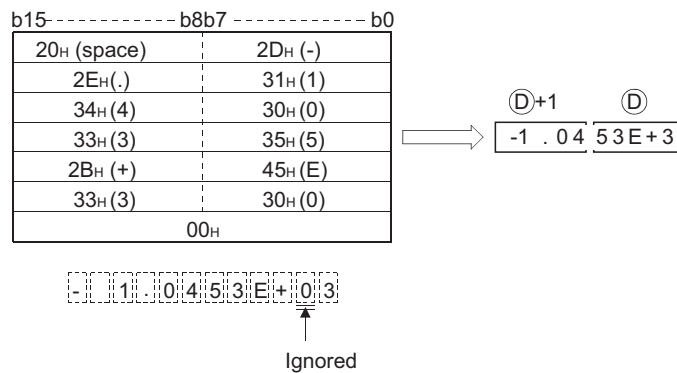


- (4) In the decimal point format, if "2B<sub>H</sub>" (+) is specified for the sign or if the designation of sign is omitted, conversion is made assuming a positive value.  
If "2D<sub>H</sub>" (-) is specified for the sign, the character string is converted assuming a negative value.
- (5) In the exponent format, if "2B<sub>H</sub>" (+) is specified for the sign in the exponent portion or if the designation of sign is omitted, conversion is made assuming a positive value.  
If "2D<sub>H</sub>" (-) is specified for the sign in the exponent portion, the character string is converted assuming a negative value.

- (6) In a case where the ASCII code "20H (space)" or "30H" (0) exists between numbers not including the initial zero in a character string specified by Ⓢ, it will be ignored when the conversion is done.



- (7) In a case where the ASCII code "30H (0)" exists between the character "E" and a number in an exponent format character string, the "30H" would be ignored when the conversion is performed.



- (8) If the "20H" (space) code is contained in the character string, the code is ignored in the conversion.
- (9) Up to 24 characters can be set for a character string.  
The codes "20H" (space) and "30H" (0) contained in the character string are also counted as a character.

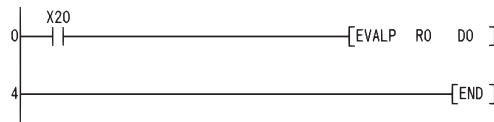
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The integer portion or the decimal fraction portion contains a character other than one in the range of from "30H" (0) to "39H" (9). (Error code: 4100)
  - There are two or more occurrences of the character "2EH" (.) in the character string designated by Ⓢ. (Error code: 4100)
  - The exponent portion contains the code (character) other than "45H"(E), "2BH"(+), "45H"(E) or "2DH"(-), or the string contains more than one exponent portion. (Error code: 4100)
  - Data after conversion is not within the following range. (Error code: 4100)  
 $0, 2^{-126} \leq |\text{data after conversion}| < 2^{128}$
  - The code "00H" does not appear in the range from Ⓢ to the relevant device. (Error code: 4101)
  - The number of characters in the character string following Ⓢ is either 0 or more than 24. (Error code: 4100)

## Program Example

- (1) The following program converts the character string stored in the area starting from R0 to a 32-bit floating decimal point type real number, and stores the result at D0 and D1 when X20 is turned ON.

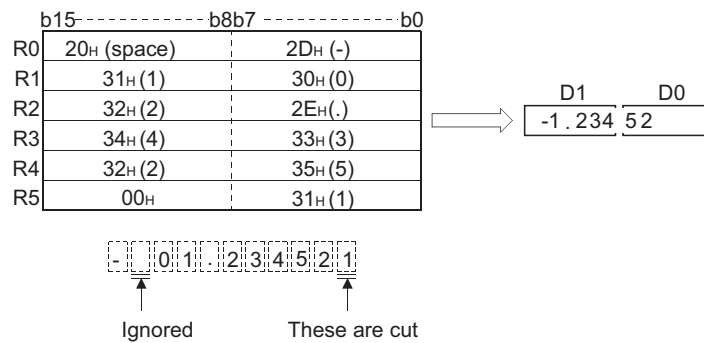
[Ladder Mode]



[List Mode]

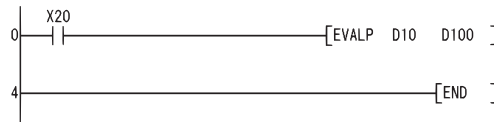
Step	Instruction	Device
0	LD	X20
1	EVALP	R0 D0
4	END	

[Operation]



- (2) The following program converts the character string stored in the area starting from D10 to a 32-bit floating decimal point type real number, and stores the result at D100 and D101 when X20 is turned ON.

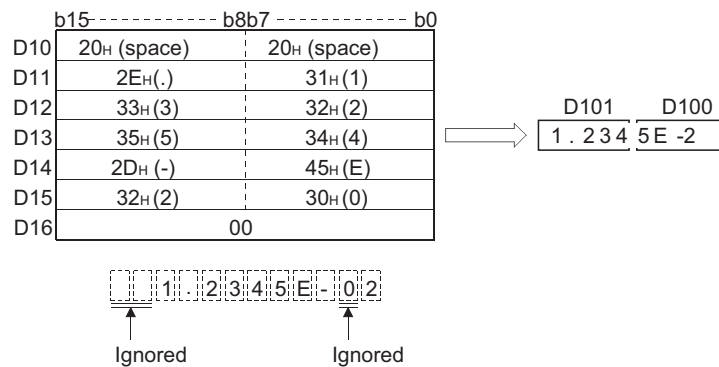
[Ladder Mode]



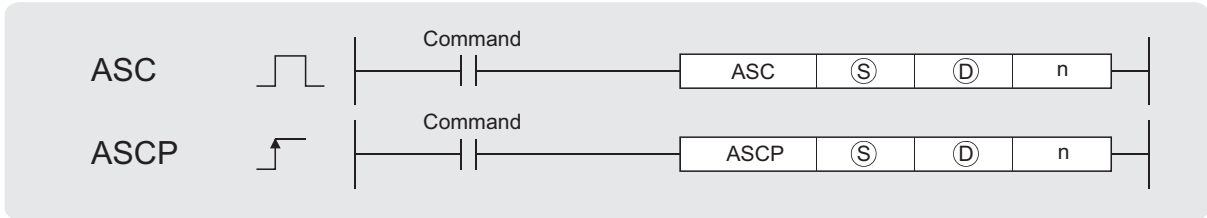
[List Mode]

Step	Instruction	Device
0	LD	X20
1	EVALP	D10 D100
4	END	

[Operation]



# 7.11.13 Conversion from hexadecimal BIN to ASCII (ASC(P))

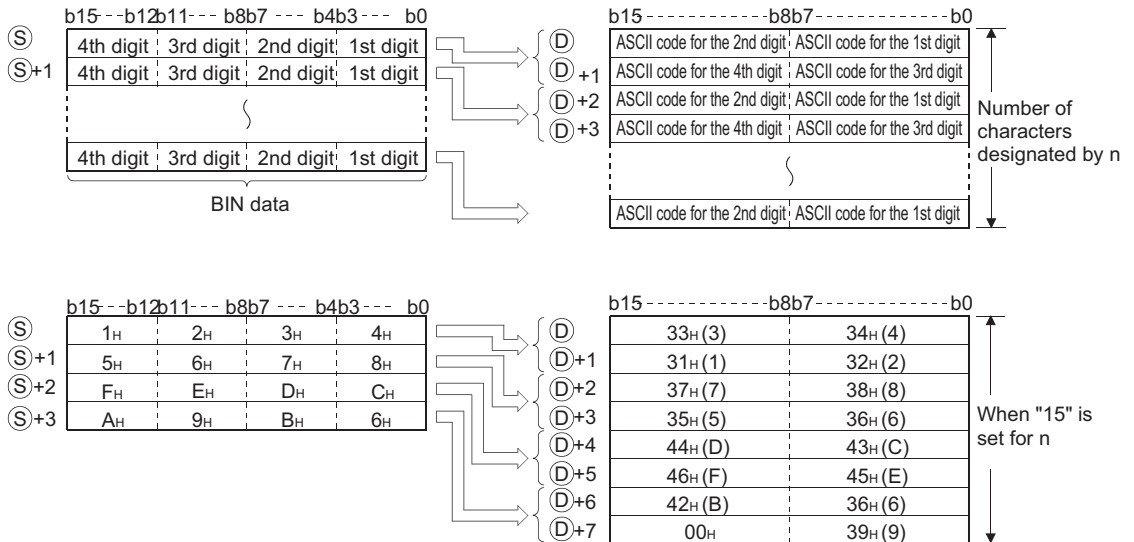


Ⓢ : Head number of the devices where BIN data to be converted to a character string is stored (BIN 16 bits)  
 ⓓ : Head number of the devices where the converted character string will be stored (character string)  
 n : Number of characters to be stored (BIN 16 bits)

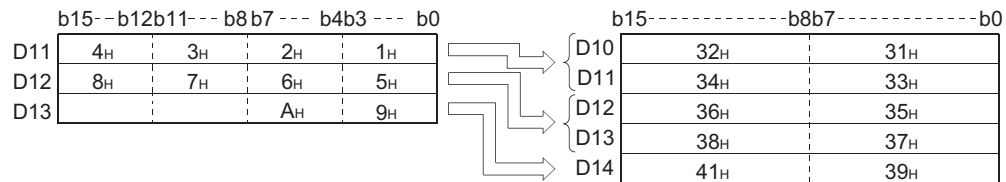
Setting Data	Internal Devices		R, ZR	JMO		UJGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
ⓓ	—	○				—			—
n	○	○				○			—

## ★ Function

- Converts the BIN 16-bit data stored in the area starting from the device designated by Ⓢ to ASCII by treating the BIN data in hexadecimal representation. Then, stores the converted data into the area starting from the device designated by ⓓ, for the number of characters specified by n.

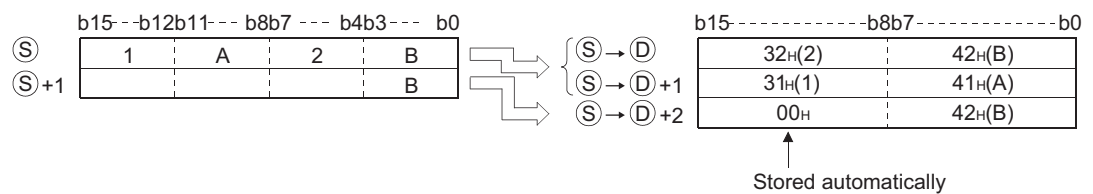


- (2) The use of n to set the number of characters causes the BIN data range designated by Ⓢ and the character string storage device range designated by Ⓧ to be set automatically.
- (3) Processing will be performed accurately even if the device range where BIN data to be converted is being stored overlaps with the device range where the converted ASCII data will be stored.



- (4) If an odd number of characters has been designated by n, the ASCII code "00H" will be automatically stored in the upper 8 bits of the final device in the range where the character string is to be stored.

When 5 characters have been designated by n.



- (5) If the number of characters designated by n is "0", conversion processing will not be conducted.

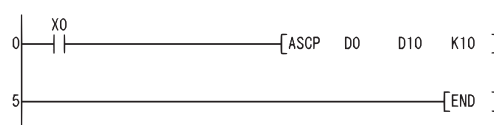
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The range for the number of characters designated by n following the device number designated by Ⓢ exceeds the relevant device range. (Error code: 4101)
  - The range for the number of characters designated by n following the device number designated by Ⓧ exceeds the relevant device range. (Error code: 4101)

## Program Example

- (1) The following program reads the BIN data being stored at D0 as hexadecimal values, converts them to a character string, and stores the result from D10 to D14 when X0 is turned ON.

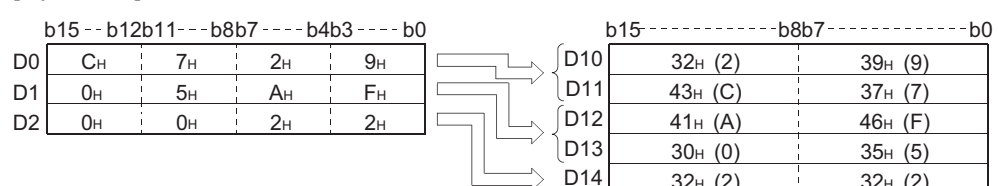
[Ladder Mode]



[List Mode]

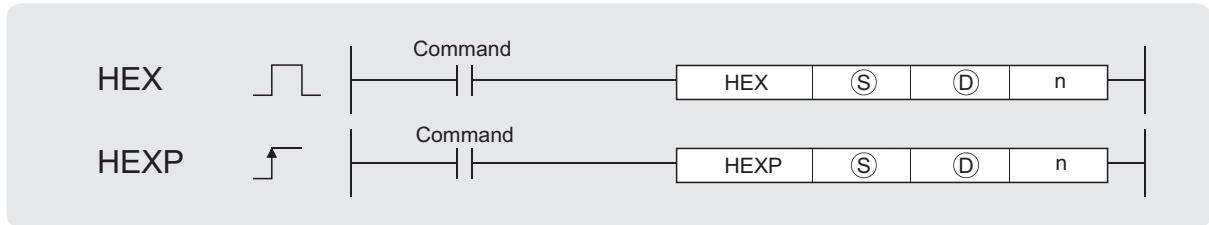
Step	Instruction	Device
0	LD	X0
1	ASCP	D0 D10 K10
5	END	

[Operation]



# 7.11.14 Conversion from ASCII to hexadecimal BIN (HEX(P))

Basic
High performance
Process
Redundant
Universal

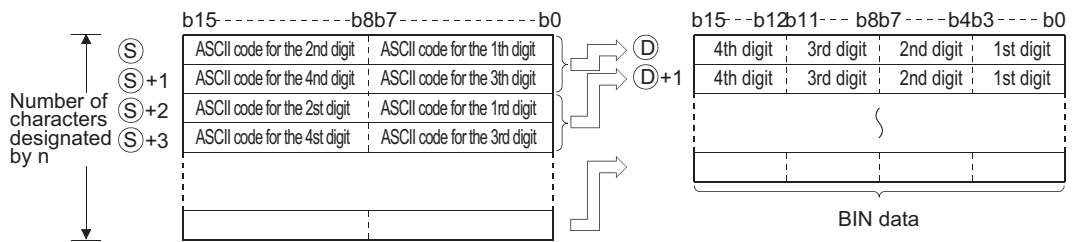


- Ⓢ : Head number of the devices where a character string to be converted to BIN data is stored (character string)
- Ⓣ : Head number of the devices where the converted BIN data will be stored (BIN 16 bits)
- n : Number of characters to be stored (BIN 16 bits)

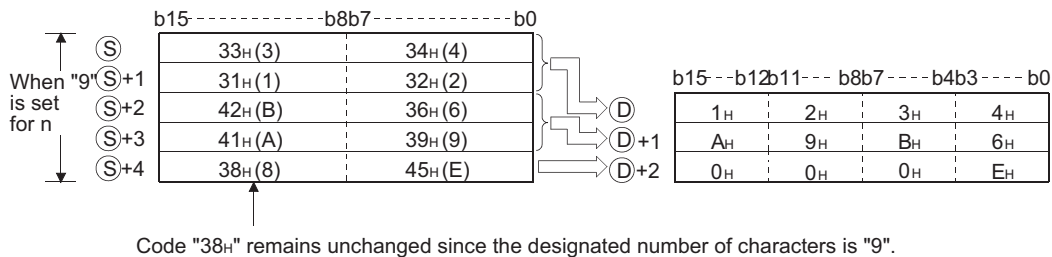
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—			—
Ⓣ	—	○				—			—
n	○	○				○			—

## ★ Function

- Converts the number of characters of hexadecimal ASCII data designated by n stored in the area starting from the device number designated by Ⓢ into BIN values and stores them in the area starting from the device number designated by Ⓣ.

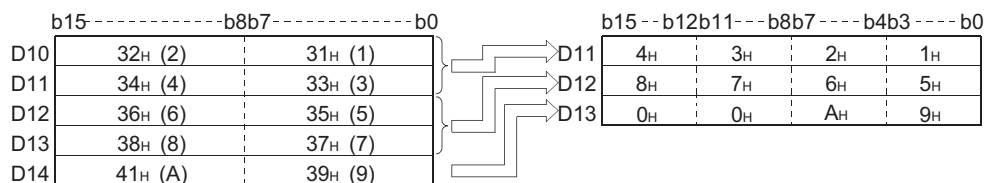


For example, if the number 9 has been designated by n, the operation would be as follows:

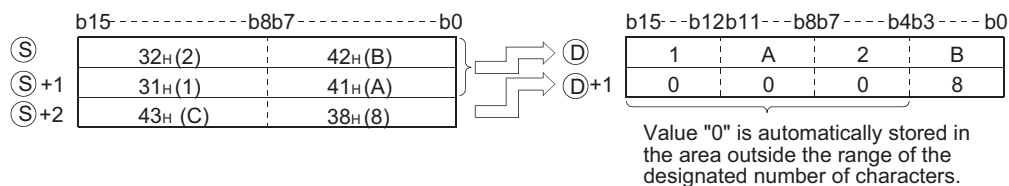


- When the number of characters is specified for n, the range of characters designated by Ⓢ as well as the device range designated by Ⓣ in which the BIN data will be stored are automatically decided.

- (3) Accurate processing will be conducted even in cases where the range of devices where the ASCII code to be converted is being stored overlaps with the range of devices that will store the converted BIN data.



- (4) If the number of characters designated by n is not divisible by 4, "0" will be automatically stored after the designated number of characters in the final device number of the devices which are storing the converted BIN values.



- (5) If the number of characters designated by n is "0", conversion processing will not be conducted.
- (6) ASCII code that can be designated by S includes from "30H" to "39H" and from "41H" to "46H".

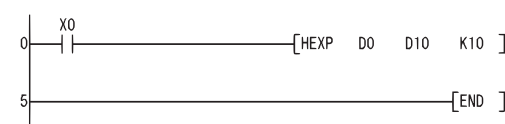
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Characters outside the hexadecimal character string (that is, characters that are not in the range of from "30H" to "39H", or from "41H" to "46H") have been set by S. (Error code: 4100)
  - The range for the number of characters designated by n following the device number designated by S exceeds the relevant device range. (Error code: 4101)
  - The range for the number of characters designated by n following the device number designated by D exceeds the relevant device range. (Error code: 4101)
  - The number of characters designated by n is a negative value. (Error code: 4101)

## Program Example

- (1) The following program converts the character string being stored from D0 to D4 to BIN data and stores the result from D10 to D14 when X0 goes ON.

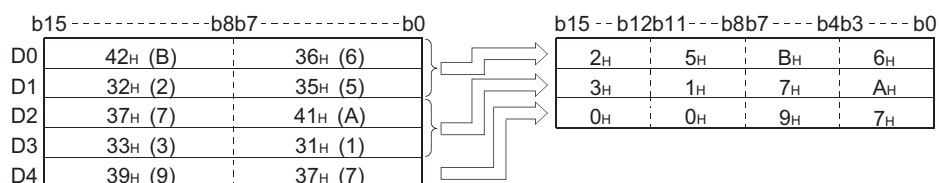
[Ladder Mode]



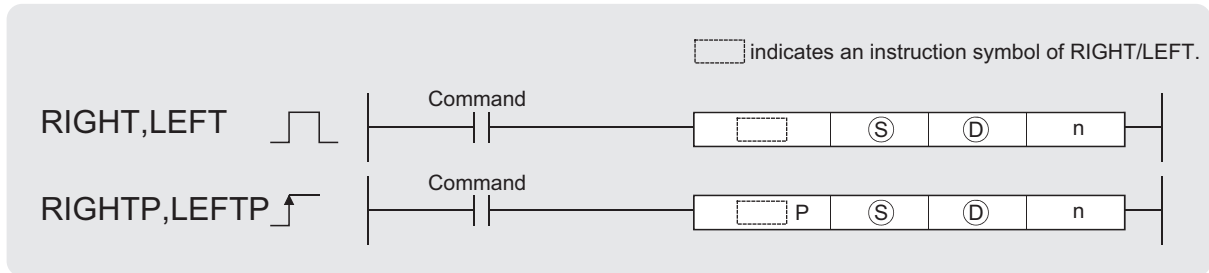
[List Mode]

Step	Instruction	Device
0	LD	X0
1	HEXP	D0 D10 K10
5	END	

[Operation]



# 7.11.15 Extracting character string data from the right or left (RIGHT(P),LEFT(P))



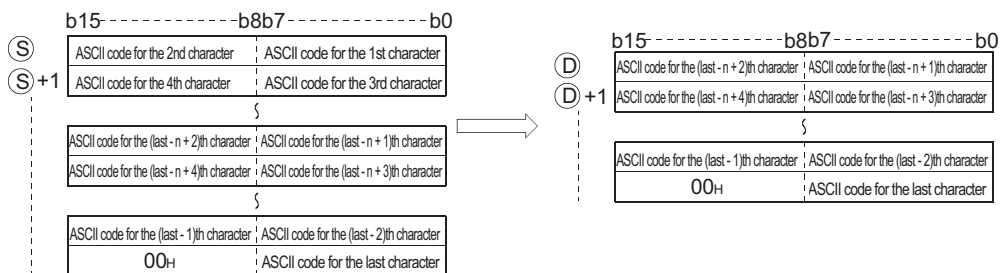
- Ⓢ : Character string or head number of the devices where the character string is stored (character string)
- Ⓣ : Head number of the devices where the character string consisting of n characters starting from the right or left of Ⓢ will be stored (character string)
- n : Number of characters to be extracted (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JAD		UDGO	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H	\$	
Ⓢ	—	○				—		—	○	—
Ⓣ	—	○				—		—	—	—
n	○	○				○		○	—	—

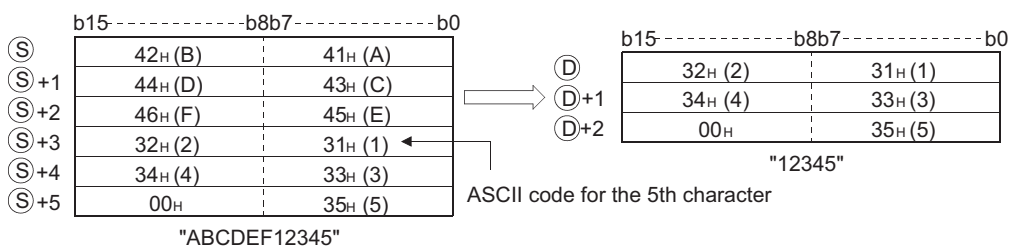
## ★ Function

### RIGHT

- (1) Stores n number of characters from the right side of the character string (the end of the character string) being stored in devices starting from that whose number is designated by Ⓢ, in devices starting from that whose number is designated by Ⓣ.



When n = 5

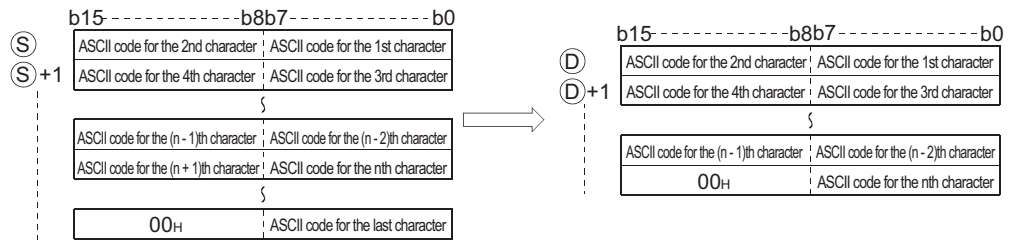




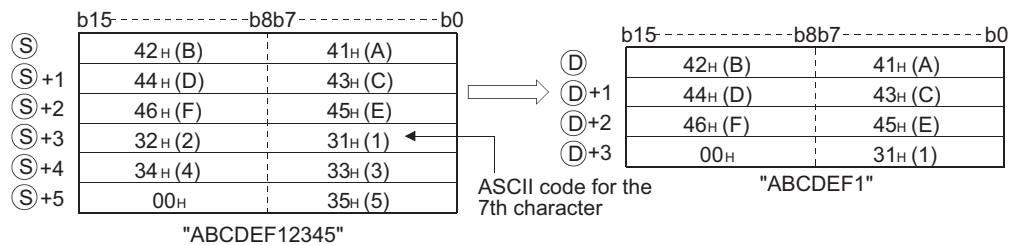
- (2) The NULL code (00H) indicating the end of the character string is automatically appended at the end of the character string. Refer to 3.2.5 for the format of the character string data.
- (3) If the number of characters designated by n is "0", the NULL code (00H) will be stored at  $\textcircled{D}$ .

## LEFT

- (1) Stores n number of characters from the left side of the character string (the beginning of the character string) being stored in devices starting from that whose number is designated by  $\textcircled{S}$ , in devices starting from that whose number designated by  $\textcircled{D}$ .



When n = 7



- (2) The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string.  
Refer to 3.2.5 for the format of the character string data.
- (3) If the number of characters designated by n is "0", the NULL code (00H) will be stored at  $\textcircled{D}$ .



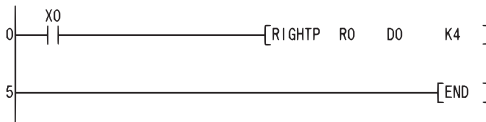
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of n exceeds the number of characters designated by  $\textcircled{S}$ . (Error code: 4101)
  - The range of n characters from  $\textcircled{D}$  exceeds the relevant device. (Error code: 4101)

## Program Example

- (1) The following program stores 4 characters of data from the rightmost of the character string stored in the area starting from R0, and stores it into the area starting from D0 when X0 is turned ON.

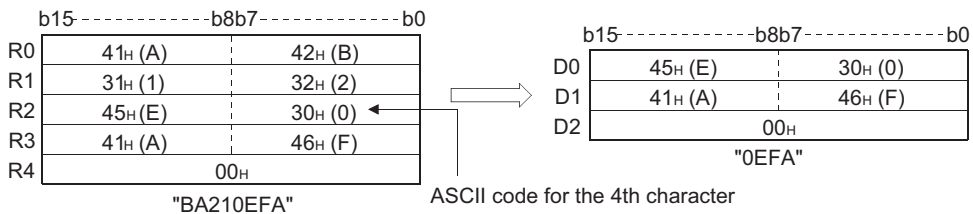
[Ladder Mode]



[List Mode]

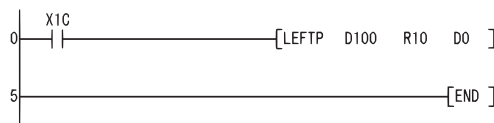
Step	Instruction	Device
0	LD	X0
1	RIGHTP	R0 D0 K4
5	END	

[Operation]



- (2) The following program stores the number of characters corresponding to the value being stored in D0 from the left of the character string data being stored at D100 to the area starting from R10 when X1C is turned ON.

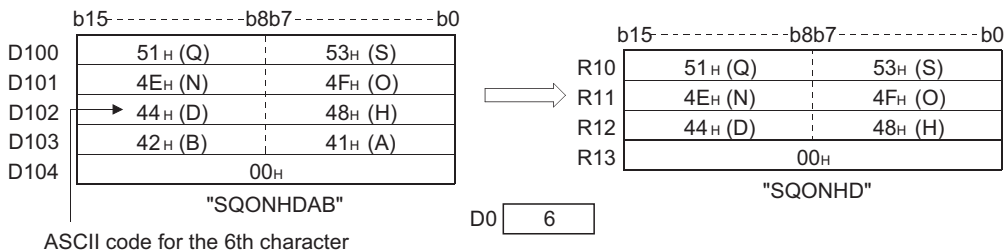
[Ladder Mode]



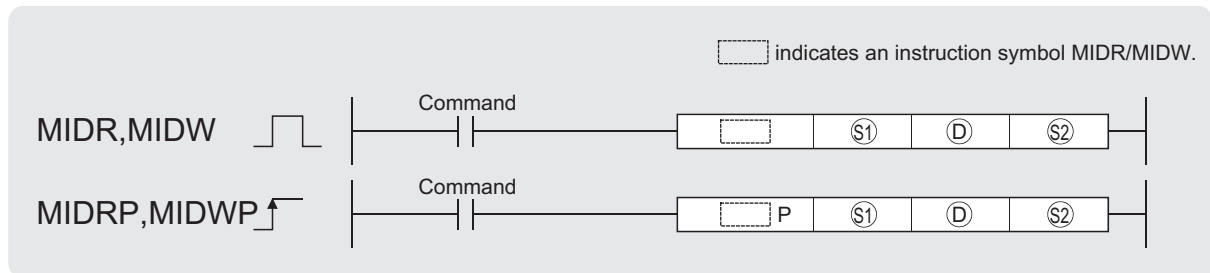
[List Mode]

Step	Instruction	Device
0	LD	X1C
1	LEFTP	D100 R10 D0
5	END	

[Operation]



## 7.11.16 Random selection from and replacement in character strings (MIDR(P),MIDW(P))



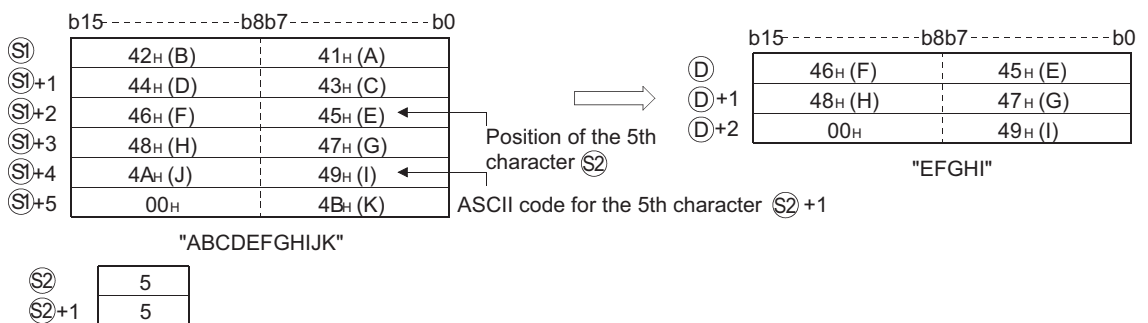
- Ⓢ1: Character string or head number of the devices where the character string is stored (character string)
- ⓈD: Head number of the devices where a character string data obtained as the result of operation will be stored (character string)
- Ⓢ2: Head number of the devices where the location of the first character and the number of characters will be stored (BIN 16 bits)
- Ⓢ2 : Position of first character
  - Ⓢ2 +1: Number of characters

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		○	—
ⓈD	—	○				—		—	—
Ⓢ2	○	○				○		—	—

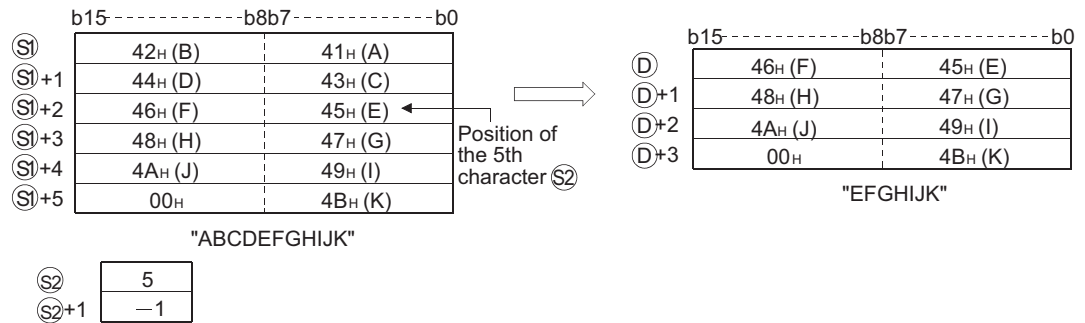
### ★ Function

#### MIDR

- (1) Extracts the character string data of Ⓢ2+1 characters, starting from the position designated by Ⓢ2, counted from the left end of the character string data designated by Ⓢ1, and stores the extracted data into the area starting from the device designated by ⓈD.

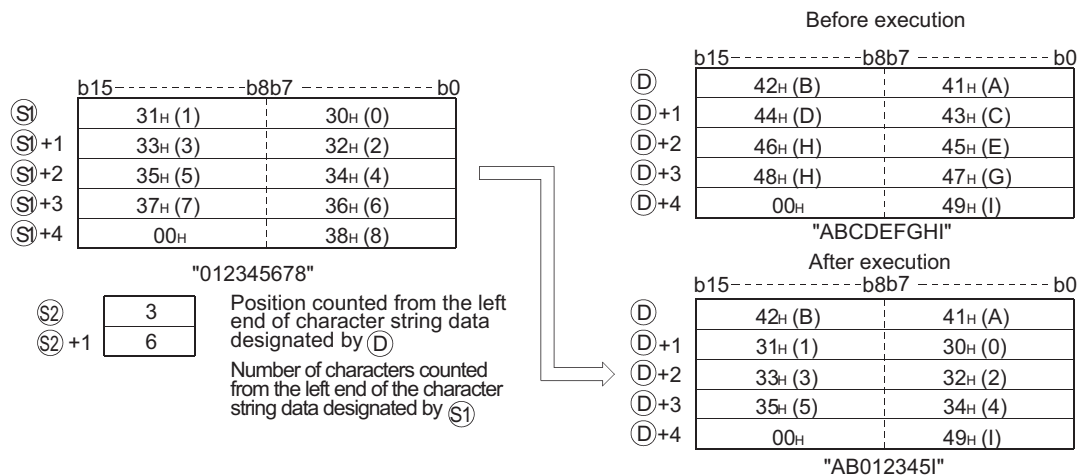


- (2) The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string.  
Refer to 3.2.5 for the format of the character string data.
- (3) No processing will be conducted if the number of characters designated by  $\textcircled{S2} + 1$  is "0".
- (4) If the number of characters designated by  $\textcircled{S2} + 1$  is "-1", stores the data up to the final character designated by  $\textcircled{S1}$  starting from the device designated by  $\textcircled{D}$ .



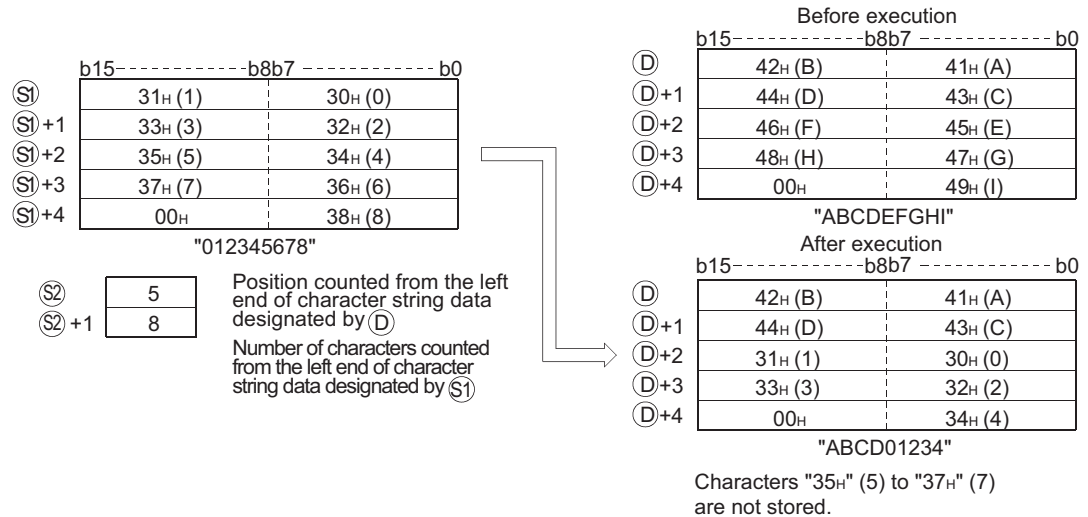
## MIDW

- (1) Extracts the character string data of  $\textcircled{S2} + 1$  characters, starting from the left end of the character string data designated by  $\textcircled{S1}$ , and stores the extracted data to the character string data designated by  $\textcircled{D}$  in the area starting from the position designated by  $\textcircled{D}$  from the left end.

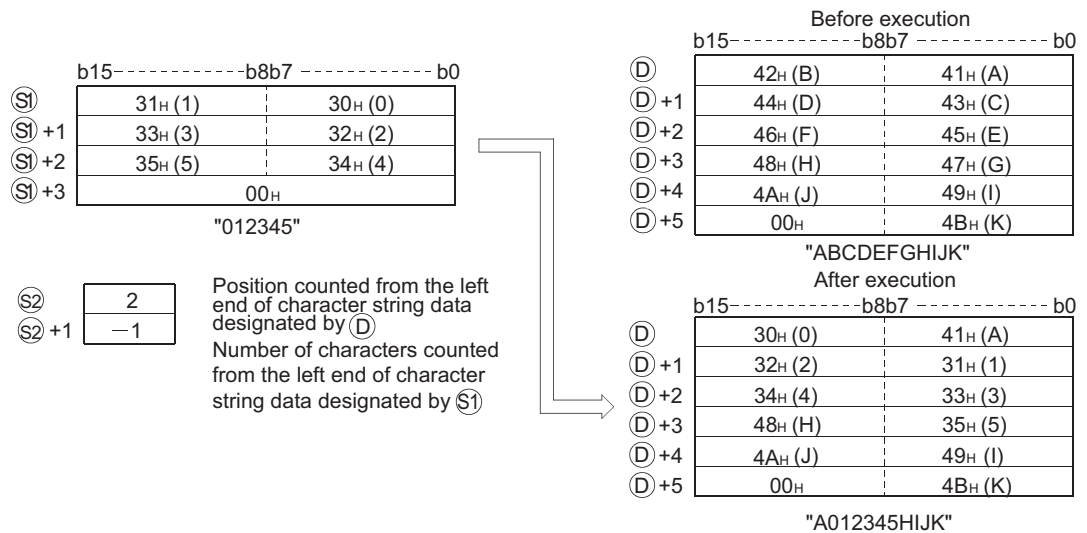


- (2) The NULL code (00H) indicating the end of the character string is automatically added to the end of the character string.  
Refer to 3.2.5 for the format of the character string data.
- (3) No processing will be conducted if the number of characters designated by  $\textcircled{S2} + 1$  is "0".

- (4) If the number of characters designated by  $\textcircled{S2}+1$  exceeds the final character from the character string data designated by  $\textcircled{D}$ , data will be stored up to the final character.



- (5) If the number of characters designated by  $\textcircled{S2}+1$  is "-1", stores the data up to the final character designated by  $\textcircled{S1}$  to the area starting from the device designated by  $\textcircled{D}$ .



## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

For MIDR instruction

- The value of  $\textcircled{S2}$  exceeds the number of characters designated by  $\textcircled{S1}$ . (Error code: 4101)
- The  $\textcircled{S2}+1$  number of characters from position  $\textcircled{D}$  exceeds the  $\textcircled{D}$  device range. (Error code: 4101)
- The  $\textcircled{S2}+0$  value is 0. (Error code: 4101)
- "00<sub>H</sub>" does not exist in the specified devices that follow the device specified for  $\textcircled{S1}$ . (Error code: 4101)

For MIDW instruction

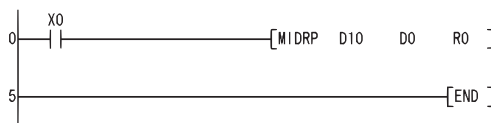
- The value of  $\textcircled{S2}$  exceeds the number of characters designated by  $\textcircled{D}$ . (Error code: 4101)

- The (S2) + 1 value exceeds the number of characters for (S1). (Error code: 4101)
- The (S2) + 0 value is 0. (Error code: 4101)
- "00H" does not exist in the specified devices that follow the device specified for (S1). (Error code: 4101)

## Program Example

- (1) The following program stores the 3rd character through the 6th character from the left of the character string stored in the area starting from D10 at devices starting from D0 when X0 is turned ON.

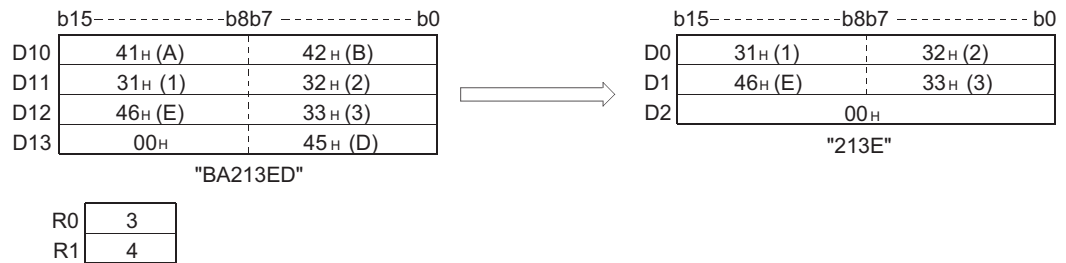
[Ladder Mode]



[List Mode]

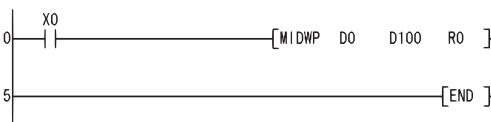
Step	Instruction	Device
0	LD	X0
1	MIDRP	D10 D0 R0
5	END	

[Operation]



- (2) The following program stores 4 characters of the character string data stored in the area starting from D0 into the area starting from the 3rd character from the left of the character string data in the area starting from D100 when X0 is turned ON.

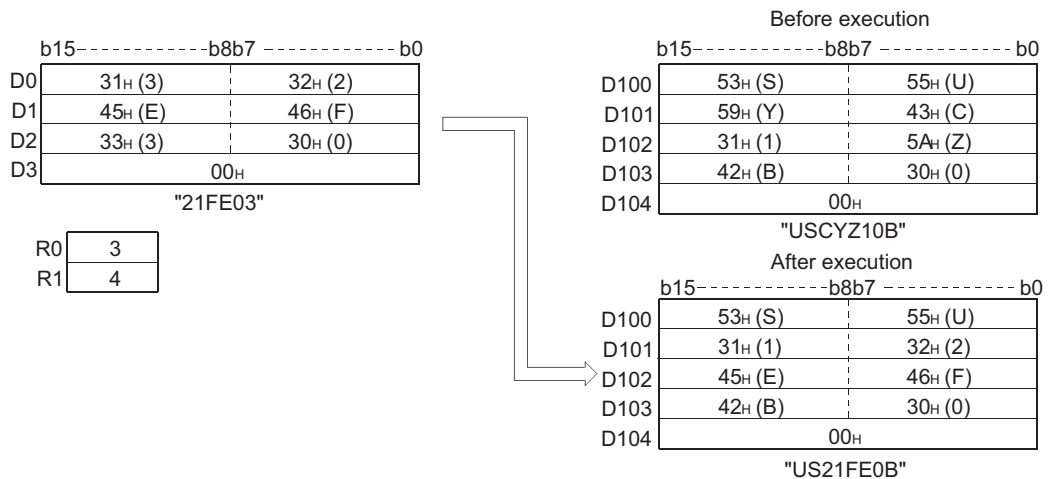
[Ladder Mode]



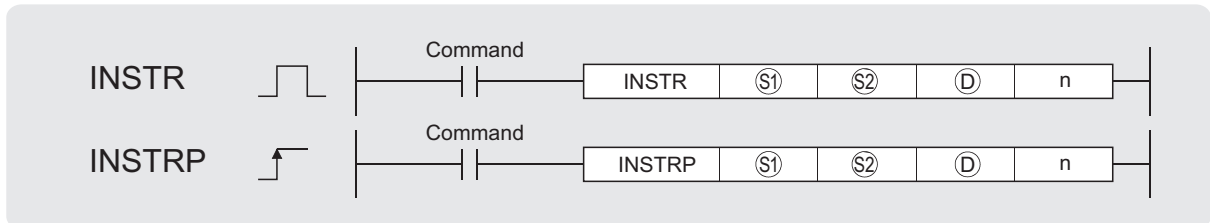
[List Mode]

Step	Instruction	Device
0	LD	X0
1	MIDWP	D0 D100 R0
5	END	

[Operation]



## 7.11.17 Character string search (INSTR(P))



- Ⓢ<sub>1</sub> : Character string to be searched or head number of the devices where the character string to be searched is stored (character string)
- Ⓢ<sub>2</sub> : Character string in which a search is performed or head number of the devices where the character string is stored (character string)
- ⓓ : Head number of the devices where the result of search will be stored (BIN 16 bits)
- n : Location to start the search (BIN 16 bits)

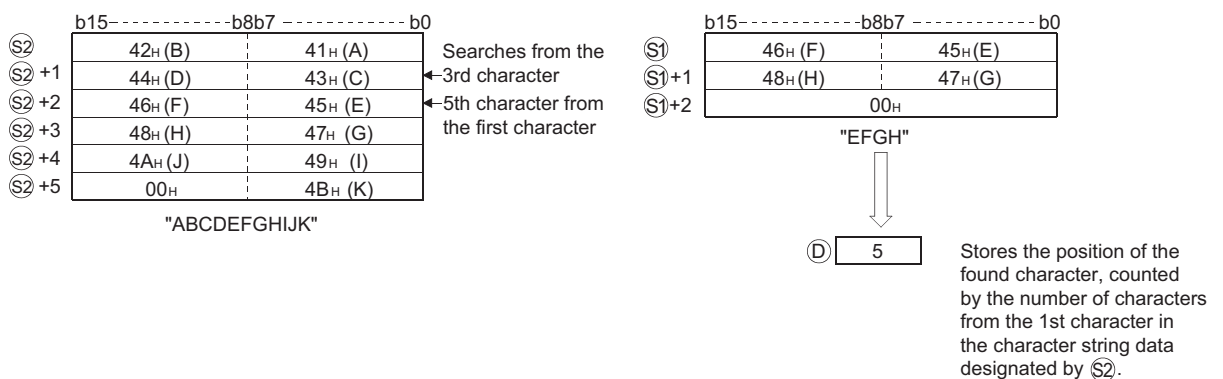
Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H	\$	
Ⓢ <sub>1</sub>	—	○				—		—	○	—
Ⓢ <sub>2</sub>	—	○				—		—	○	—
ⓓ	○	○				○		—	—	—
n	○	○				○		○	—	—

### ★ Function

- (1) Searches for the character string data designated by Ⓢ<sub>1</sub> in the area starting from the nth character from the left of the character string data designated by Ⓢ<sub>2</sub> and stores the result of search at the device designated by ⓓ.

As the result of search, the location of match, counted in the number of characters from the first character of the character string data designated by Ⓢ<sub>2</sub>, is stored.

When n = 3



- (2) If there is no matching character string data, stores "0" at ⓓ.

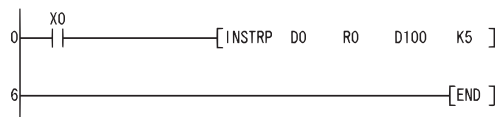
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of n exceeds the number of characters for ②. (Error code: 4100)
  - 00H (NULL) does not exist within the corresponding device range after the device designated by ①, ②. (Error code: 4100)
  - The value of n is negative number or "0". (Error code: 4100)

## Program Example

- (1) The following program searches from the 5th character from the left of the character string data stored in devices starting from R0 for the character string data in devices starting from D0, and stores the results at D100 when X0 goes ON.

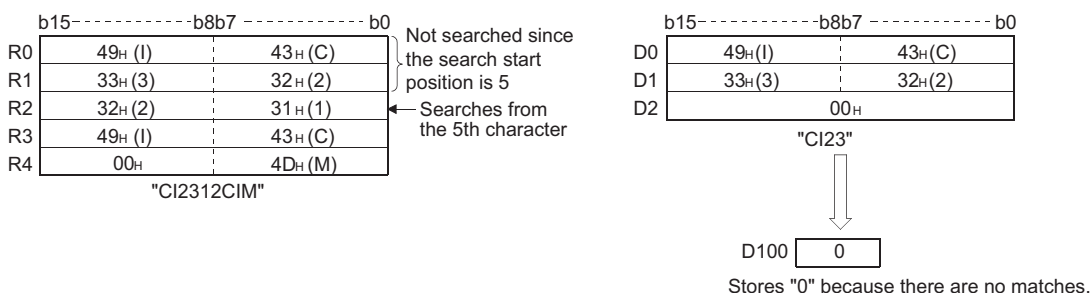
[Ladder Mode]



[List Mode]

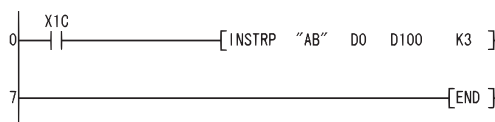
Step	Instruction	Device
0	LD	X0
1	INSTRP	D0 R0 D100 K5
6	END	

[Operation]



- (2) The following program searches from the 3rd character from the left of the character string data being stored in devices starting from D0 for the character string data "AB", and stores the results of the search at D100 when X1C goes ON.

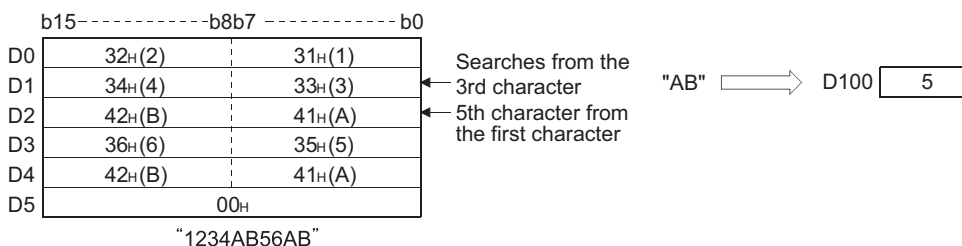
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	INSTRP	"AB" D0 D100 K3
7	END	

[Operation]



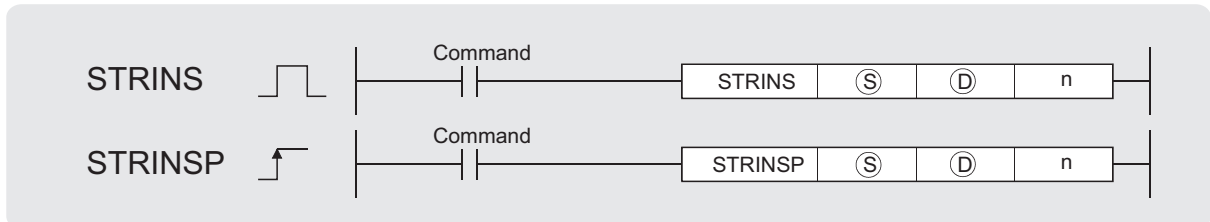


## 7.11.18 Insertion of character string (STRINS(P))



QnU(D)H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



(S) : Character string to be inserted or head number (character string) of the devices where insert character strings are stored

(D) : Head number (character string) of the devices where insert character strings are stored

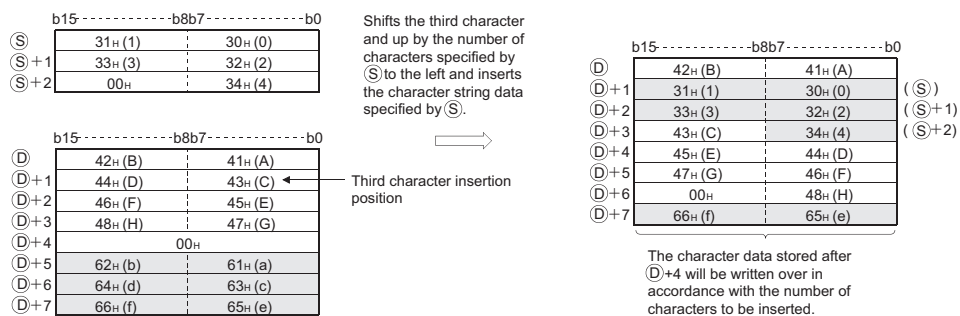
n : Insert position (Setting range:  $1 \leq n \leq 16383$ ) (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H	\$	
(S)	—	○				—		—	○	—
(D)	—	○				—		—	—	—
n	—	○				○		○	—	—

### ★ Function

- (1) This instruction inserts the character string data specified by (S) to the nth device (insert position) from the initial character string data stored in the devices specified by (D).

Insert position: n = 3



- (2) This instruction stores the NULL code (00H) into the device (1 word) that positions after the last device where the character string data are stored, if the character string ((S) + (D)) value is even after the insertion.
- (3) This instruction stores the NULL code (00H) into the last device (high 8 bits) where the character string data are stored, if the character string ((S) + (D)) value is odd after the insertion.
- (4) This instruction links the device, where the character string data are stored, specified by (S) with the last device specified by (D), if n is specified by the number of devices specified by (D) plus one.

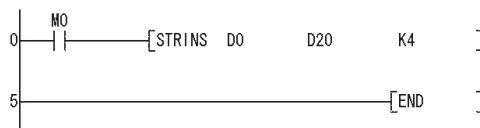
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The number of characters in the devices specified by  $\textcircled{S}$ ,  $\textcircled{D}$ , or the devices specified by  $(\textcircled{S} + \textcircled{D})$  after the insertion exceeds 16383 characters. (Error code: 4100)
  - The value specified by  $n$  is not within the specified range. ( $1 \leq n \leq 16383$ ) (Error code: 4100)
  - The value specified by  $n$  exceeds the number of the devices specified by  $\textcircled{D}$  plus one. (Error code: 4100)
  - The devices, that store character strings, specified by  $\textcircled{S}$  overlaps with even one of the devices specified by  $\textcircled{D}$ . (Error code: 4101)
  - The range of the devices specified by  $(\textcircled{S} + \textcircled{D})$  in which character strings data have been inserted exceeds the specified device range. (Error code: 4101)
  - The NULL code (00H) does not exist within the specified device range after the device specified by  $\textcircled{S}$  or  $\textcircled{D}$ . (Error code: 4101)
  - The range of the devices specified by  $(\textcircled{S} + \textcircled{D})$  in which character strings data have been inserted overlaps with the range of the devices specified by  $\textcircled{S}$  that store the character string data. (Error code: 4101)

## Program Example

- (1) The following program inserts the character string data stored in the device D0 and up to the fourth device from the initial character string data stored in D20 and up, when M0 is turned on.

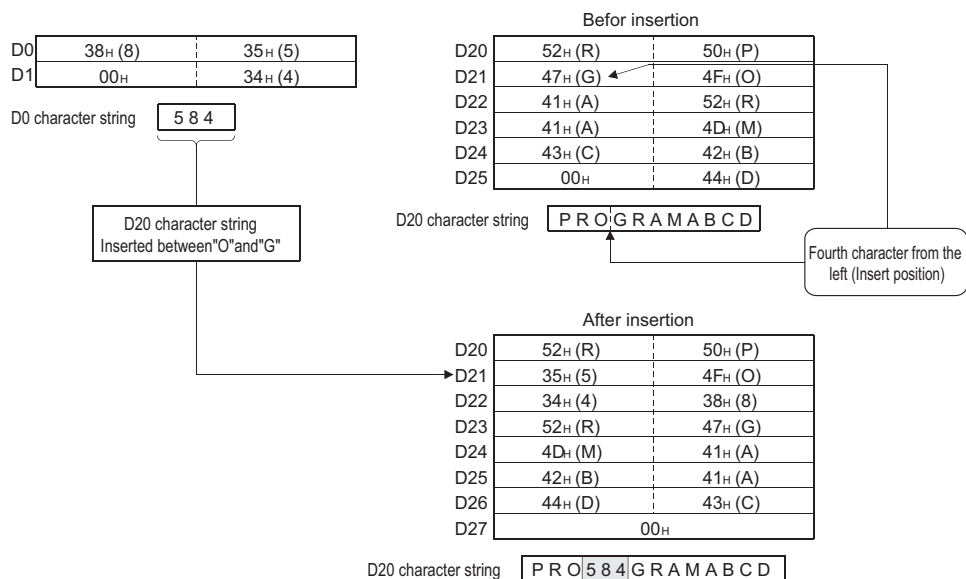
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	STRINS	D0 D20 K4
5	END	

[Operation]

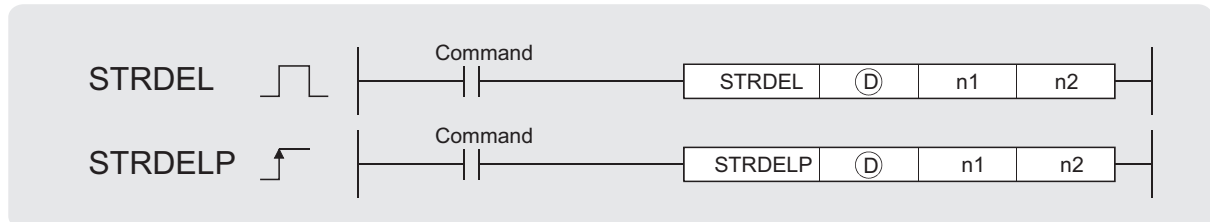


## 7.11.19 Deletion of character string (STRDEL(P))



QnU(D)H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓓ : Head number (character string) of the devices where character strings to be deleted are stored

n1 : Deletion start position (Setting range  $1 \leq n1 \leq 16383$ ) (BIN 16 bits)

n2 : Number of characters to be deleted (Setting range  $1 \leq n2 \leq 16384-n1$ ) (BIN 16 bits)

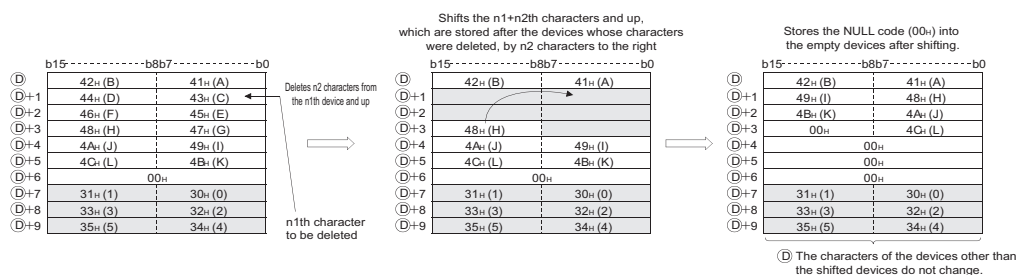
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H		
Ⓓ	—	○				—		—	—	
n1	—	○				○		○	—	
n2	—	○				○		○	—	

### Function

- This instruction deletes n2 characters data in the devices specified by Ⓓ starting from the device (insert position) specified by n1.

Device position where character string data to be deleted: n1 = 3

Number of characters to be deleted: n2 = 5



- This instruction stores the NULL code (00H) into the device (one word) that positions after the last device that stores the character string data when the character string data specified by Ⓓ is even, after the characters are deleted.
- This instruction stores the NULL code (00H) into the last device (high 8 bits) that stores the character string data when the character string data specified by Ⓓ is odd, after the characters are deleted.
- This instruction shifts the characters stored in the devices that position after the deleted devices by n2 characters to the right, and then stores the NULL code (00H) into the empty device.

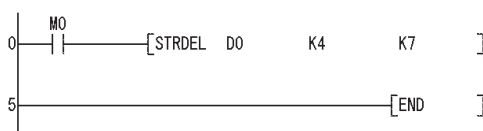
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The number of characters in the devices specified by ① exceeds 16383. (Error code: 4100)
  - The value specified by n1 is not within the range. ( $1 \leq n1 \leq 16383$ ) (Error code: 4100)
  - The value specified by n1 exceeds the number of characters in the devices specified by ①. (Error code: 4100)
  - The value specified by n2 exceeds the number of characters in the devices starting from n1th to the last devices position. (Error code: 4100)
  - The value specified by n2 is negative. (Error code: 4100)

## Program Example

- (1) The following program deletes the fourth to the seventh characters in the character string data stored in the devices D0 and up, when M0 is turned on.

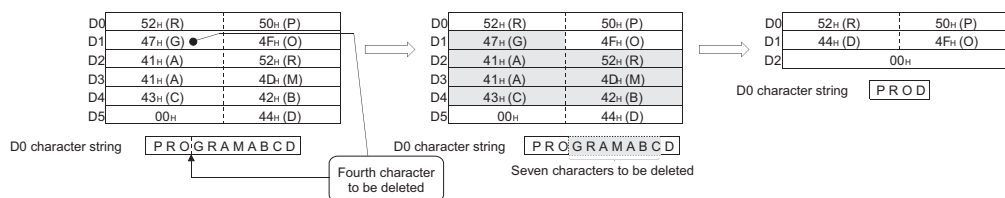
[Ladder Mode]



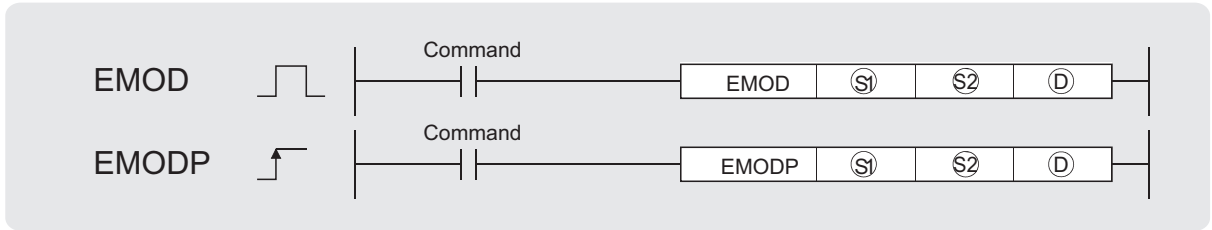
[List Mode]

Step	Instruction	Device
0	LD	M0
1	STRDEL	D0 K4 K7
5	END	

[Operation]



# 7.11.20 Floating decimal point to BCD (EMOD(P))



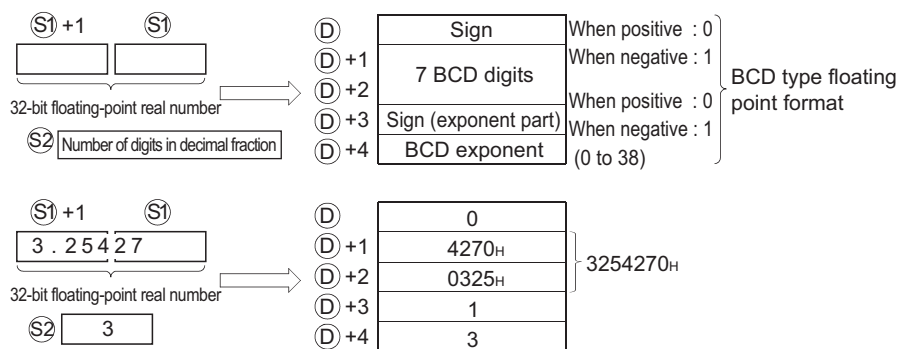
- Ⓢ1: 32-bit floating decimal point real number data or head number of the devices where the floating decimal point real number data is stored (real number)
- Ⓢ2: Decimal fraction digits data (BIN 16 bits)
- Ⓧ: Head number of the devices where the data after break down into BCD will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J0A0		U0A0G0	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H	E	
Ⓢ1	—	○	○	—	○	○*1	—	○	—	
Ⓢ2	○	○	○	○	○	○	○	—	—	
Ⓧ	—	○	○	—	—	—	—	—	—	

\*1: Available only in multiple Universal model QCPU

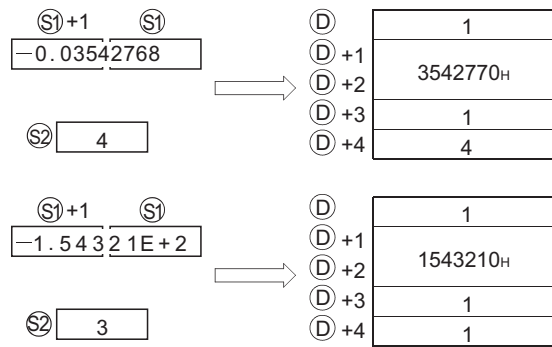
## ★ Function

- (1) Dissociate the 32-bit floating decimal point data designated by Ⓢ1 into BCD type floating point format based on the decimal fraction digits specified by Ⓢ2, and stores the result into the area starting from the device designated by Ⓧ.

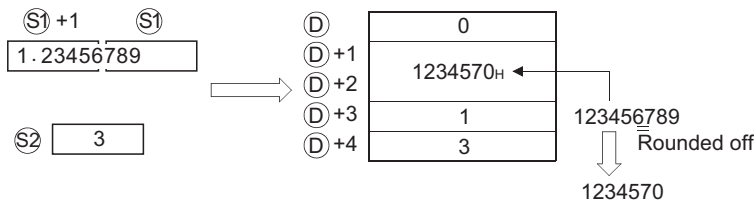


Ⓢ2 specifies the decimal fraction digits of the 32-bit floating decimal point real number data of Ⓢ1. In the example above, a decimal fraction digit is designated as shown below:

3.25427  
 ↑↑↑  
 Ⓢ2=3



- (2) The 7th digit of the significant digits being stored at  $\textcircled{D} + 1$  and  $\textcircled{D} + 2$  is rounded off to make a 6-digit number.



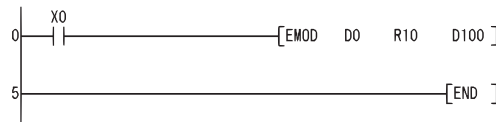
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The decimal fraction digit designated by  $\textcircled{S2}$  is outside the range of from 0 to 7.  
(Error code: 4100)
  - The device range designated by  $\textcircled{D}$  exceeds the range of the relevant device.  
(Error code: 4101)
  - The 32-bit floating point real number specified by  $\textcircled{S1}$  is outside the following range.  
 $0, 2^{-126} \leq | \text{device} | < 2^{128}$   
(Error code: 4100)
  - The device specified by  $\textcircled{D}$  exceeds the range of the corresponding device.  
(For the Universal model QCPU only.)  
(Error code: 4101)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
(For the Universal model QCPU only)  
(Error code: 4140)

## Program Example

- (1) The following program breaks down the 32-bit floating decimal point type real number data stored at D0 and D1 into BCD according to the decimal fraction digits as designated by R10, and stores the results into the area starting from D100 when X0 is turned ON.

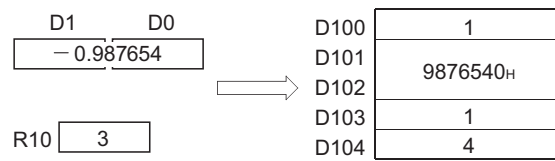
[Ladder Mode]



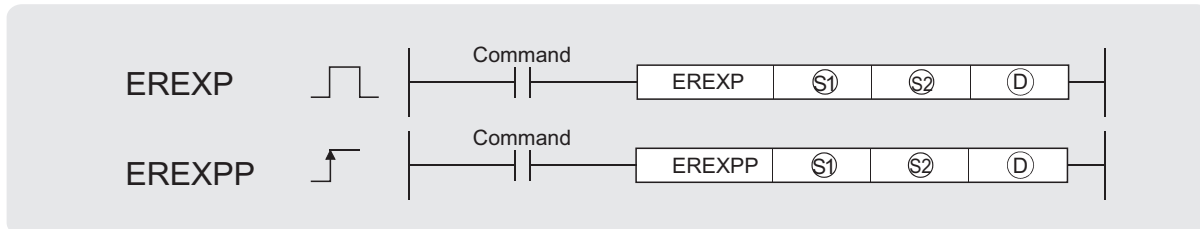
[List Mode]

Step	Instruction	Device
0	LD	X0
1	EMOD	D0 R10 D100
5	END	

[Operation]



# 7.11.21 From BCD format data to floating decimal point (EREXP(P))



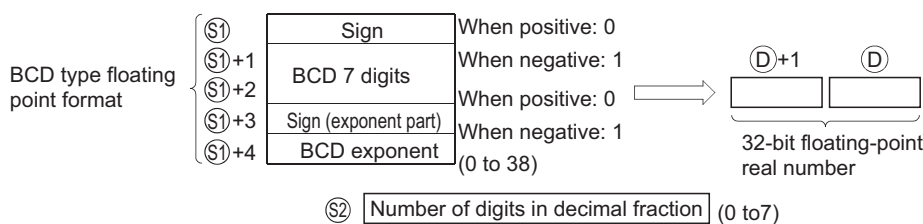
- Ⓢ1: Head number of the devices where BCD type floating point format data is stored (BIN 16 bits)
- Ⓢ2: Decimal fraction digits data (BIN 16 bits)
- Ⓧ: The device where the converted 32-bit floating point real number data will be stored (real number)

Setting Data	Internal Devices		R, ZR	JDO		UAGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○	—	—	—	—	—	—	—
Ⓢ2	○	○	○	○	○	○	○	○	—
Ⓧ	—	○	—	—	○	○	○*1	—	—

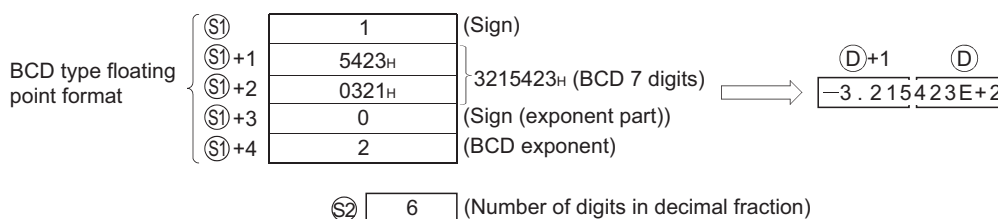
\*1: Available only in multiple Universal model QCPU

## ★ Function

- (1) Converts the BCD type floating point data designated by Ⓢ1 to the 32-bit floating decimal point real number data according to the decimal fraction digits specified by Ⓢ2, and stores the result into the area starting from the device designated by Ⓧ.



- (2) The sign at Ⓢ1 and the sign for the exponent part at Ⓢ1+3 is set at 0 for a positive value and at 1 for a negative value.
- (3) 0 to 38 can be set for the BCD exponent of Ⓢ1+4.
- (4) 0 to 7 can be set for the decimal fraction digits of Ⓢ2.





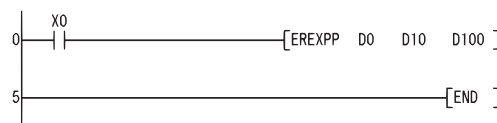
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The format designated by (S1) was neither 0 nor 1. (Error code: 4100)
  - A value other than 0 to 9 exists in the each digit of (S1) + 1 and (S1) + 2. (Error code: 4100)
  - The format designation made by (S1) + 3 is something other than 0 or 1. (Error code: 4100)
  - The exponent data designated by (S1) + 4 is outside the range of from 0 to 38. (Error code: 4100)
  - The decimal fraction digit designated by (S2) is outside the range of from 0 to 7. (Error code: 4100)
  - The device specified by (S1) exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program converts the BCD type floating decimal point format data being stored in devices starting from D0 to 32-bit floating decimal point type real number data based on the decimal fraction digit being stored at D10, and stores the result at D100 and D101 when X0 goes ON.

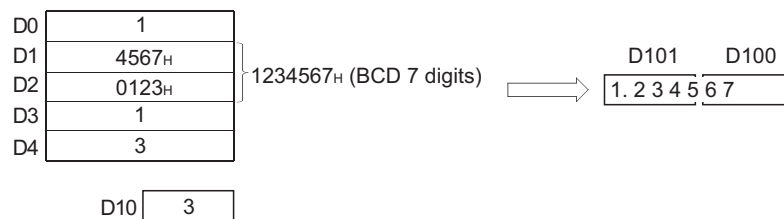
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	EREXPP	D0 D10 D100
5	END	

[Operation]

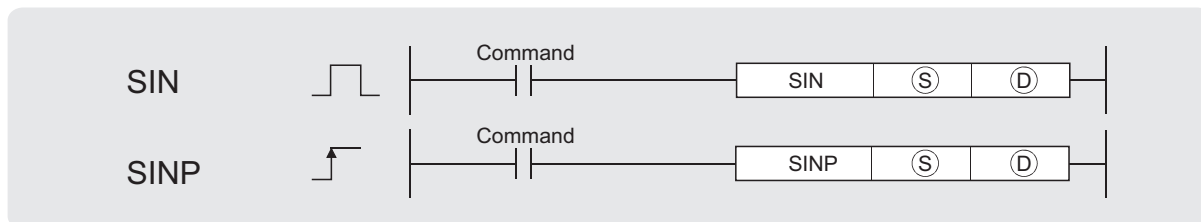


## 7.12 Special function instructions

### 7.12.1 SIN operation on floating-point data (Single precision) (SIN(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Angle data of which the SIN (sine) value is obtained or head number of the devices where the angle data is stored (real number)

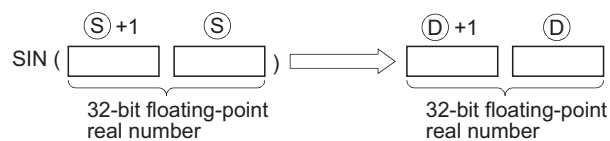
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J00		U000	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○		—	○		○*1	○	—
Ⓣ	—	○		—	○		○*1	—	—

\*1: Applicable for the Universal model QCPU only.

## ★ Function

- (1) Returns the SIN (sine) value of the angle designated at Ⓢ and stores the operation result in the device number designated at Ⓣ.



- (2) Angles designated at Ⓢ are set in radian units ( $\text{degrees} \times \pi / 180$ ).  
For conversion between degrees and radian values, see the RAD and DEG instructions.

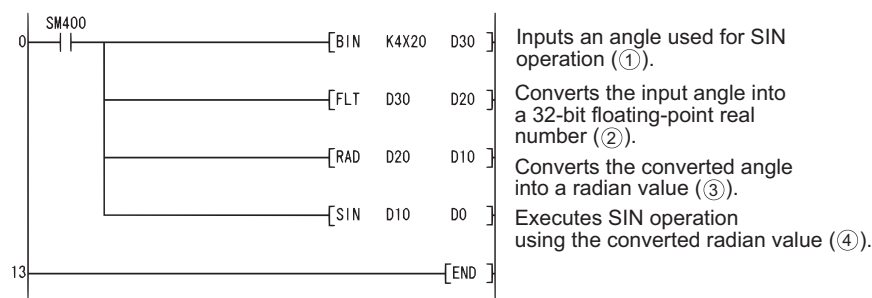
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is  $-0.^{*2}$   
(For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)  
\*2: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
  - The result exceeds the following range (Operation results in an overflow)  
(For the Universal model QCPU only)  
 $2^{128} \cong | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
(For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program conducts a SIN operation on the angles stored in the four BCD digits from X20 to X2F and stores the results at D0 and D1 as 32-bit floating decimal point type real numbers.

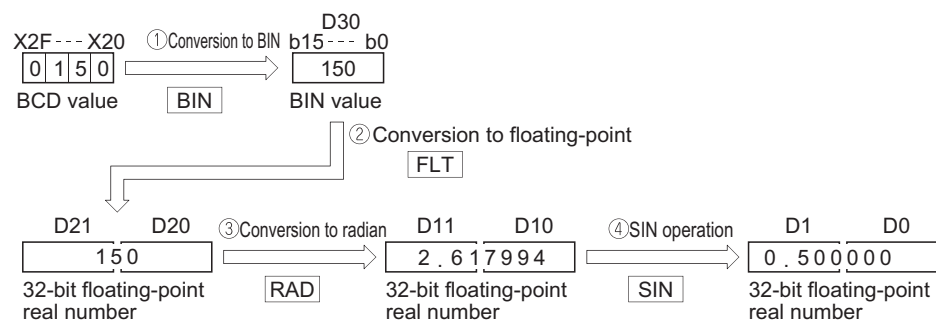
[Ladder Mode]



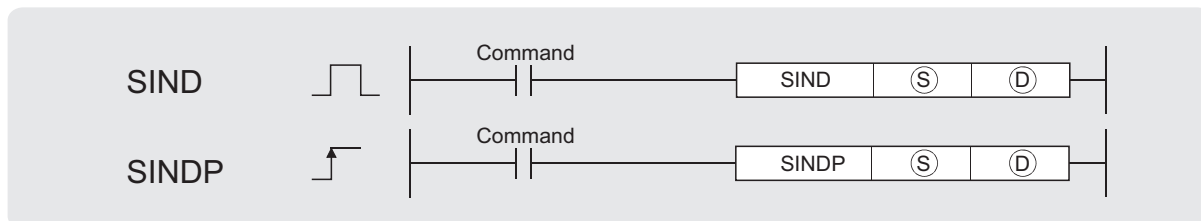
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D30
4	FLT	D30 D20
7	RAD	D20 D10
10	SIN	D10 D0
13	END	

[Operations involved when X20 to X2F designate a value of 150]



## 7.12.2 SIN operation on floating-point data (Double precision) (SIND(P))



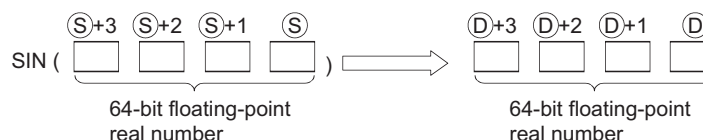
Ⓢ : Angle data of which the SIN (sine) value is obtained or head number of the devices where the angle data is stored (real number)

Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JDO		UOGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- The SIN (sine) value of the angle specified by Ⓢ is calculated and its result is stored into the device specified by Ⓣ.



- Angles designated at Ⓢ are set in radian units (degrees  $\times \pi / 180$ ).  
For conversion between degrees and radian values, see the RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.

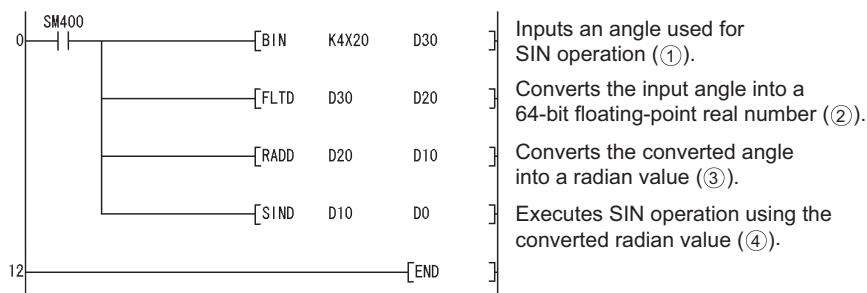
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Operation result} |$  (Error code: 4141)

## Program Example

- (1) The following program conducts a SIN operation on the angles stored in the four BCD digits from X20 to X2F and stores the results at D0 to D3 as 64-bit floating decimal point type real numbers.

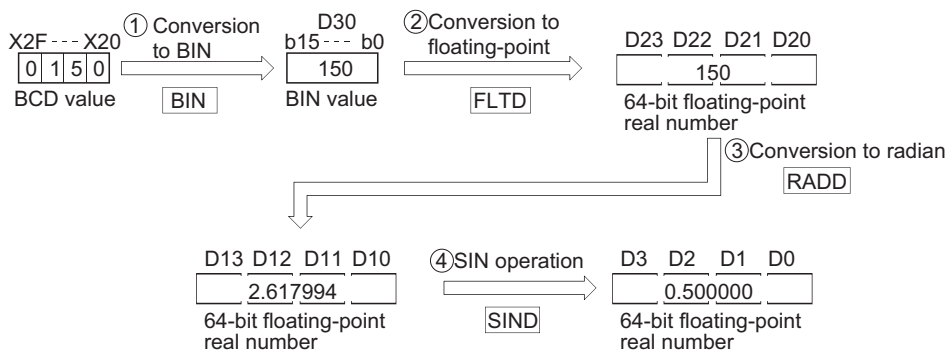
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D30
3	FLTD	D30 D20
6	RADD	D20 D10
9	SIND	D10 D0
12	END	

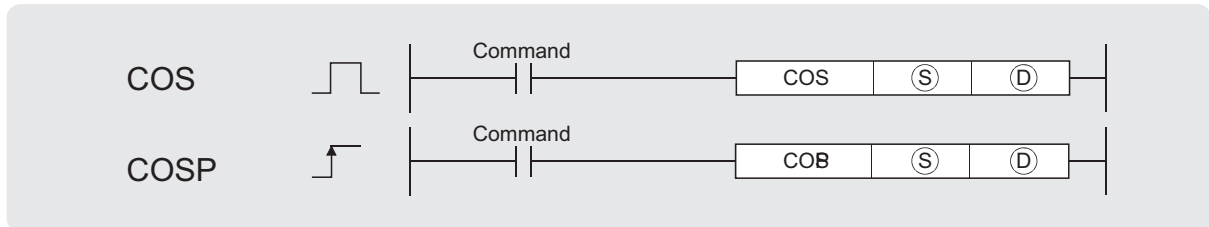
[Operations involved when X20 to X2F designate a value of 150]



## 7.12.3 COS operation on floating-point data (Single precision) (COS(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Angle data of which the COS (cosine) value is obtained or head number of the devices where the angle data is stored (real number)

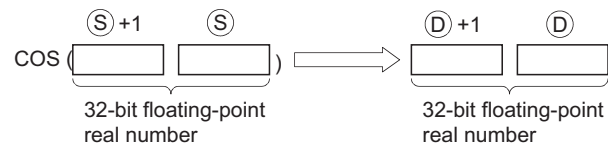
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JGO		UAGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○*1	○	—	—	
Ⓣ	—	○	—	○	○*1	○	—	—	

\*1: Applicable for the Universal model QCPU only.

### ★ Function

- Returns the COS (cosine) value of the angle designated by Ⓢ and stores operation result at device number designated by Ⓣ.



- Angles designated at Ⓢ are set in radian units ( $\text{degrees} \times \pi / 180$ ).  
For conversion between degrees and radian values, see the RAD and DEG instructions.

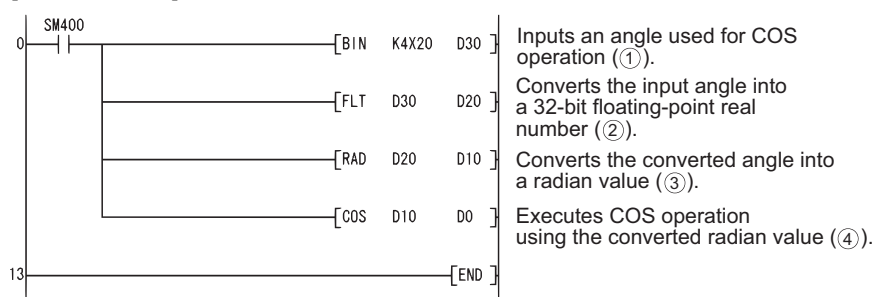
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is  $-0.$  \*2  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)  
 (Error code: 4100)
  - \*2: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
  - The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$   
 (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only)  
 (Error code: 4140)

## Program Example

- (1) The following program performs a COS operation on the angle data designated by the 4 BCD digits from X20 to X2F, and stores results as 32-bit floating decimal point type real numbers at D0 and D1.

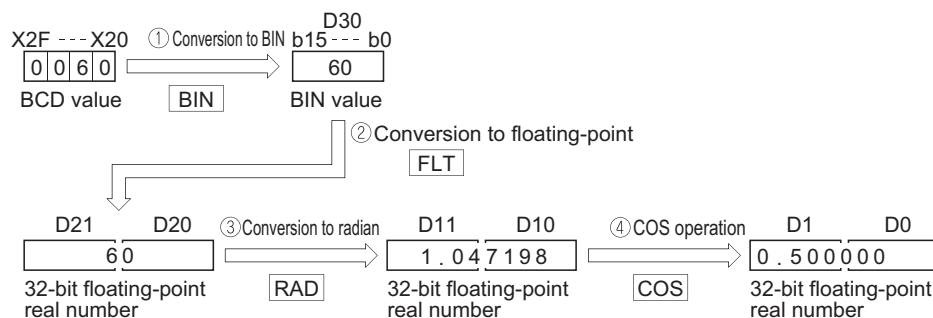
[Ladder Mode]



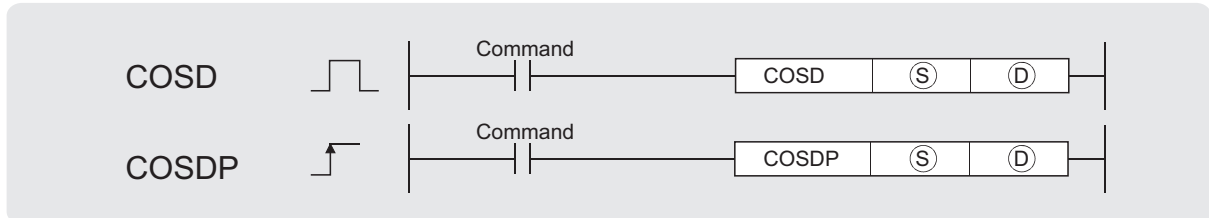
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D30
4	FLT	D30 D20
7	RAD	D20 D10
10	COS	D10 D0
13	END	

[Operations involved when X20 to X2F designate a value of 60]



## 7.12.4 COS operation on floating-point data (Double precision) (COSD(P))



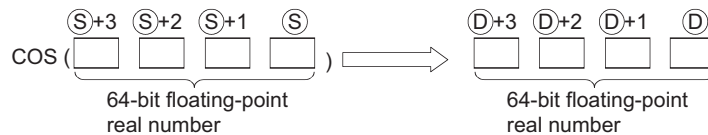
Ⓢ : Angle data of which the COS (cosine) value is obtained or head number of the devices where the angle data is stored (real number)

ⓓ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
ⓓ	—	○				—		—	—

### ★ Function

- The COS (cosine) value of the angle specified by Ⓢ is calculated and its result is stored into the device specified by ⓓ.



- Angles designated at Ⓢ are set in radian units (degrees  $\times \pi / 180$ ).  
For conversion between degrees and radian values, see the RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.

### ! Operation Error

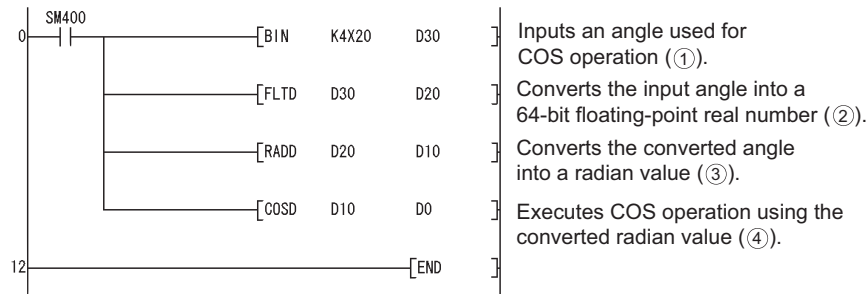
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \cong | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \cong | \text{Operation result} |$



# Program Example

- (1) The following program performs a COS operation on the angle data designated by the 4 BCD digits from X20 to X2F, and stores results as 64-bit floating decimal point type real numbers at D0 to D3.

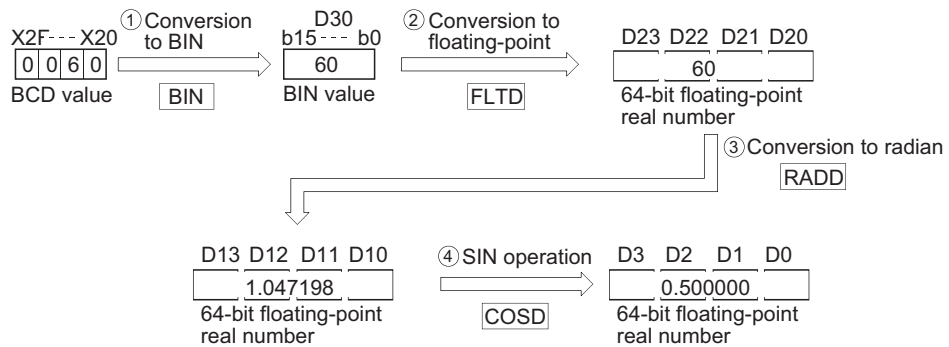
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D30
3	FLTD	D30 D20
6	RADD	D20 D10
9	COSD	D10 D0
12	END	

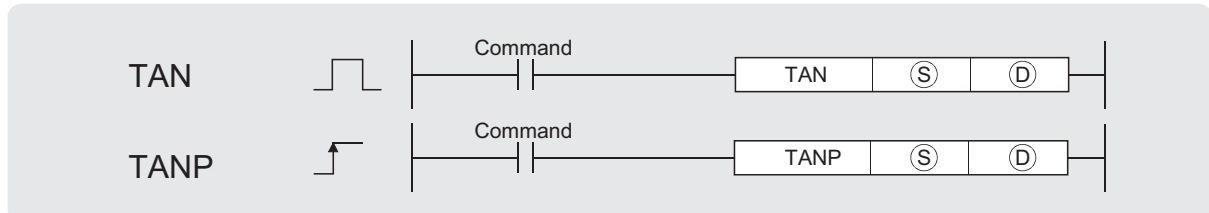
[Operations involved when X20 to X2F designate a value of 60]



## 7.12.5 TAN operation on floating-point data (Single precision) (TAN(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Angle data of which the TAN (tangent) value is obtained or head number of the devices where the angle data is stored (real number)

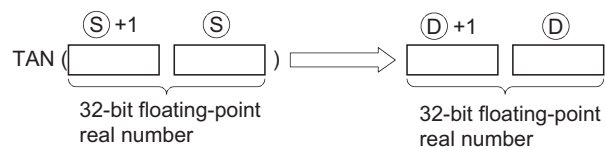
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	—	○	○ <sup>*2</sup>	○	—	
Ⓣ	—	○	—	—	○	○ <sup>*2</sup>	—	—	

\*2: Applicable for the Universal model QCPU only.

### ★ Function

- Returns the tangent (TAN) value of the angle data designated by Ⓢ, and stores operation result in device designated by Ⓣ.



- Angles designated at Ⓢ are set in radian units ( $\text{degrees} \times \pi / 180$ ).  
For conversion between degrees and radian values, see the RAD and DEG instructions.
- When angles designated by Ⓢ are  $\pi/2$  radians, or  $(3/2)\pi$  radians, an operation error will be generated in the calculation of the radian value, so care must be taken to avoid such errors.

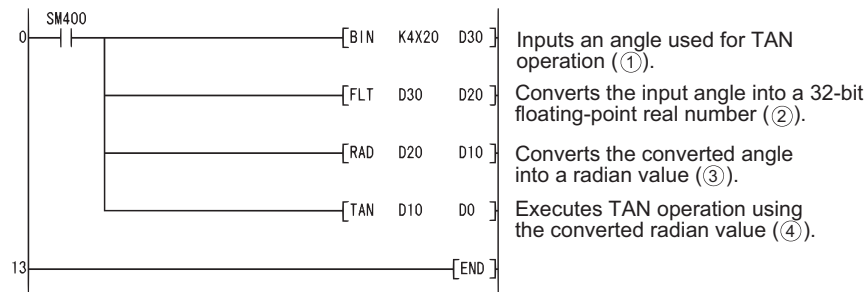
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Operation results are outside the range shown below:  
 $0, 2^{-126} \leq |\text{operation result}| < 2^{128}$   
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
  - The value of the specified device is  $-0$ . \*3  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
- \*3: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
- The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq |\text{Operation result}|$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program performs a TAN operation on the angle data set by the 4 BCD digits from X20 to X2F, and stores the results as 32-bit floating decimal point type real numbers at D0 and D1.

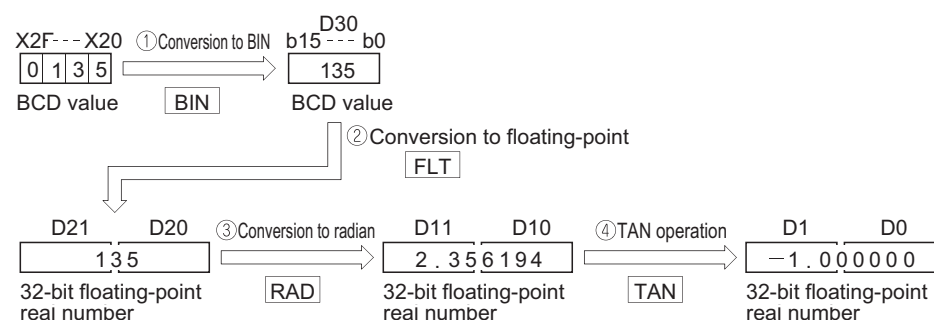
[Ladder Mode]



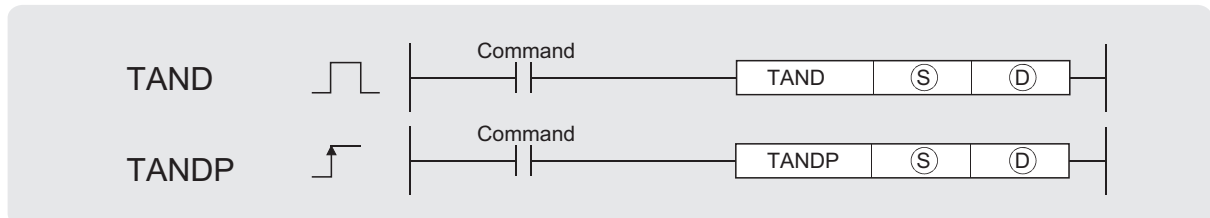
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D30
4	FLT	D30 D20
7	RAD	D20 D10
10	TAN	D10 D0
13	END	

[Operations involved when X20 to X2F designate a value of 135]



## 7.12.6 TAN operation on floating-point data (Double precision) (TAND(P))



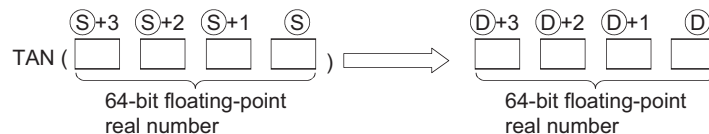
Ⓢ : Angle data of which the TAN (tangent) value is obtained or head number of the devices where the angle data is stored (real number)

ⓓ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JAD		UAGD	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○						○	—
ⓓ	—	○						—	—

### ★ Function

- (1) The TAN (tangent) value of the angle specified by Ⓢ is calculated and its result is stored into the device specified by ⓓ.



- (2) Angles designated at Ⓢ are set in radian units (degrees  $\times \pi / 180$ ).  
For conversion between degrees and radian values, see the RADD and DEGD instructions.
- (3) When angles designated by Ⓢ are  $\pi/2$  radians, or  $(3/2)\pi$  radians, an operation error will be generated in the calculation of the radian value, so care must be taken to avoid such errors.
- (4) When the operation results in -0 or an underflow, the result is processed as 0.

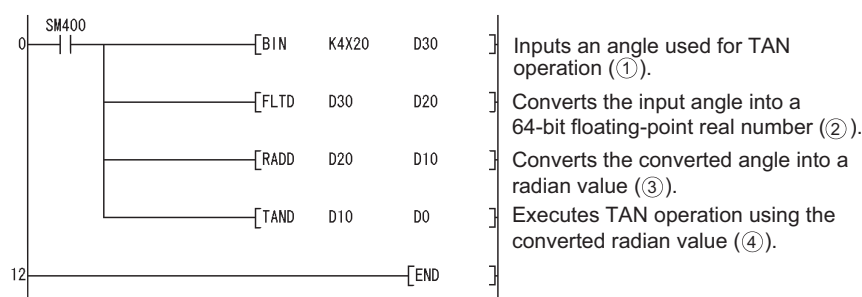
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Operation result} |$  (Error code: 4141)

## Program Example

- (1) The following program performs a TAN operation on the angle data set by the 4 BCD digits from X20 to X2F, and stores the results as 64-bit floating decimal point type real numbers at D0 to D3.

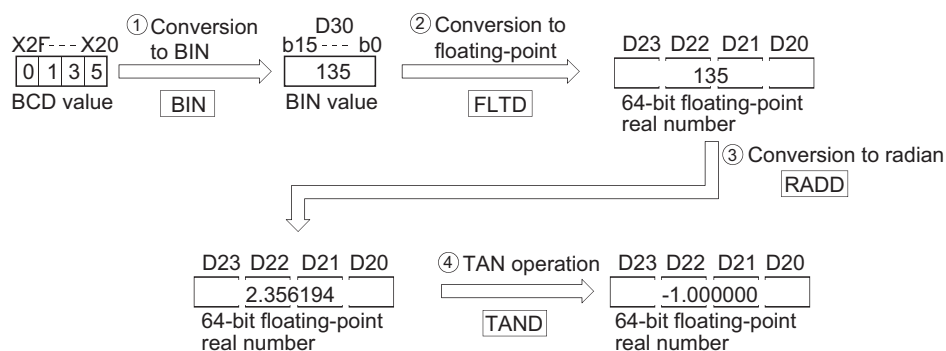
[Ladder Mode]



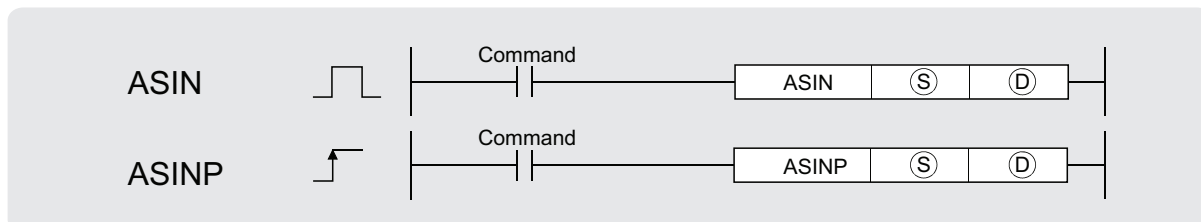
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D30
3	FLTD	D30 D20
6	RADD	D20 D10
9	TAND	D10 D0
12	END	

[Operations involved when X20 to X2F designate a value of 135]



## 7.12.7 SIN<sup>-1</sup> operation on floating point data (Single precision) (ASIN(P))



Ⓢ : SIN value of which the SIN<sup>-1</sup> (inverse sine) value is obtained or head number of the devices where the SIN value is stored (real number)

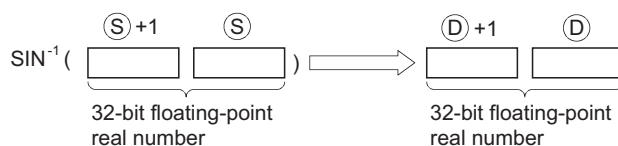
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	—	○	○*1	○	—	
Ⓣ	—	○	—	—	○	○*1	—	—	

\*1: Applicable for the Universal model QCPU only.

### ★ Function

- (1) Returns the SIN<sup>-1</sup> angle of the SIN value designated by Ⓢ, and stores operation results at word device designated by Ⓣ.



- (2) The SIN value designated by Ⓢ can be in the range from  $-1.0$  to  $1.0$ .
- (3) The angle (operation result) stored at Ⓣ is stored in radian units.  
For more information on the conversion between radian and angle data, see description of RAD and DEG instructions.

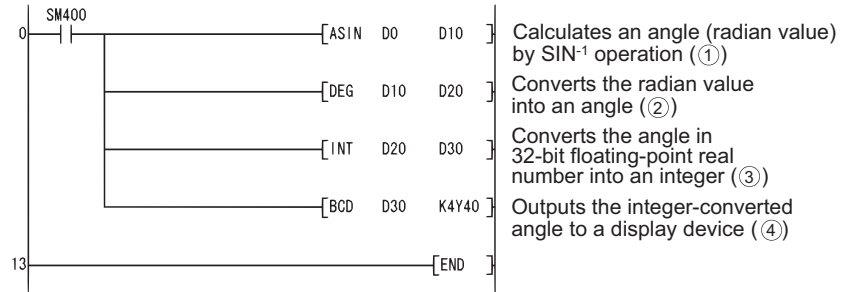
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value designated by  $\textcircled{S}$  is outside the range of from  $-1.0$  to  $1.0$ . (Error code: 4100)
  - The contents of the designated device or the result of the addition are not "0", or not within the following range(For the Universal model QCPU only): (Error code: 4140)  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$
  - The value of the specified device is  $-0$ . \*2  
 (For the High Performance model QCPU, Process CPU, Redundant CPU)  
 (Error code: 4100)  
 \*2: There are CPU modules that will not result in an operation error if  $-0$  is specified. Refer to 3.2.4 for details.
  - The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program seeks the inverse sine of the 32-bit floating decimal point real number at D0 and D1, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

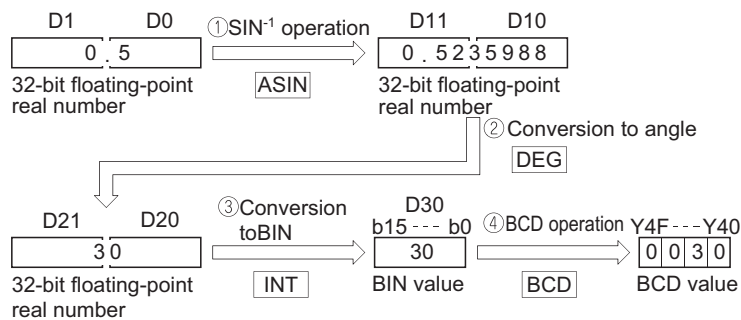
[Ladder Mode]



[List Mode]

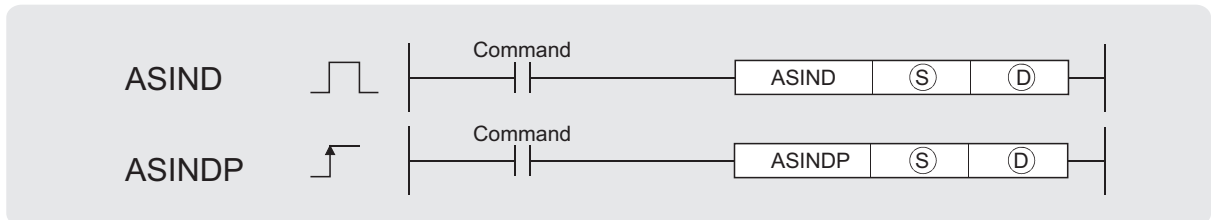
Step	Instruction	Device
0	LD	SM400
1	ASIN	D0 D10
4	DEG	D10 D20
7	INT	D20 D30
10	BCD	D30 K4Y40
13	END	

[Operations involved when the D0 and D1 value is 0.5]





## 7.12.8 $\text{SIN}^{-1}$ operation on floating-point data (Double precision) (ASIND(P))



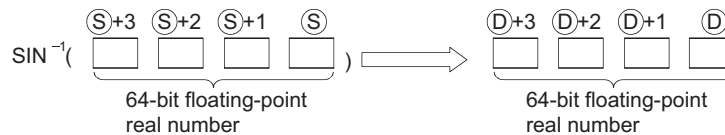
Ⓢ : SIN value of which the  $\text{SIN}^{-1}$  (inverse sine) value is obtained or head number of the devices where the SIN value is stored (real number)

ⓓ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
ⓓ	—	○				—		—	—

### ★ Function

- The angle is calculated from the SIN (sine) value specified by Ⓢ is and its result is stored into the device specified by ⓓ.



- The SIN value designated by Ⓢ can be in the range from  $-1.0$  to  $1.0$ .
- The angle (operation result) stored at ⓓ is stored in radian units.  
For more information on the conversion between radian and angle data, see description of RADD and DEGD instructions.
- When the operation results in  $-0$  or an underflow, the result is processed as  $0$ .

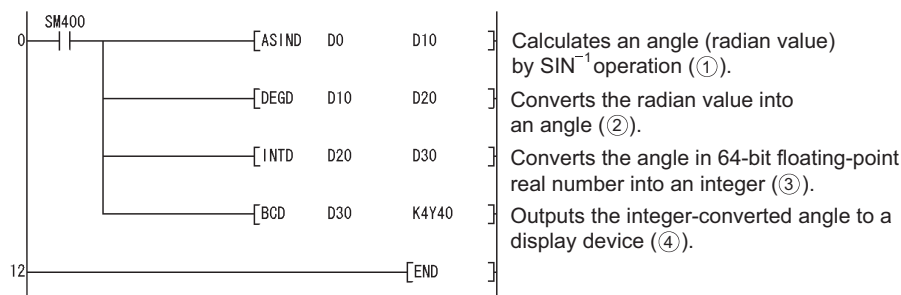
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The value specified by  $\textcircled{S}$  is within the double-precision floating-point range and outside the range of  $-1.0$  to  $1.0$ . (Error code: 4100)
  - The result exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Operation result} |$  (Error code: 4141)

## Program Example

- (1) The following program seeks the inverse sine of the 64-bit floating decimal point real number at D0 to D3, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

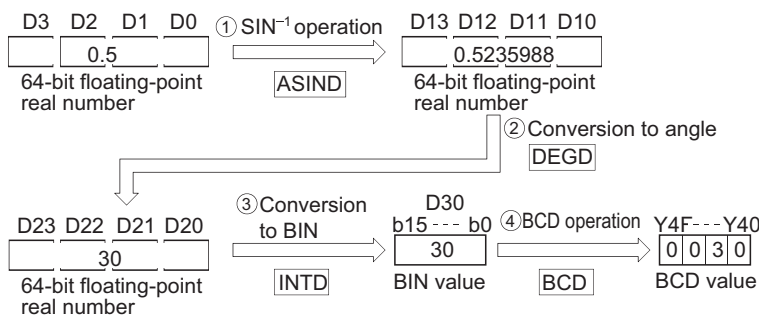
[Ladder Mode]



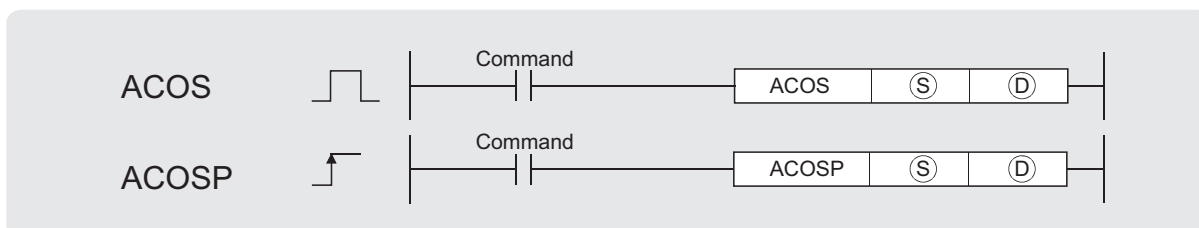
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	ASIND	D0 D10
4	DEGD	D10 D20
7	INTD	D20 D30
10	BCD	D30 K4Y40
12	END	

[Operations involved when the D0 to D3 value is 0.5]



## 7.12.9 $\text{COS}^{-1}$ operation on floating-point data (Single precision) (ACOS(P))



Ⓢ : COS value of which the  $\text{COS}^{-1}$  (inverse cosine) value is obtained or head number of the devices where the COS value is stored (real number)

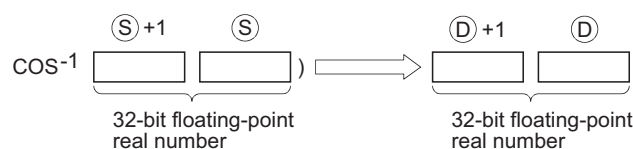
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:R:Z		U:G:Q	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○ <sup>*1</sup>	○	—	—	
Ⓣ	—	○	—	○	○	○ <sup>*1</sup>	—	—	

\*1: Applicable for the Universal model QCPU only.

### ★ Function

- Returns the  $\text{COS}^{-1}$  angle of the COS value designated by Ⓢ, and stores operation result at word device designated by Ⓣ.



- The COS value designated by Ⓢ can be in the range of from  $-1.0$  to  $1.0$ .
- The angle (operation result) stored at Ⓣ is stored in radian units.  
For more information on the conversion between radian and angle data, see description of RAD and DEG instructions.

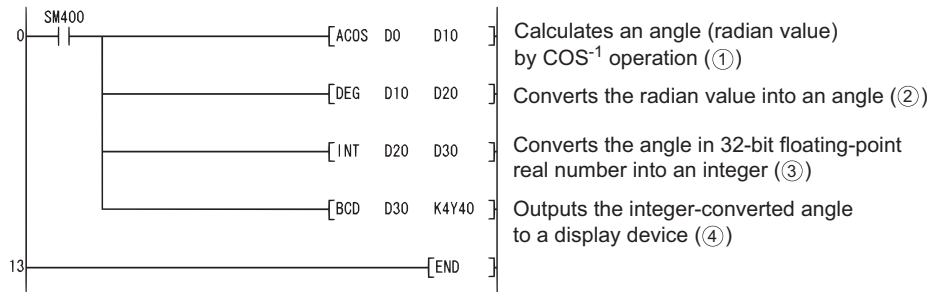
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value designated by ⑤ is outside the range of from  $-1.0$  to  $1.0$ . (Error code: 4100)
  - The contents of the designated device or the result of the addition are not "0", or not within the following range(For the Universal model QCPU only): (Error code: 4140)  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$
  - The value of the specified device is  $-0$ . \*2  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)  
 \*2: There are CPU modules that will not result in an operation error if  $-0$  is specified. Refer to 3.2.4 for details.
  - The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program seeks the inverse cosine of the 32-bit floating decimal point real number at D0 and D1, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

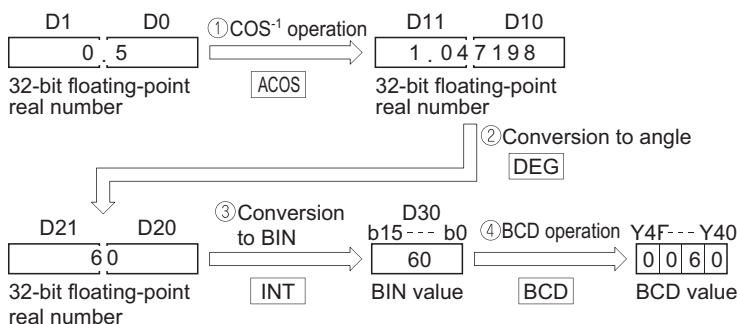
[Ladder Mode]



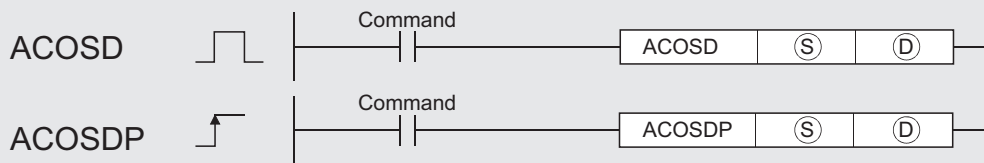
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	ACOS	D0 D10
4	DEG	D10 D20
7	INT	D20 D30
10	BCD	D30 K4Y40
13	END	

[Operations involved when the D0 and D1 value is 0.5]



## 7.12.10 $\text{COS}^{-1}$ operation on floating-point data (Double precision) (ACOSD(P))



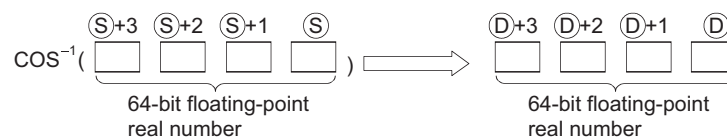
Ⓢ : COS value of which the  $\text{COS}^{-1}$  (inverse cosine) value is obtained or head number of the devices where the COS value is stored (real number)

Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:REG		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- The angle is calculated from the COS (cosine) value specified by Ⓢ is and its result is stored into the device specified by Ⓣ.



- The COS value designated by Ⓢ can be in the range of from  $-1.0$  to  $1.0$ .
- The angle (operation result) stored at Ⓣ is stored in radian units.  
For more information on the conversion between radian and angle data, see description of RADD and DEGD instructions.
- When the operation results in  $-0$  or an underflow, the result is processed as  $0$ .



## Operation Error

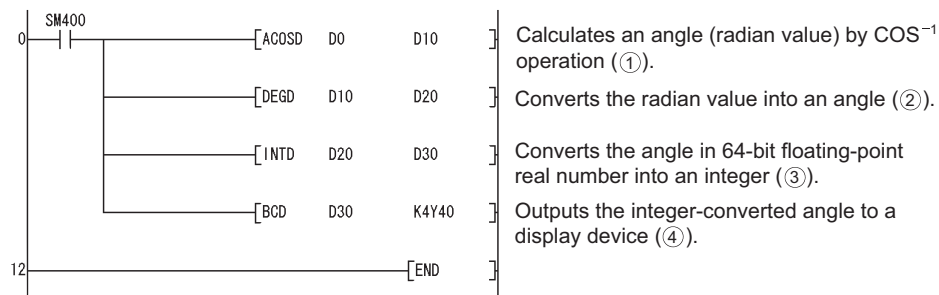
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The value specified by (S) is within the double-precision floating-point range and outside the range of  $-1.0$  to  $1.0$ . (Error code: 4100)
  - The result exceeds the following range (Operation results in an overflow):  
 $2^{1024} \leq | \text{Operation result} |$  (Error code: 4141)



## Program Example

- (1) The following program seeks the inverse cosine of the 64-bit floating decimal point real number at D0 to D3, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

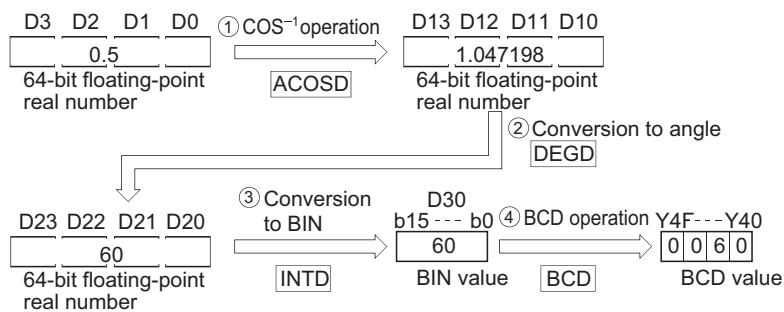
[Ladder Mode]



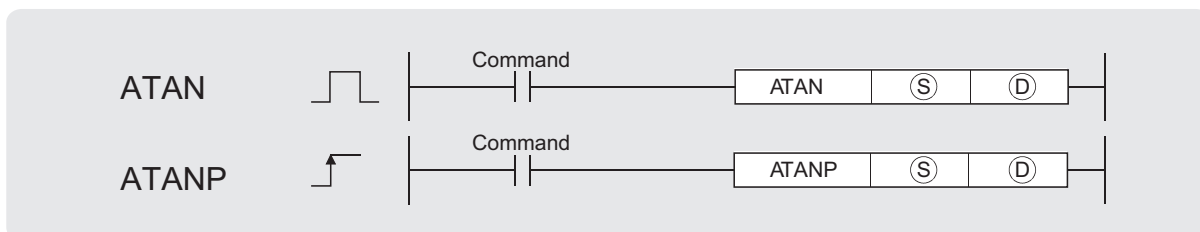
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	ACOSD	D0 D10
4	DEGD	D10 D20
7	INTD	D20 D30
10	BCD	D30 K4Y40
12	END	

[Operations involved when the D0 to D3 value is 0.5]



# 7.12.11 TAN<sup>-1</sup> operation on floating-point data (Single precision) (ATAN(P))



Ⓢ : TAN value of which the TAN<sup>-1</sup> (inverse tangent) value is obtained or head number of the devices where the TAN value is stored (real number)

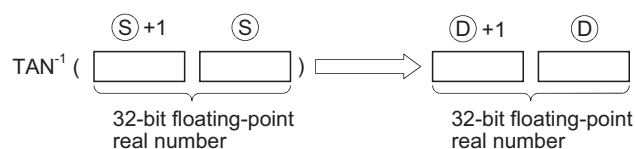
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○ <sup>*1</sup>	○	—	—	
Ⓣ	—	○	—	○	○ <sup>*1</sup>	—	—	—	

\*1: Applicable for the Universal model QCPU only.

## ★ Function

- Returns the TAN<sup>-1</sup> angle of the TAN value designated by Ⓢ, and stores operation results at word device designated by Ⓣ.



- The angle (operation result) stored at Ⓣ is stored in radian units.  
For more information on the conversion between radian and angle data, see description of RAD and DEG instructions.

7.12 Special function instructions  
7.12.11 TAN<sup>-1</sup> operation on floating-point data (Single precision) (ATAN(P))

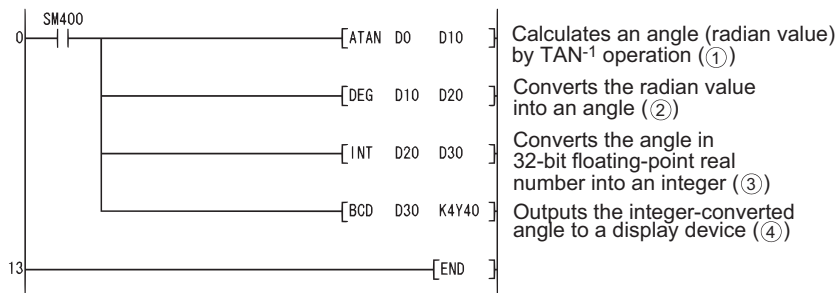
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range(For the Universal model QCPU only): (Error code: 4140)  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$
  - The value of the specified device is  $-0$ . \*2  
 (For the High Performance model QCPU, Process CPU, Redundant CPU)  
 (Error code: 4100)
- \*2: There are CPU modules that will not result in an operation error if  $-0$  is specified. Refer to 3.2.4 for details.
- The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program seeks the inverse tangent of the 32-bit floating decimal point real number at D0 and D1, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

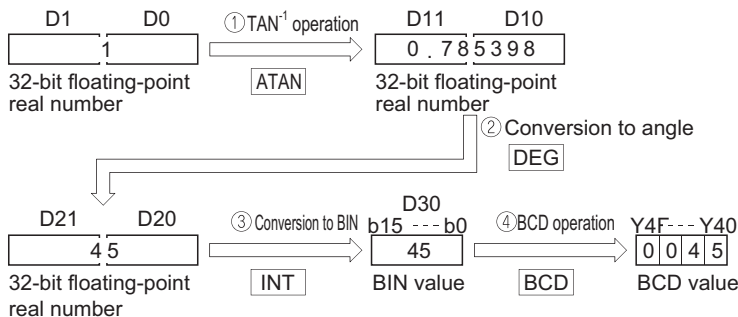
[Ladder Mode]



[List Mode]

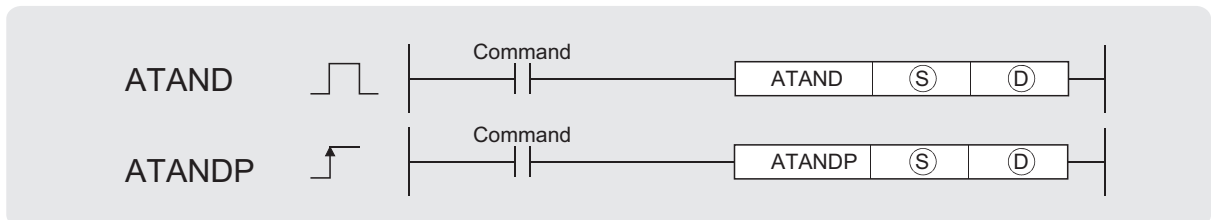
Step	Instruction	Device
0	LD	SM400
1	ATAN	D0 D10
4	DEG	D10 D20
7	INT	D20 D30
10	BCD	D30 K4Y40
13	END	

[Operations involved when D0 and D1 value is 1]





# 7.12.12 TAN<sup>-1</sup> operation on floating-point data (Double precision) (ATAND(P))

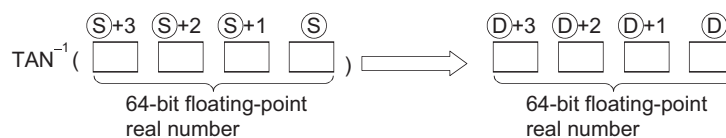


Ⓢ : TAN value of which the TAN<sup>-1</sup> (inverse tangent) value is obtained or head number of the devices where the TAN value is stored (real number)  
 Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

## ★ Function

- The angle is calculated from the TAN (tangent) value specified by Ⓢ and its result is stored into the device specified by Ⓣ.



- The angle (operation result) stored at Ⓣ is stored in radian units. For more information on the conversion between radian and angle data, see description of RADD and DEGD instructions.
- When the operation results in -0 or an underflow, the result is processed as 0.

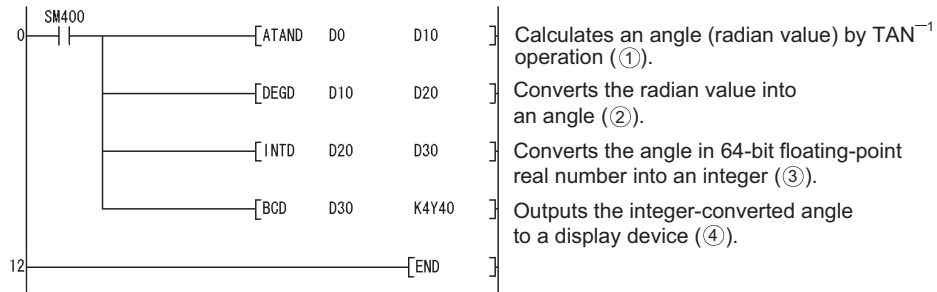
## ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \leq | \text{Operation result} |$

## Program Example

- (1) The following program seeks the inverse tangent of the 64-bit floating decimal point real number at D0 to D3, and outputs the angle to the 4 BCD digits at Y40 to Y4F.

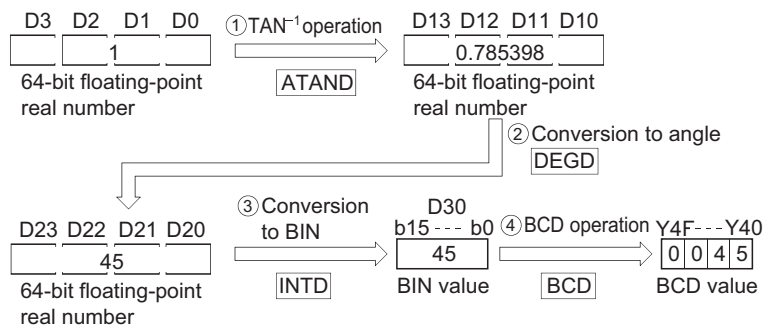
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	ATAND	D0 D10 D11
4	DEGD	D10 D20 D21
7	INTD	D20 D30 D31
10	BCD	D30 K4Y40
12	END	

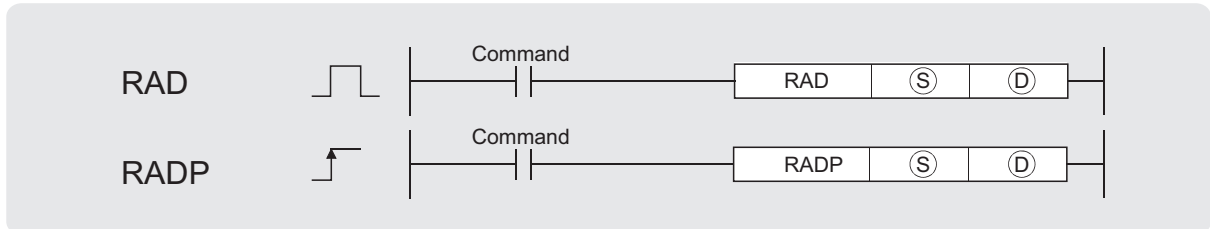
[Operations involved when D0 to D3 value is 1]



# 7.12.13 Conversion from floating-point angle to radian (Single precision) (RAD(P))

Ver. Basic High performance Process Redundant Universal

Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Angle to be converted to radian units or head number of the devices where the angle is stored (real number)  
 Ⓣ : Head number of the devices where the value converted in radian units will be stored (real number)

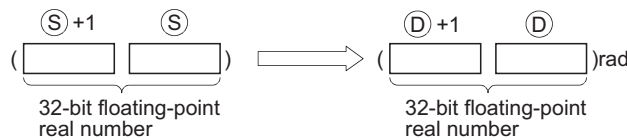
Setting Data	Internal Devices		R, ZR	J0A0		U0A0	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	—	○	○ <sup>*2</sup>	○	—	
Ⓣ	—	○	—	—	○	○ <sup>*2</sup>	—	—	

\*2: Applicable for the Universal model QCPU only.

7

## ★ Function

- Converts units of angle size from angle units designated by Ⓢ to radian units, and stores result at device number designated by Ⓣ.



- Conversion from degree to radian units is performed according to the following equation:

$$\text{Radian unit} = \text{Degree unit} \times \frac{\pi}{180}$$

7.12 Special function instructions  
7.12.13 Conversion from floating-point angle to radian (Single precision) (RAD(P))

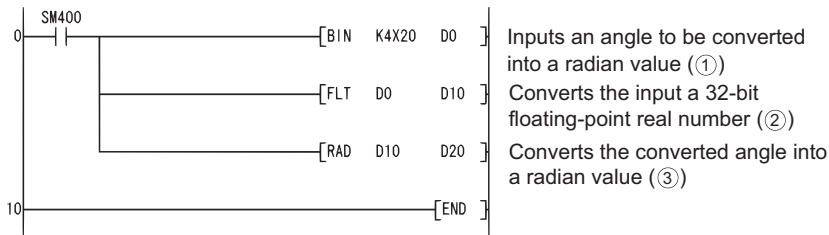
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The contents of the designated device or the result of the addition are not "0", or not within the following range(For the Universal model QCPU only): (Error code: 4140)  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$
  - The value of the specified device is  $-0$ . \*3  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)  
 \*3: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
  - The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program converts the angle set by the 4 BCD digits at X20 to X2F to radians, and stores results as 32-bit floating decimal point type real number at D20 and D21.

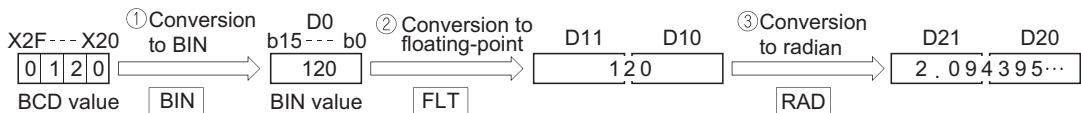
[Ladder Mode]



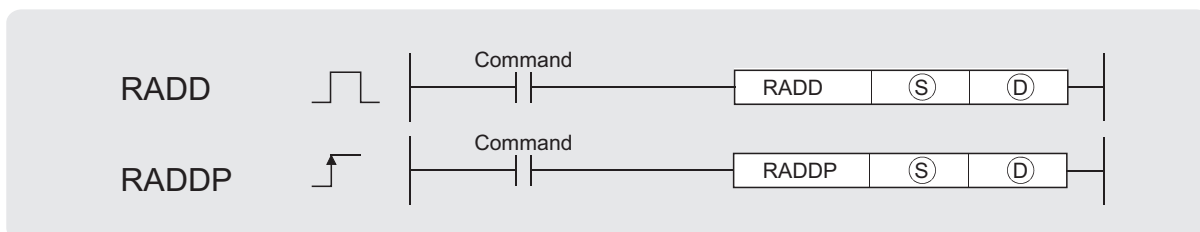
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D0
4	FLT	D0 D10
7	RAD	D10 D20
10	END	

[Operations involved when X20 to X2F designate a value of 120]



# 7.12.14 Conversion from floating-point angle to radian (Double precision) (RADD(P))

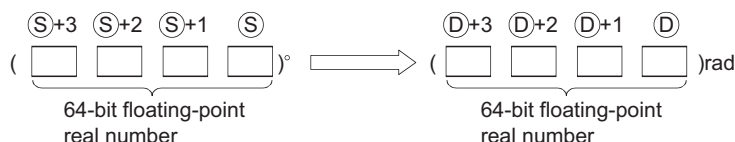


Ⓢ : Angle to be converted to radian units or head number of the devices where the angle is stored (real number)  
 ⓓ : Head number of the devices where the value converted in radian units will be stored (real number)

Setting Data	Internal Devices		R, ZR	J <small>0</small> ~ <small>3</small>		U <small>0</small> ~ <small>3</small>	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
ⓓ	—	○				—		—	—

## ★ Function

- The unit expressing the size of an angle is converted into the radian unit from the degree unit specified by Ⓢ, and its result is stored into the device specified by ⓓ.



- Conversion from degree to radian units is performed according to the following equation:

$$\text{Radian unit} = \text{Degree unit} \times \frac{\pi}{180}$$

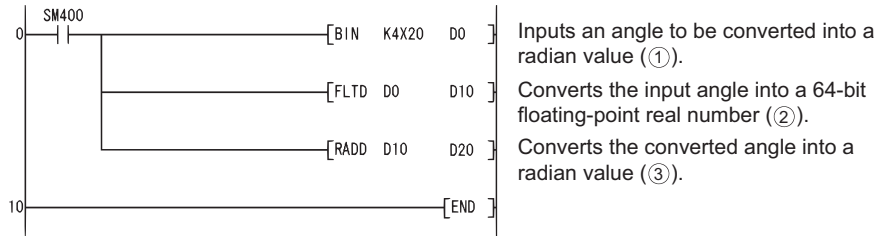
## ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0, 2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is  $-0$ . (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \leq | \text{Operation result} |$

## Program Example

- (1) The following program converts the angle set by the 4 BCD digits at X20 to X2F to radians, and stores results as 64-bit floating decimal point type real number at D20 to D23.

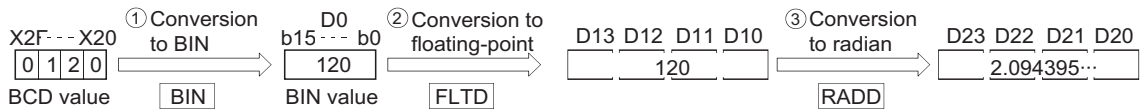
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D0
3	FLTD	D0 D10
6	RADD	D10 D20
9	END	

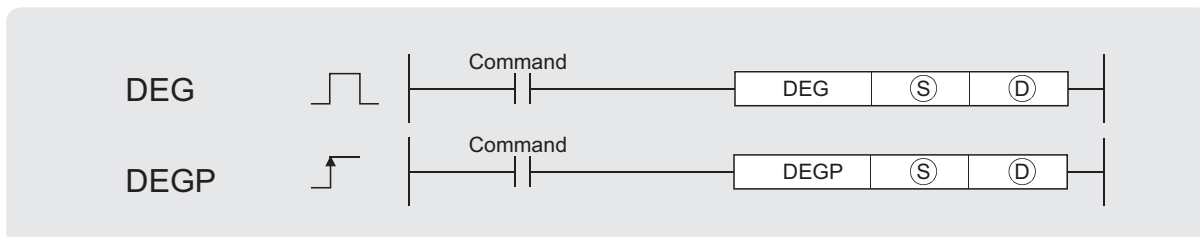
[Operations involved when X20 to X2F designate a value of 120]



# 7.12.15 Conversion from floating-point radian to angle (Single precision) (DEG(P))

Ver. Basic High performance Process Redundant Universal

Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



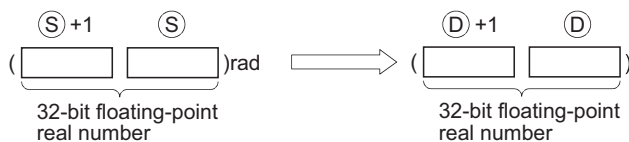
Ⓢ : Radian angle to be converted to degrees or head number of the devices where the radian angle is stored (real number)  
 Ⓣ : Head number of the devices where the value converted in degrees will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○*1	○	—	—	
Ⓣ	—	○	—	○	○*1	—	—		

\*1: Applicable for the Universal model QCPU only.

## ★ Function

- Converts units of angle size from radian units designated by Ⓢ to angles, and stores result at device number designated by Ⓣ.



- The conversion from radians to angles is performed according to the following equation:

$$\text{Degree unit} = \text{Radian unit} \times \frac{180}{\pi}$$

7.12 Special function instructions  
7.12.15 Conversion from floating-point radian to angle (Single precision) (DEG(P))

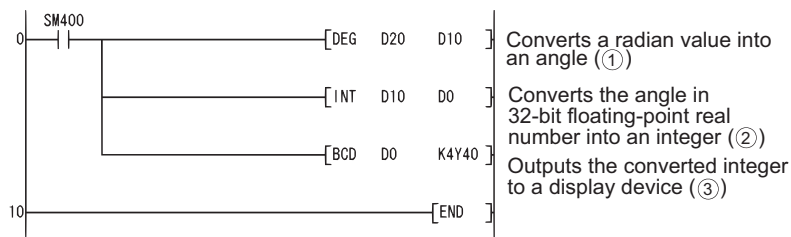
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value of the specified device is  $-0$ . \*2  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
  - \*2: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
  - The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program converts the radian value set with 32-bit floating decimal point type real number at D20 and D21 to angles, and stores the result as a BCD value at Y40 to Y4F.

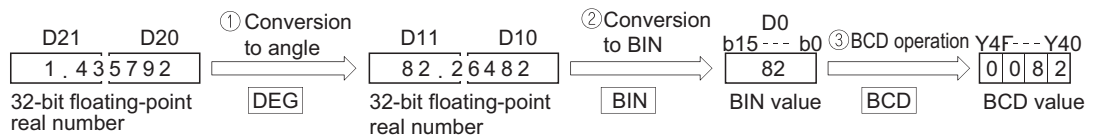
[Ladder Mode]



[List Mode]

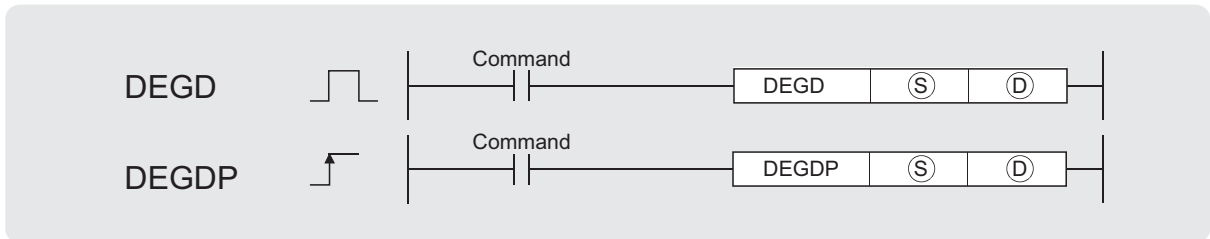
Step	Instruction	Device
0	LD	SM400
1	DEG	D20 D10
4	INT	D10 D0
7	BCD	D0 K4Y40
10	END	

[Operations involved when the values at D20 and D21 are 1.435792]





# 7.12.16 Conversion from floating-point radian to angle (Double precision) (DEGD(P))

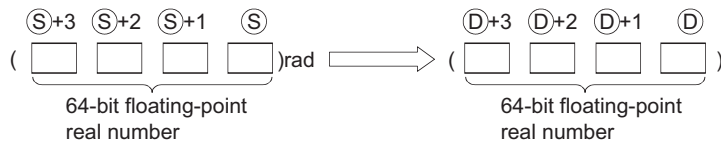


Ⓢ : Radian angle to be converted to degrees or head number of the devices where the radian angle is stored (real number)  
 Ⓣ : Head number of the devices where the value converted in degrees will be stored (real number)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

## ★ Function

- The unit expressing the size of an angle is converted into the degree unit from the radian unit specified by Ⓢ, and its result is stored into the device specified by Ⓣ.



- The conversion from radians to angles is performed according to the following equation:

$$\text{Degree unit} = \text{Radian unit} \times \frac{180}{\pi}$$

- When the operation results in -0 or an underflow, the result is processed as 0.

## ! Operation Error

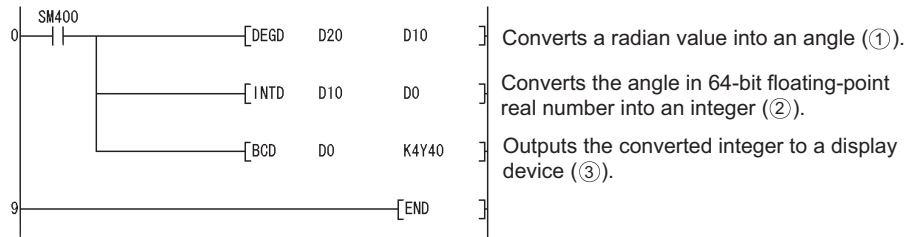
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \leq | \text{Operation result} |$

7.12 Special function instructions  
7.12.16 Conversion from floating-point radian to angle (Double precision) (DEGD(P))

## Program Example

- (1) The following program converts the radian value set with 64-bit floating decimal point type real number at D20 to D23 to angles, and stores the result as a BCD value at Y40 to Y4F.

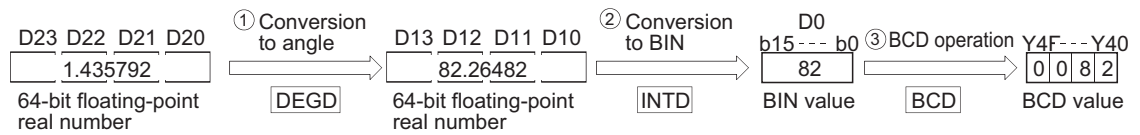
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	DEGD	D20 D10
4	INTD	D10 D0
7	BCD	D0 K4Y40
9	END	

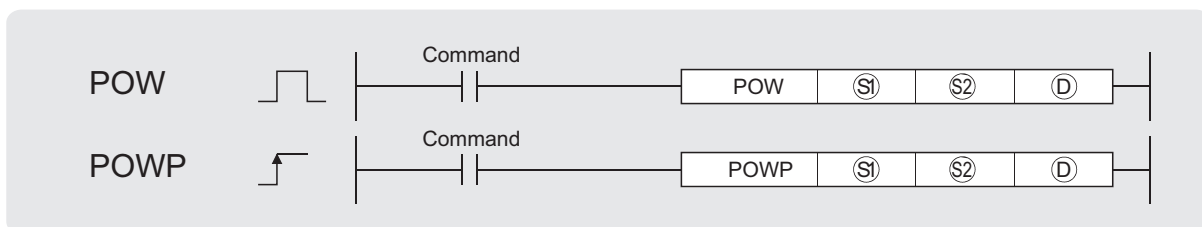
[Operations involved when the values at D20 to D23 are 1.435792]



# 7.12.17 Exponentiation operation on floating-point data (Single precision) (POW(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.  
 QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



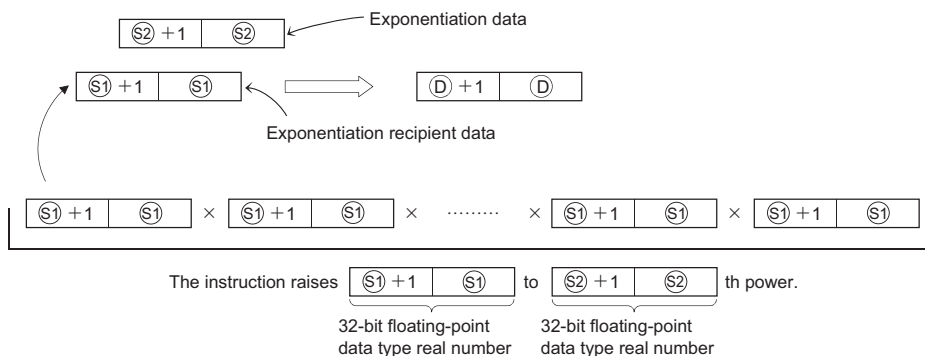
- Ⓢ1 : Exponentiation recipient data or head number of the devices where the exponentiation recipient data are stored (real number)
- Ⓢ2 : Exponentiation data or head number of the devices where the data are stored (real number)
- Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J <small>DATA</small>		U <small>DATA</small>	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○		—	○		○	△ *1	—
Ⓢ2	—	○		—	○		○	△ *1	
Ⓣ	—	○		—	○		○	—	—

\*1: Available only for real number

## ★ Function

- (1) This instruction raises the 32-bit floating-point data type real number specified by Ⓢ1 to the number nth specified by Ⓢ2 power, and then stores the operation result into the device specified by Ⓣ.



- (2) The following shows the values to be specified by and stored into Ⓢ1 or Ⓢ2.  
 $0, 2^{-126} \leq | \text{Set values (Storage values)} | < 2^{128}$
- (3) If the value resulted from the operation is  $-0$  or an underflow occurs, the result will be processed as 0.



## Operation Error

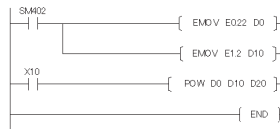
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The values specified by ① or ② are out of the ranges shown below. (Error code: 4140)  
 $0, 2^{-126} \leq | \text{Specified value (Storage value)} | < 2^{128}$
  - The value of ① or ② is  $-0$ . (Error code: 4140)
  - The values in the operation result is within the range shown below. (Error code: 4141)  
 $2^{126} \leq | \text{Value resulted from operation} |$



## Program Example

- (1) The following program raises the 32-bit floating-point data type real number data specified by D0 and D1 to the data specified by (D10 and D11)th power, when X10 is turned on. Then the program stores the operation result into D20 and D21.

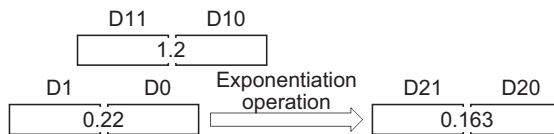
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM402
1	EMOV	E022 D0
4	EMOV	E12 D10
7	LD	X10
8	POW	D0 D10 D20
12	END	

[Operation]

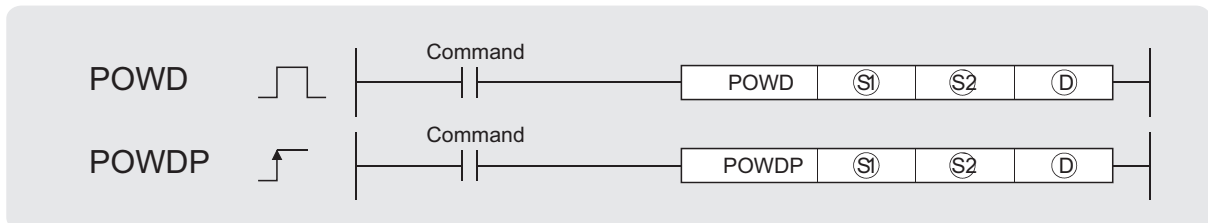


## 7.12.18 Exponentiation operation on floating-point data (Single precision) (POWD(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ<sub>1</sub> : Exponentiation recipient data or head number of the devices where the exponentiation recipient data are stored (real number)

Ⓢ<sub>2</sub> : Exponentiation data or head number of the devices where the data are stored (real number)

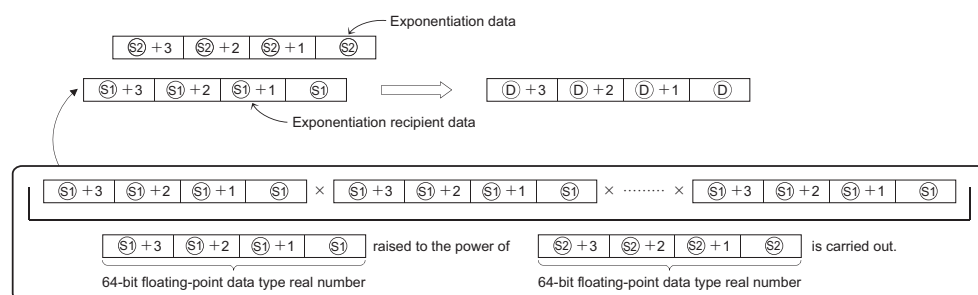
ⓓ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J <small>DATA</small>		U <small>DATA</small>	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○		—	○		○	△ *1	—
Ⓢ <sub>2</sub>	—	○		—	○		○	△ *1	—
ⓓ	—	○		—	○		○	—	—

\*1: Available only for real number

### ★ Function

- This instruction raises the 64-bit floating-point data type real number specified by Ⓢ<sub>1</sub> to the number nth specified by Ⓢ<sub>2</sub> power, and then stores the operation result into the device specified by ⓓ.



- The following shows the values to be specified by and stored into Ⓢ<sub>1</sub> or Ⓢ<sub>2</sub>  
 $0, 2^{-1022} \leq | \text{Set values (Storage values)} | < 2^{1024}$
- If the value resulted from the operation is  $-0$  or an underflow occurs, the result will be processed as 0.



## Operation Error

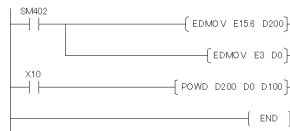
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The values specified by  $\textcircled{S1}$  or  $\textcircled{S2}$  are out of the range shown below. (Error code: 4140)  
 $0, 2^{-1022} \leq | \text{Set values (Storage values)} | < 2^{1024}$
  - The value of  $\textcircled{S1}$  or  $\textcircled{S2}$  is  $-0$ . (Error code: 4140)
  - The values resulted from the operation is within the range shown below.  
 $2^{1024} \leq | \text{Value resulted from operation} |$  (Error code: 4141)



## Program Example

- (1) The following program raises the 64-bit floating-point data type real number specified by D200 to D203 to the number nth specified by D0 to D3 power, when X10 is turned on. Then the program stores the operation result into D100 to D103.

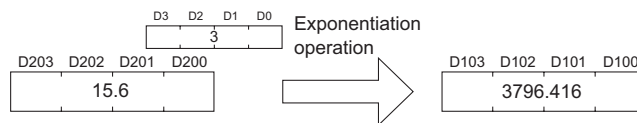
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM402
1	EDMOV	E15.6 D200
4	EDMOV	E3 D0
7	LD	X10
8	POWD	D200 D0 D100
12	END	

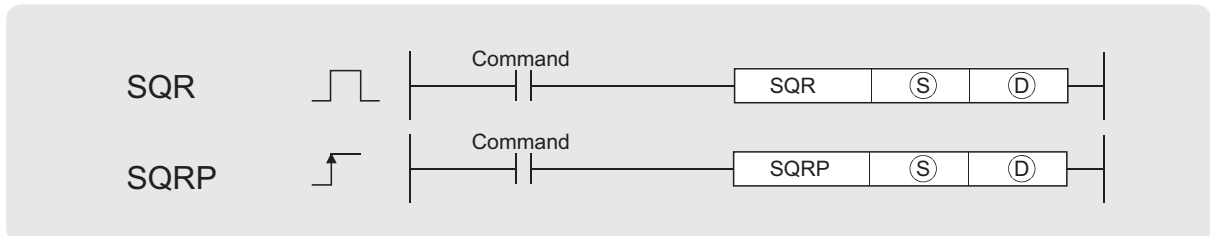
[Operation]



# 7.12.19 Square root operation for floating-point data (Single precision) (SQR(P))

Ver. Basic High performance Process Redundant Universal

Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



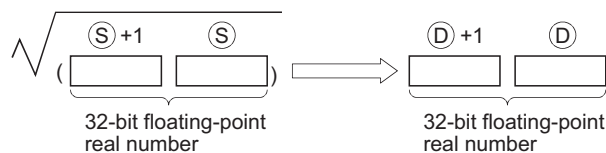
Ⓢ : Data of which the square root is obtained or head number of the devices where the data is stored (real number)  
 Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J <small>DATA</small>		U <small>DATA</small>	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○ <sup>*2</sup>	○	—	—	
Ⓣ	—	○	—	○	○ <sup>*2</sup>	—	—	—	

\*2: Applicable for the Universal model QCPU only.

## ★ Function

- Returns the square root of the value designated at Ⓢ, and stores the operation result in the device number designated at Ⓣ.



- Only positive values can be designated by Ⓢ. (Operation cannot be performed on negative numbers.)

7.12 Special function instructions  
7.12.19 Square root operation for floating-point data (Single precision) (SQR(P))

7

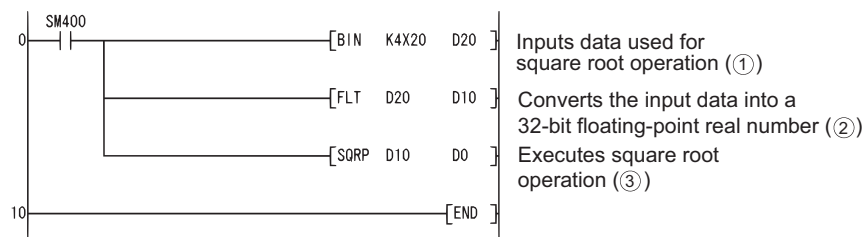
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value designated by ⑤ is a negative number. (Error code: 4100)
  - The contents of the designated device or the result of the addition are not "0", or not within the following range (For the Universal model QCPU only): (Error code: 4140)  
 $0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$
  - The value of the specified device is  $-0$ . \*3  
 (For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU, and Q4ARCPU) (Error code: 4100)  
 \*3: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
  - The result exceeds the following range (Operation results in an overflow)  
 (For the Universal model QCPU only)  
 $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
 (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program seeks the square root of the value set by the 4 BCD digits from X20 to X2F, and stores the result as a 32-bit floating decimal point type real number at D0 and D1.

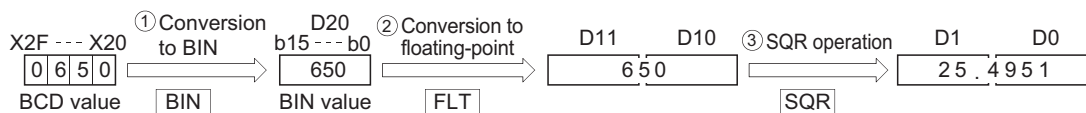
[Ladder Mode]



[List Mode]

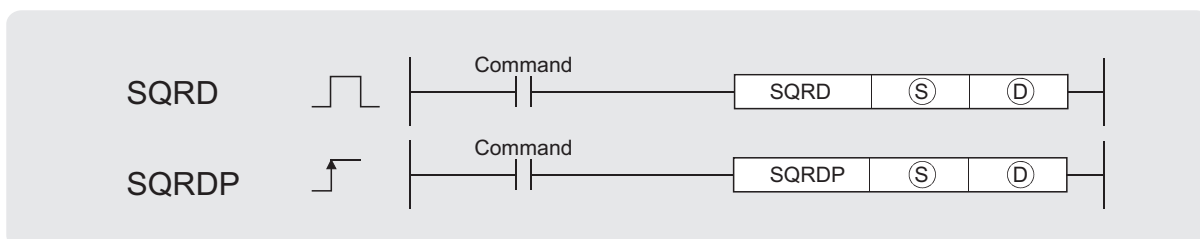
Step	Instruction	Device
0	LD	SM400
1	BIN	K4X20 D20
4	FLT	D20 D10
7	SQR	D10 D0
10	END	

[Operations involved when value designated by X20 to X2F is 650]





## 7.12.20 Square root operation for floating-point data (Double precision) (SQRD(P))

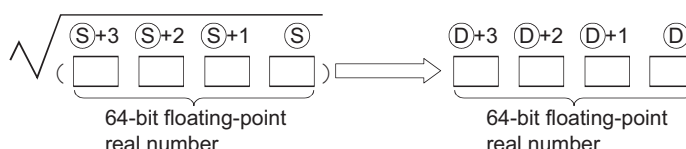


Ⓢ : Data of which the square root is obtained or head number of the devices where the data is stored (real number)  
 ⓓ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J0:0		U0:0	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
ⓓ	—	○				—		—	—

### ★ Function

- Returns the square root of the value designated at Ⓢ, and stores the operation result in the device number designated at ⓓ.



- Only positive values can be designated by Ⓢ. (Operation cannot be performed on negative numbers.)
- When the operation results in -0 or an underflow, the result is processed as 0.

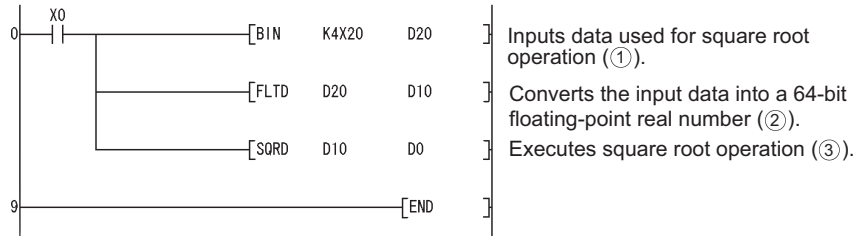
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value designated by Ⓢ is a negative number. (Error code: 4100)
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \leq | \text{Operation result} |$

## Program Example

- (1) The following program seeks the square root of the value set by the 4 BCD digits from X20 to X2F, and stores the result as a 64-bit floating decimal point type real number at D0 to D3.

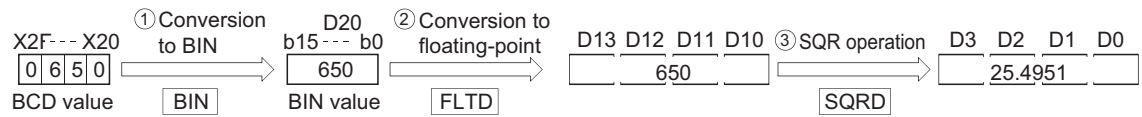
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	BIN	K4X20 D20
3	FLTD	D20 D10
6	SQRD	D10 D0
9	END	

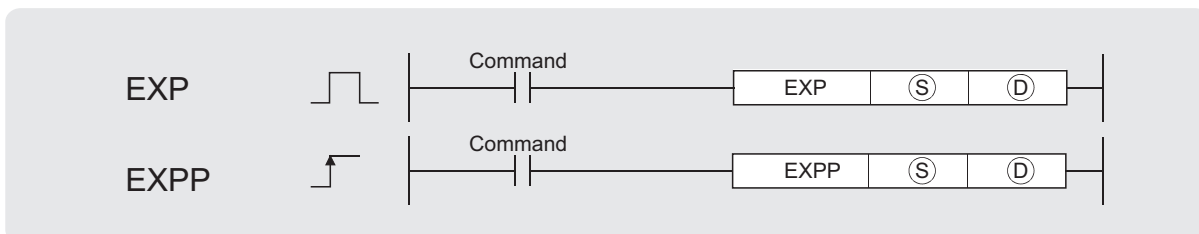
[Operations involved when value designated by X20 to X2F is 650]



# 7.12.21 Exponent operation on floating-point data (Single precision) (EXP(P))

Ver. Basic High performance Process Redundant Universal

Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Data of which the exponential value is obtained or head number of the devices where the data is stored (real number)  
 ⓓ : Head number of the devices where the operation result will be stored (real number)

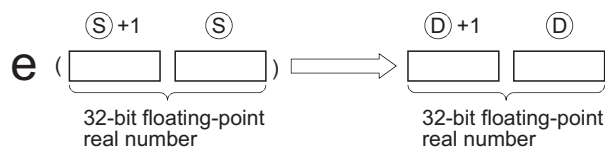
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○ <sup>*2</sup>	○	—	—	
ⓓ	—	○	—	○	○ <sup>*2</sup>	—	—	—	

\*2: Applicable for the Universal model QCPU only.

7

## ★ Function

- Returns the exponent of the value designated by Ⓢ, and stores the results of the operation at the device designated by ⓓ.



- Exponent operations are calculated taking the base (e) to be "2.71828".

7.12 Special function instructions  
7.12.21 Exponent operation on floating-point data (Single precision) (EXP(P))



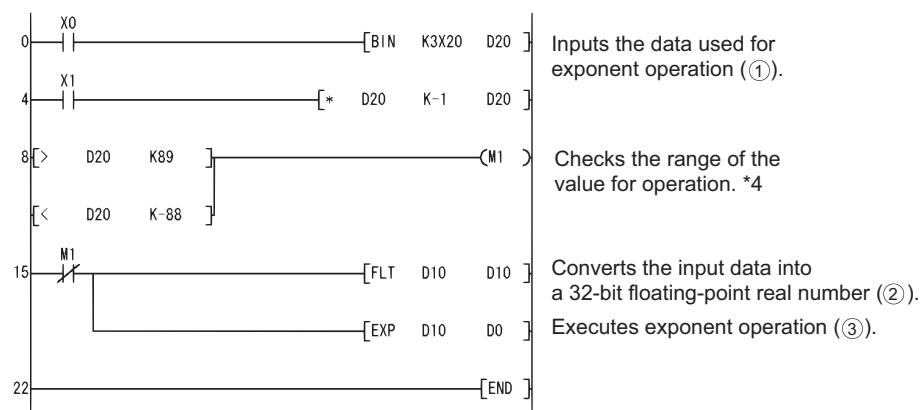
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Operation results are outside the range shown below: (Error code: 4100)
    - $2^{-126} \leq | \text{operation result} | \leq 2^{128}$   
(For a High Performance model QCPU)
    - $2^{-126} \leq | \text{operation result} | < 2^{128}$   
(For a Basic model QCPU/Process CPU/Redundant CPU)
  - The value of the specified device is  $-0$ . \*3  
(For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 4100)
- \*3: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
- The result exceeds the following range (Operation results in an overflow)  
(For the Universal model QCPU only)
    - $2^{128} \leq | \text{Operation result} |$  (Error code: 4141)
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ .  
(For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program performs an exponent operation on the value set by the 2 BCD digits at X20 to X27, and stores the results as a 32-bit floating decimal point real number at D0 and D1.

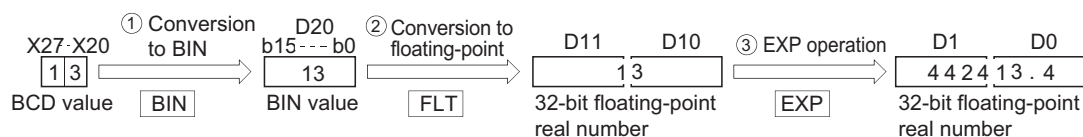
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	BIN	K3X20 D20
4	LD	X1
5	*	D20 K-1 D20
8	LD>	D20 K89
11	OR<	D20 K-88
14	OUT	M1
15	LDI	M1
16	FLT	D10 D10
19	EXP	D10 D0
22	END	

[Operations involved when value designated by X20 to X27 is 13]



\*4: The operation result will be under  $2^{129}$  if the BCD value of X20 to X27 is less than 89, from the calculation  $\log_e 2^{129} = 89.4$ .  
Because setting a value of over 90 will return an operation error, turn M0 ON if a value of over 90 has been set to avoid the error.

### POINT

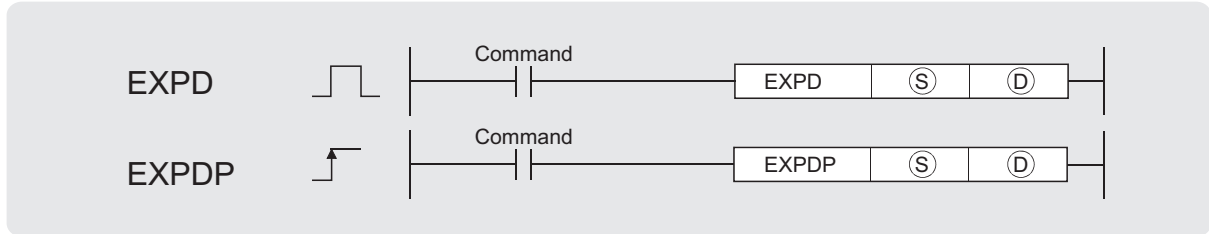
Conversion from natural logarithm to common logarithm

In the CPU module, calculation is made using a natural logarithm.

To obtain a common logarithm value, enter in, ⑤ a common logarithm value divided by 0.43429.

$$10^x = e^{\frac{x}{0.43429}}$$

## 7.12.22 Exponent operation on floating-point data (Double precision) (EXPDP(P))



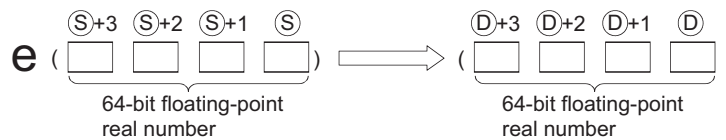
Ⓢ : Data of which the exponential value is obtained or head number of the devices where the data is stored (real number)

Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- Returns the exponent of the value designated by Ⓢ, and stores the results of the operation at the device designated by Ⓣ.



- Exponent operations are calculated taking the base (e) to be "2.71828".
- When the operation results in -0 or an underflow, the result is processed as 0.

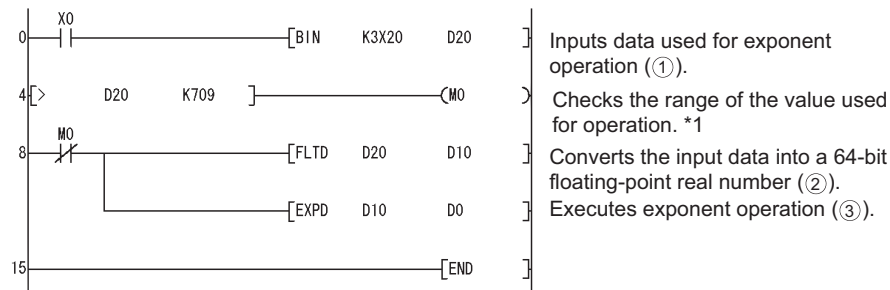
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value of the specified device is not in the following range: (Error code: 4140)  
 $0,2^{-1022} \leq | \text{value of specified device} | < 2^{1024}$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)  
 $2^{1024} \leq | \text{Operation result} |$

# Program Example

- (1) The following program performs an exponent operation on the value set by the 2 BCD digits at X20 to X31, and stores the results as a 64-bit floating decimal point real number at D0 to D3.

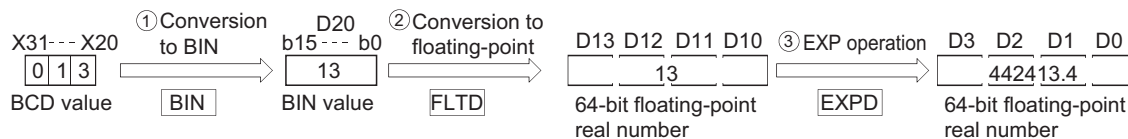
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	BIN	K3X20 D20
4	LD>	D20 K709
7	OUT	MO
8	LDI	MO
9	FLTD	D20 D10
12	EXPD	D10 D0
15	END	

[Operations involved when value designated by X20 to X31 is 13]



\*1: The operation result will be under  $2^{1024}$  if the BCD value of X20 to X31 is less than 709, from the calculation  $\log_e 2^{1024} = 709.7832$ .

Because setting a value of over 710 will return an operation error, turn M0 ON if a value of over 710 has been set to avoid the error.

## POINT

Conversion from natural logarithm to common logarithm

In the CPU module, calculation is made using a natural logarithm.

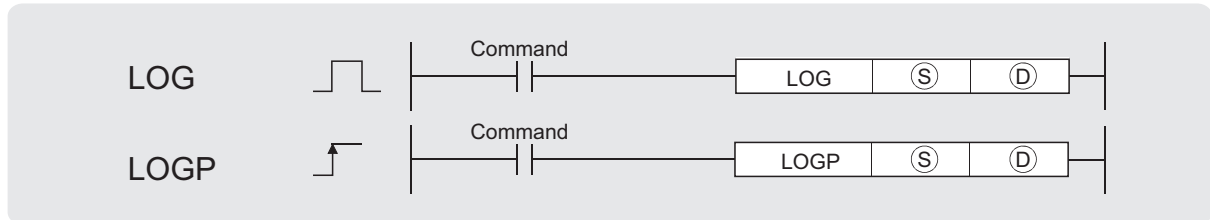
To obtain a common logarithm value, enter in,  $\text{Ⓢ}$  a common logarithm value divided by 0.43429.

$$10^x = e^{\frac{x}{0.43429}}$$

# 7.12.23 Natural logarithm operation on floating-point data (Single precision) (LOG(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ : Data of which the natural logarithm is obtained or head number of the devices where the data is stored (real number)

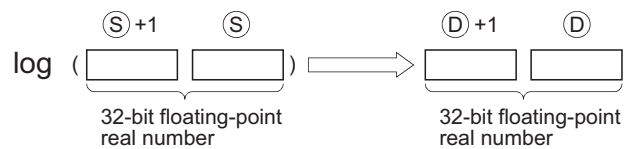
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JMO		UMGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	○*2	○	—	—	
Ⓣ	—	○	—	○	○*2	○	—	—	

\*2: Applicable for the Universal model QCPU only.

## ★ Function

- Returns the natural logarithm of the value designated by Ⓢ taking (e) as base, and stores operation results at device designated by Ⓣ.



- Only positive values can be designated by Ⓢ. (Operation cannot be performed on negative numbers.)



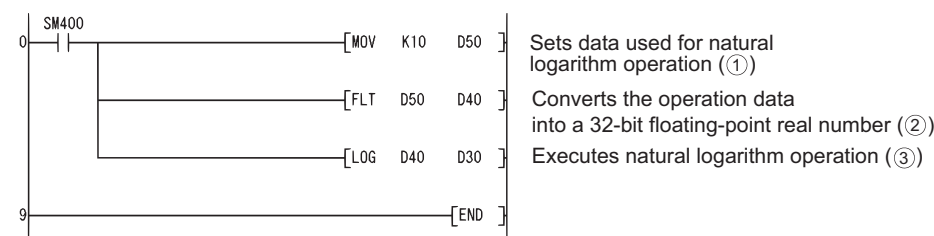
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value designated by (S) is negative. (Error code: 4100)
  - The value designated by (S) is 0. (Error code: 4100)
  - The contents of the designated device or the result of the addition are not "0", or not within the following range (For the Universal model QCPU only): (Error code: 4140)
 
$$0, 2^{-126} \leq | \text{Contents of designated device} | < 2^{128}$$
  - The value of the specified device is  $-0.$  \*3 (Error code: 4100)  
(For the Basic model QCPU, High Performance model QCPU, Process CPU, Redundant CPU)
- \*3: There are CPU modules that will not result in an operation error if  $-0$  is specified. For details, refer to 3.2.4.
- The result exceeds the following range (Operation results in an overflow) (For the Universal model QCPU only) (Error code: 4141)
 
$$2^{128} \leq | \text{Operation result} |$$
  - The value of the specified device is  $-0$ , unnormalized number, nonnumeric, and  $\pm\infty$ . (For the Universal model QCPU only) (Error code: 4140)

## Program Example

- (1) The following program seeks the natural logarithm of the value "10" set by D50, and stores the result at D30 and D31.

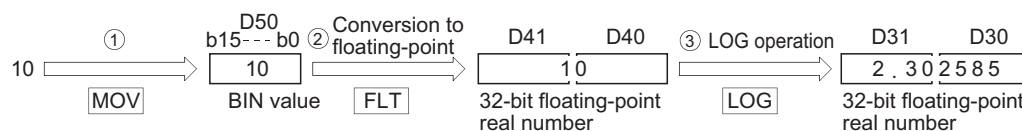
[Ladder Mode]



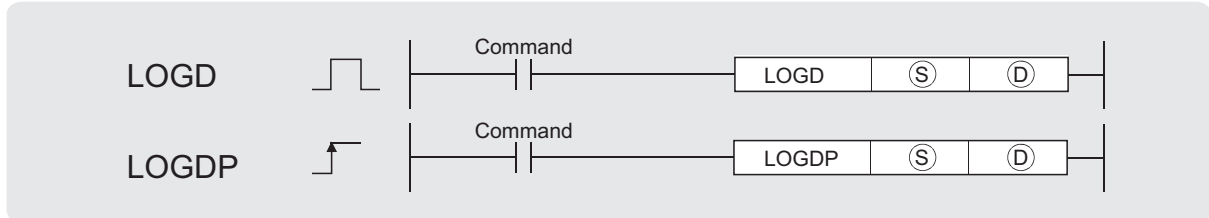
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOV	K10 D50
3	FLT	D50 D40
6	LOG	D40 D30
9	END	

[Operation]



# 7.12.24 Natural logarithm operation on floating-point data (Double precision) (LOGD(P))



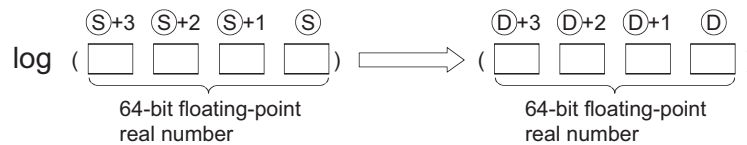
Ⓢ : Data of which the natural logarithm is obtained or head number of the devices where the data is stored (real number)

Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	—	○				—		—	—

## ★ Function

- Returns the natural logarithm of the value designated by Ⓢ taking (e) as base, and stores operation results at device designated by Ⓣ.



- Only positive values can be designated by Ⓢ. (Operation cannot be performed on negative numbers.)
- When the operation results in -0 or an underflow, the result is processed as 0.

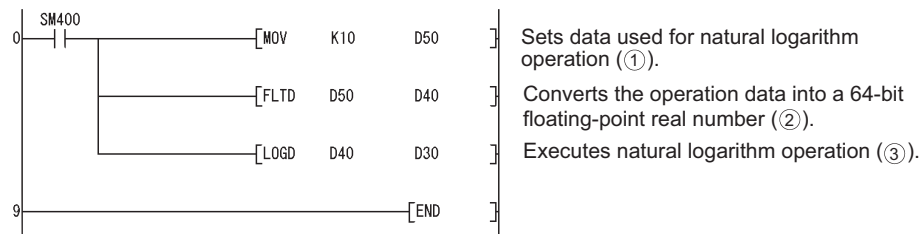
## ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The value designated by Ⓢ is negative. (Error code: 4100)
  - The value designated by Ⓢ is 0. (Error code: 4100)
  - The value of the specified device is not in the following range: (Error code: 4140)
 
$$0,2^{-1022} \cong | \text{value of specified device} | < 2^{1024}$$
  - The value of the designated device is -0. (Error code: 4140)
  - The result exceeds the following range (Operation results in an overflow): (Error code: 4141)
 
$$2^{1024} \cong | \text{Operation result} |$$

## Program Example

- (1) The following program seeks the natural logarithm of the value "10" set by D50, and stores the result at D30 to D33.

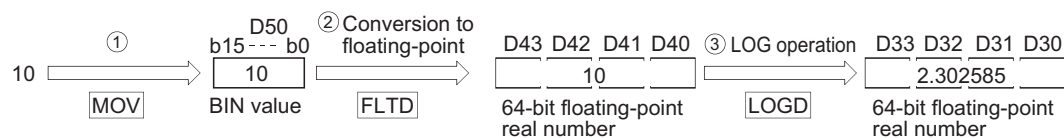
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	MOV	K10 D50
3	FLTD	D50 D40
6	LOGD	D40 D30
9	END	

[Operation]

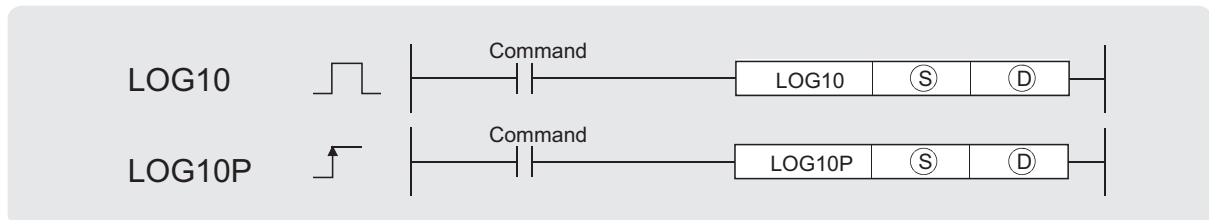


## 7.12.25 Common logarithm operation on floating-point data (Single precision) (LOG10(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ : Data of which the common logarithm is obtained or head number of the devices where the data are stored (real number)

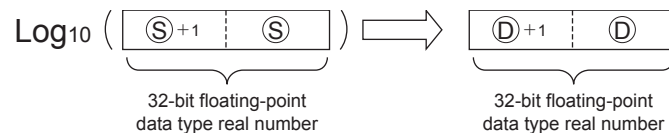
Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JAN		UNGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○	—	○	—	△ *1	—		
Ⓣ	—	○	—	○	—	—	—		

\*1: Available only for real number.

### ★ Function

- (1) This instruction obtains the value specified by Ⓢ for common logarithm (logarithm with base 10), and then stores the operation result into the device specified by Ⓣ.



- (2) Only positive values can be specified by Ⓢ. (Operation cannot be performed on negative numbers.)
- (3) If the value resulted from the operation is  $-0$  or an underflow occurs, the result will be processed as 0.

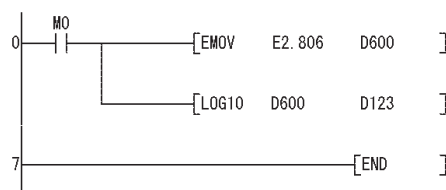
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The value specified by  $\textcircled{S}$  is negative. (Error code: 4100)
  - The value specified by  $\textcircled{S}$  is 0. (Error code: 4100)
  - The value of the specified device is not in the following range. (Error code: 4140)  
 $0, 2^{-126} \leq | \text{value of specified device} | < 2^{128}$
  - The value specified by  $\textcircled{S}$  is  $-0$ . (Error code: 4140)
  - The value resulted from the operation is within the range shown below. (Error code: 4141)  
 (When an overflow occurs):  
 $2^{128} \leq | \text{value of specified device} |$

## Program Example

- (1) The following program obtains the value for common logarithm of the 32-bit floating-point data type real number specified by D600 or D601, when X10 is turned on. Then the program stores the operation result into D123 or D124.

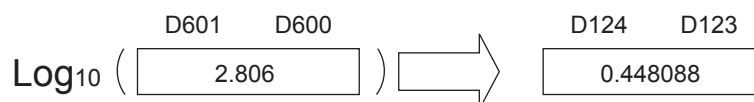
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	MO
1	EMOV	E2.806 D600
4	LOG10	D600 D123
7	END	

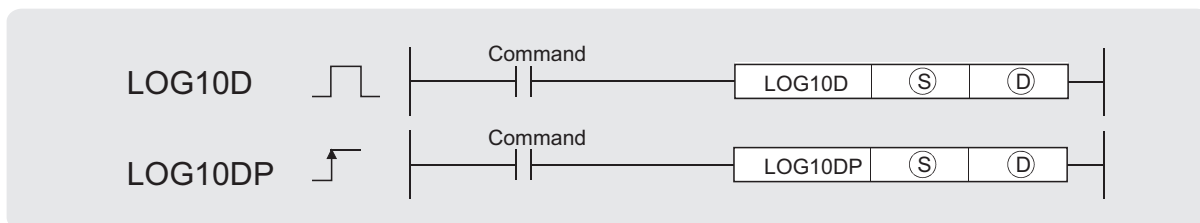
[Operation]



# 7.12.26 Common logarithm operation on floating-point data (Double precision) (LOG10D(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.  
 QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



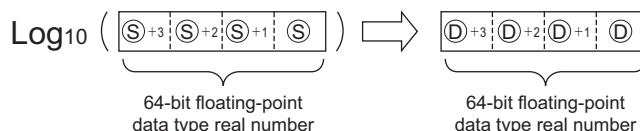
Ⓢ : Data of which the common logarithm is obtained or head number of the devices where the data are stored (real number)  
 Ⓣ : Head number of the devices where the operation result will be stored (real number)

Setting Data	Internal Devices		R, ZR	JAN		UNGO	Zn	Constants E	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		△ *1	—
Ⓣ	—	○				—		—	—

\*1: Available only for real number.

## ★ Function

- (1) This instruction obtains the value specified by Ⓢ for common logarithm (logarithm with base 10), and then stores the operation result into the device specified by Ⓣ.



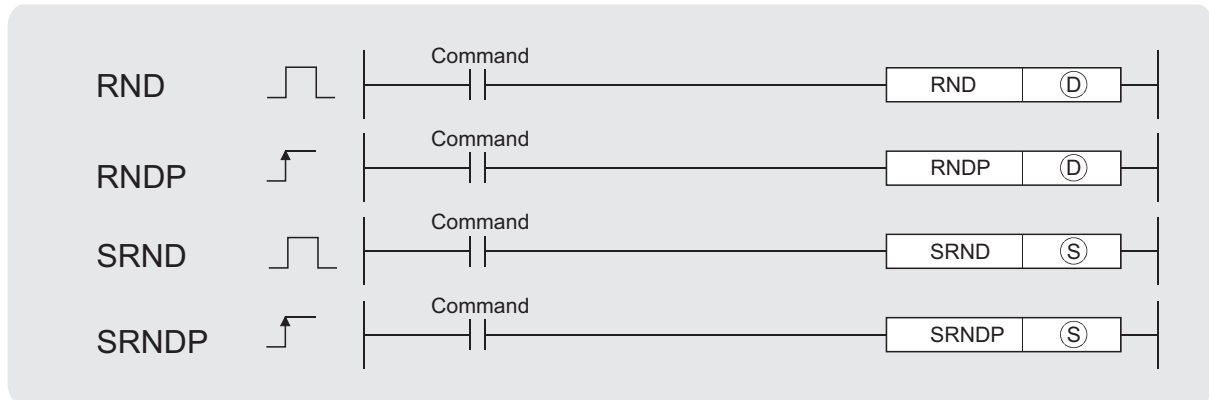
- (2) Only positive values can be specified by Ⓢ. (Operation cannot be performed on negative numbers.)
- (3) If the value resulted from the operation is  $-0$  or an underflow occurs, the result will be processed as 0.



## 7.12.27 Random number generation and series updates (RND(P),SRND(P))



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓧ : Head number of the devices where random numbers will be stored (BIN 16 bits)

Ⓢ : Random number serial data or the first number of the devices where the random number serial data is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓧ								—	—
Ⓢ								○	—

### ★ Function

The random number generation instruction generates random numbers conforming to a certain calculation formula. In the calculation using the formula, the result of previous calculation is used as a coefficient.

The random series change instruction can change the random number generation pattern.

#### RND

Generates random number of from 0 to 32767, and stores at device designated by Ⓧ.

#### SRND

Updates random number series according to the 16-bit BIN data being stored in device designated by Ⓢ.



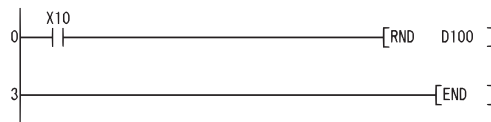
## ! Operation Error

- (1) There are no operation errors associated with the RND(P) or SRND(P) instructions.

## Program Example

- (1) The following program stores random number at D100 when X10 is turned ON.

[Ladder Mode]

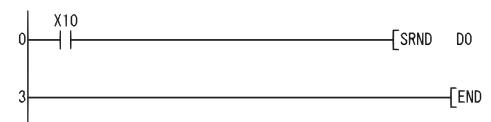


[List Mode]

Step	Instruction	Device
0	LD	X10
1	RND	D100
3	END	

- (2) The following program updates a random number series according to the contents of D0 when X10 is turned ON.

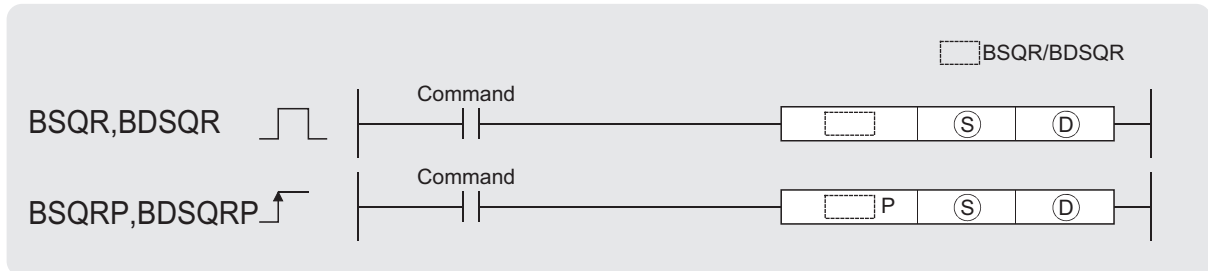
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	SRND	D0
3	END	

## 7.12.28 BCD 4-digit and 8-digit square roots (BSQR(P),BDSQR(P))



Ⓢ : Data of which the square root is obtained or the number of the device where the data is stored  
(BSQR(P): BCD 4 digits, BDSQR(P): BCD 8 digits)

Ⓣ : Head number of the devices where the square root calculation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	JOG		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—
Ⓣ								—	—

### ★ Function

#### BSQR

- Returns the square root of the value designated at Ⓢ, and stores the operation result in the device number designated at Ⓣ.

$$\sqrt{\text{Ⓢ}} = \boxed{\text{Integer part}} \boxed{\text{Decimal fraction part}}^{\text{Ⓣ}+1}$$

- Values that can be designated at Ⓢ are BCD values with a maximum of 4 digits (from 0 to 9999).
- The operation results of Ⓣ and Ⓣ+1 are stored as their respective BCD values of between 0 and 9999.
- Operation results are rounded off from the fifth decimal place.  
For this reason, the fourth decimal place has an error of  $\pm 1$ .

## BDSQR

- (1) Calculates the square root of the values designated by  $\textcircled{S}$  and  $\textcircled{S}+1$  and stores the results at the device designated by  $\textcircled{D}$ .

$$\sqrt{\underbrace{(\textcircled{S}+1 \quad \textcircled{S})}_{\text{2-word data}}} = \begin{array}{|c|} \hline \textcircled{D} \\ \hline \text{Integer part} \\ \hline \end{array} . \begin{array}{|c|} \hline \textcircled{D}+1 \\ \hline \text{Decimal fraction part} \\ \hline \end{array}$$

- (2) BCD value of a maximum of 8 digits (0 to 99999999) can be designated by  $\textcircled{S}$  and  $\textcircled{S}+1$ .
- (3) The operation results of  $\textcircled{D}$  and  $\textcircled{D}+1$  are stored as their respective BCD values of between 0 and 9999.
- (4) Operation results are rounded off from the fifth decimal place.  
For this reason, the fourth decimal place has an error of  $\pm 1$ .

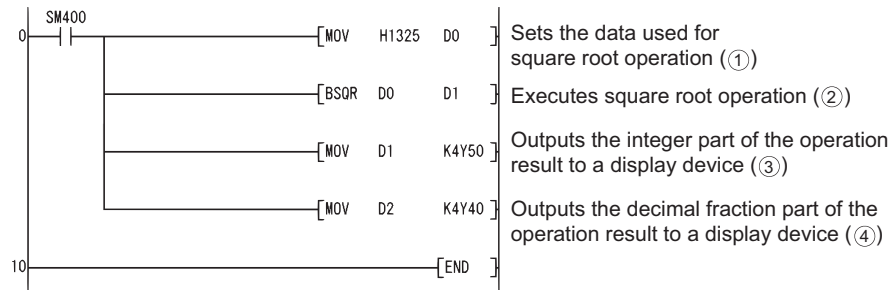
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data designated by  $\textcircled{S}$  is not a BCD value. (Error code: 4100)

## Program Example

- (1) The following program calculates the square root of BCD value 1325 and outputs the integer part to the 4 BCD digits from Y50 to Y5F, and the decimal fraction part to the 4 BCD digits from Y40 to Y4F.

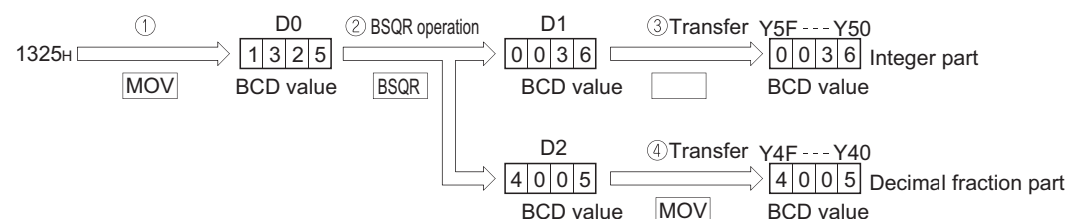
[Ladder Mode]



[List Mode]

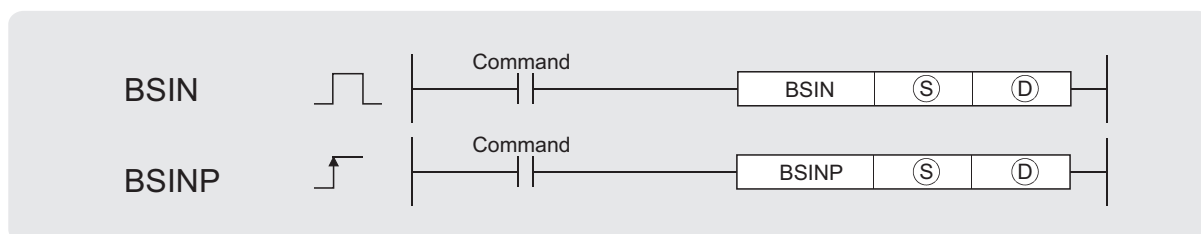
Step	Instruction	Device
0	LD	SM400
1	MOV	H1325 D0
3	BSQR	D0 D1
6	MOV	D1 K4Y50
8	MOV	D2 K4Y40
10	END	

[Operation]





## 7.12.29 BCD type SIN operation (BSIN(P))



Ⓢ : Data of which the SIN (sine) value is obtained or the number of the device where the data is stored (BCD 4 digits)

Ⓣ : Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○			—
Ⓣ	—	○				—			—

### ★ Function

- Calculates the SIN (sine) value of value (angle) designated by Ⓢ, and stores the sign of the operation result in the device designated at Ⓣ, and the operation result in the devices designated at Ⓣ+1 and Ⓣ+2.

$$\text{SIN } \textcircled{\text{S}} = \begin{array}{|c|} \hline \textcircled{\text{D}} \\ \hline \text{Sign} \\ \hline \end{array} \begin{array}{|c|} \hline \textcircled{\text{D}} + 1 \\ \hline \text{Integer part} \\ \hline \end{array} \begin{array}{|c|} \hline \textcircled{\text{D}} + 2 \\ \hline \text{Decimal fraction part} \\ \hline \end{array}$$

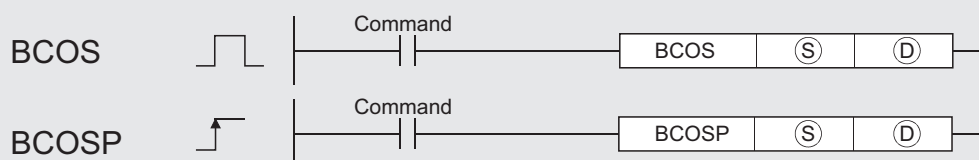
- The value designated at Ⓢ is a BCD value which can be between 0 and 360 degrees (in units of degrees).
- The sign for the operation result stored in Ⓣ will be "0" if the result is a positive value, and "1" if the result is a negative value.
- The operation results stored in Ⓣ+1 and Ⓣ+2 are BCD values between -1.000 and 1.000.
- Operation results are rounded off from the fifth decimal place.

### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The data designated by Ⓢ is not a BCD value. (Error code: 4100)
  - The data designated by Ⓢ is not in the range of from 0 to 360. (Error code: 4100)
  - The device specified by Ⓣ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)



## 7.12.30 BCD type COS operations (BCOS(P))



Ⓢ : Data of which the COS (cosine) value is obtained or head number of the devices where the data is stored (BCD 4 digits)

Ⓣ : Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○			—
Ⓣ	—	○				—			—

### ★ Function

- Calculates COS (cosine) value of value (angle) designated by Ⓢ, then stores the sign for the operation result in the word device designated by Ⓣ, and the operation result in the word device designated by Ⓣ+1 and Ⓣ+2.

$$\text{COS } \textcircled{\text{S}} = \begin{array}{|c|} \hline \textcircled{\text{D}} \\ \hline \text{Sign} \\ \hline \end{array} \begin{array}{|c|} \hline \textcircled{\text{D}}+1 \\ \hline \text{Integer part} \\ \hline \end{array} \begin{array}{|c|} \hline \textcircled{\text{D}}+2 \\ \hline \text{Decimal fraction part} \\ \hline \end{array}$$

- The value designated at Ⓢ is a BCD value which can be between 0 and 360 degrees (in units of degrees).
- The sign for the operation result stored in Ⓣ will be "0" if the result is a positive value, and "1" if the result is a negative value.
- The operation results stored in Ⓣ+1 and Ⓣ+2 are BCD values between  $-1.000$  and  $1.000$ .
- Operation results are rounded off from the fifth decimal place.

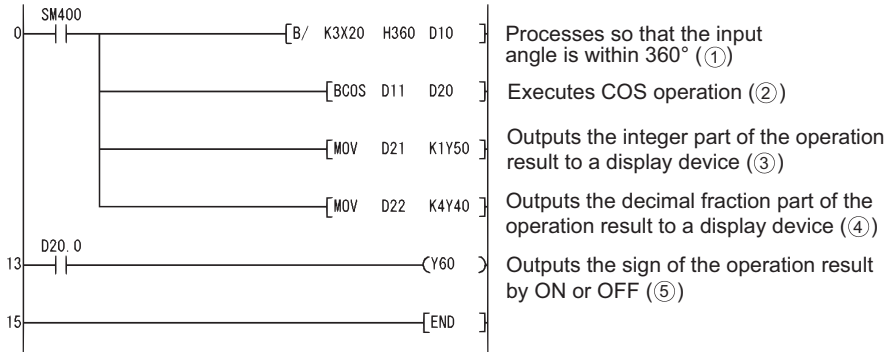
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The data designated by Ⓢ is not a BCD value. (Error code: 4100)
  - The data designated by Ⓢ is not in the range of from 0 to 360. (Error code: 4100)
  - The device specified by Ⓣ exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)

# Program Example

- (1) The following program calculates the cosine of the data designated by the 3 BCD digits from X20 to X2B and outputs the integer part of the result to 1 BCD digit from Y50 to Y53, and the decimal fraction part of the result to the 4 BCD digits from Y40 to Y4F. Y60 is turned ON if the results of the operation are negative.

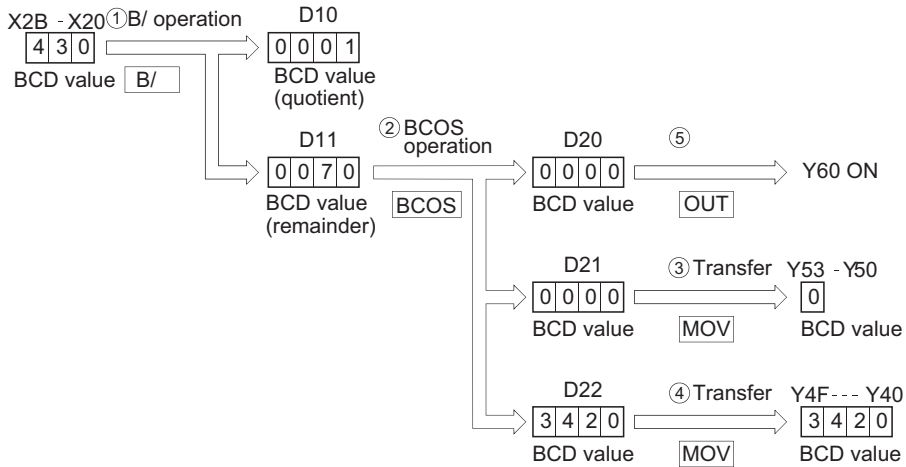
[Ladder Mode]



[List Mode]

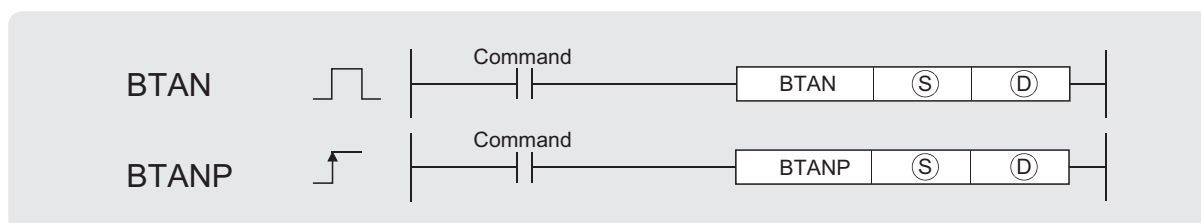
Step	Instruction	Device
0	LD	SM400
1	B/	K3X20 H360 D10
5	BCOS	D11 D20
8	MOV	D21 K1Y50
11	MOV	D22 K4Y40
13	LD	D20.0
14	OUT	Y60
15	END	

[Operations involved when value designated by X20 to X2B is 430]





## 7.12.31 BCD type TAN operation (BTAN(P))



Ⓢ : Data of which the TAN (tangent) value is obtained or head number of the devices where the data is stored (BCD 4 digits)

Ⓣ : Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○			—
Ⓣ	—	○				—			—

### ★ Function

- Calculates TAN (tangent) value for value (angle) designated by Ⓢ, and stores the sign for the operation result in the word device designated by Ⓣ, and the operation result in the word device designated by Ⓣ+1 and Ⓣ+2.

$$\text{TAN } \textcircled{\text{S}} = \begin{array}{|c|} \hline \textcircled{\text{D}} \\ \hline \text{Sign} \\ \hline \end{array} \begin{array}{|c|} \hline \textcircled{\text{D}} + 1 \\ \hline \text{Integer part} \\ \hline \end{array} \begin{array}{|c|} \hline \textcircled{\text{D}} + 2 \\ \hline \text{Decimal fraction part} \\ \hline \end{array}$$

- The value designated at Ⓢ is a BCD value which can be between 0 and 360 degrees (in units of degrees).
- The sign for the operation result stored in Ⓣ will be "0" if the result is a positive value, and "1" if the result is a negative value.
- The operation results stored at Ⓣ+1 and Ⓣ+2 are BCD values within the range of from -57.2901 and 57.2902.
- Operation results are rounded off from the fifth decimal place.

### ! Operation Error

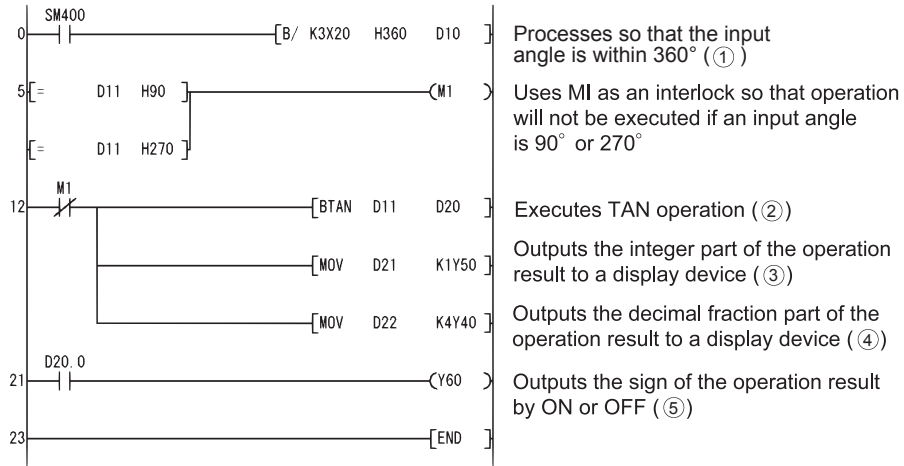
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The data designated by Ⓢ is not a BCD value. (Error code: 4100)
  - The data designated by Ⓢ is not in the range of from 0 to 360. (Error code: 4100)
  - The data designated by Ⓢ is 90° or 270°. (Error code: 4100)
  - The device specified by Ⓣ exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)

## Program Example

- (1) The following program calculates the tangent of the data stored in the 3 BCD digits from X20 to X2B, and stores the integer part of the results in the 4 BCD digits from Y50 to Y53, and the decimal fraction part in the 4 BCD digits from Y40 to Y4F.

Y60 is turned ON if the results of the operation are negative.

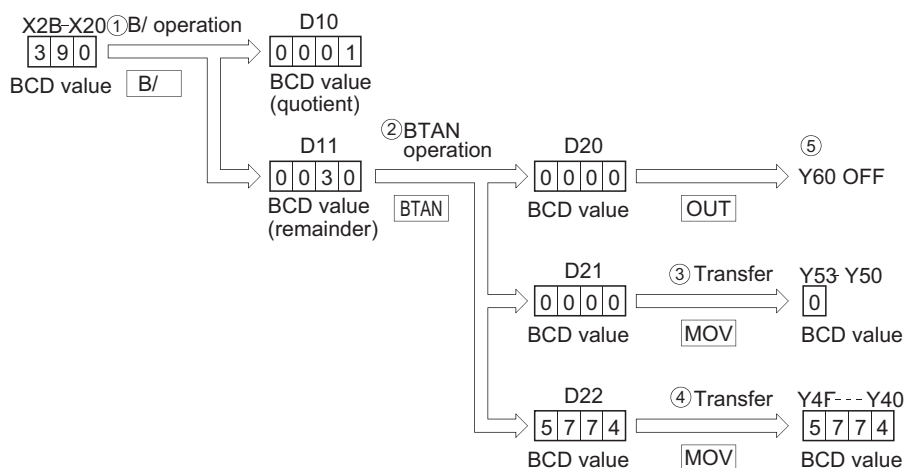
[Ladder Mode]

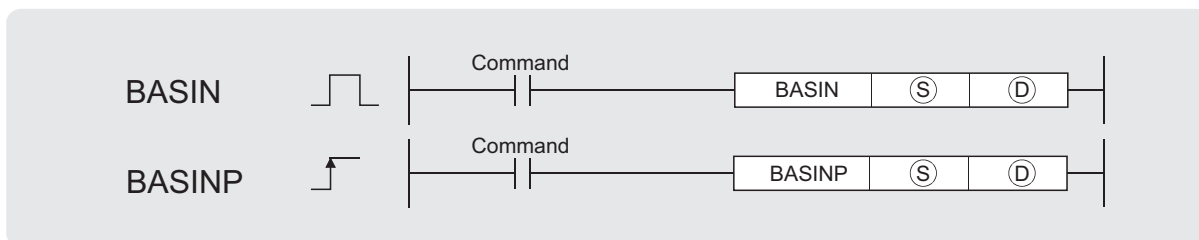


[List Mode]

Step	Instruction	Device
0	LD	SM400
1	B/	K3X20 H360 D10
5	LD=	D11 H90
8	OR=	D11 H270
11	OUT	M1
12	LD1	M1
13	BTAN	D11 D20
16	MOV	D21 K1Y50
19	MOV	D22 K4Y40
21	LD	D20.0
22	OUT	Y60
23	END	

[Operations involved when X20 to X2B designate a value of 390]



7.12.32 BCD type SIN<sup>-1</sup> operations (BASIN(P))

Ⓢ : Number of the device where data of which the SIN<sup>-1</sup> (inverse sine) value is obtained is stored (BCD 4 digits)

Ⓣ : Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		—	—
Ⓣ	○	○				○		—	—

## ★ Function

- Returns the SIN<sup>-1</sup> (inverse sine) value of the value designated by Ⓢ and stores operation results (angles) at device designated by Ⓣ.

$$\text{SIN}^{-1} \left( \begin{array}{|c|} \hline \text{Ⓢ} \\ \hline \text{Sign} \\ \hline \end{array} \begin{array}{|c|} \hline \text{Ⓢ}+1 \\ \hline \text{Integer part} \\ \hline \end{array} \begin{array}{|c|} \hline \text{Ⓢ}+2 \\ \hline \text{Decimal fraction part} \\ \hline \end{array} \right) = \text{Ⓣ}$$

- A sign for the operation data is set at Ⓢ. If the operation data is a positive value, this is set at "0", and if it is a negative value, it is set at "1".
- The part before the decimal point and fraction part are stored at Ⓢ + 1 and Ⓢ + 2 respectively, as BCD values. (Settings can be between 0 and 1.0000.)
- Operation results stored at Ⓣ are BCD values between 0 and 90 degrees, and 270 and 360 degrees (degree units).
- Calculation results are a value from which the decimal fraction part has been rounded.

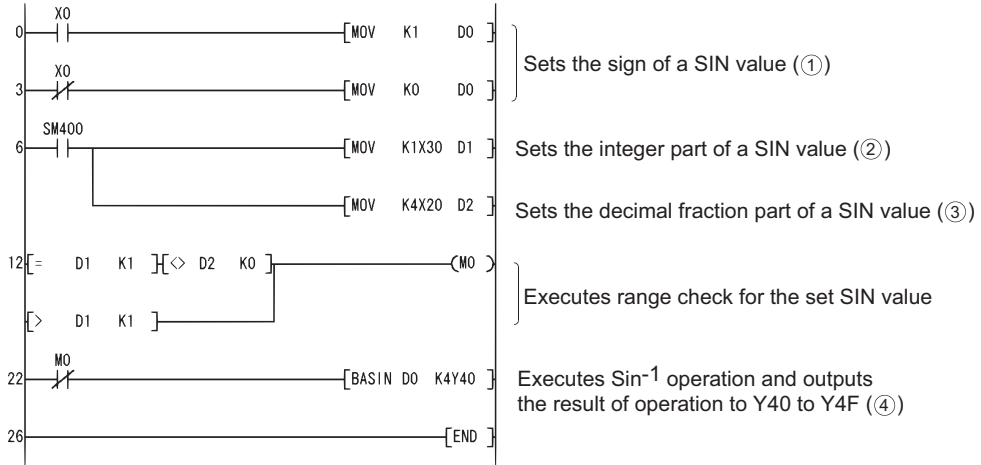
## ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The data designated by Ⓢ is not a BCD value. (Error code: 4100)
  - Data designated by Ⓢ is not in the range of from -1.0000 to 1.0000. (Error code: 4100)
  - The device specified by Ⓢ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program performs a  $\text{SIN}^{-1}$  operation on the sign (positive when X0 is OFF, and negative when X0 is ON), the BCD 1-digit integer part from X30 to X33 and the BCD 4-digit decimal fraction part from X20 to X2F, and outputs the calculated angle in 4 BCD digits from Y40 to Y4F.

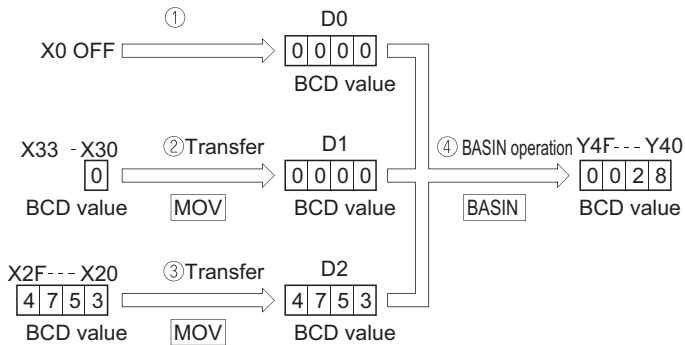
[Ladder Mode]



[List Mode]

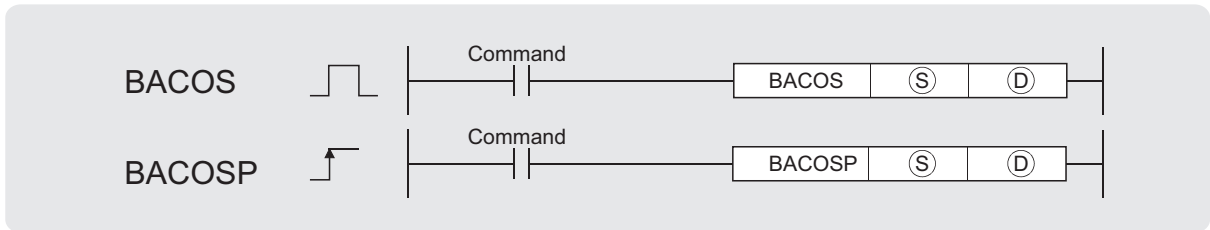
Step	Instruction	Device
0	LD	X0
1	MOV	K1 D0
3	LDI	X0
4	MOV	K0 D0
6	LD	SM400
7	MOV	K1X30 D1
10	MOV	K4X20 D2
12	LD=	D1 K1
15	AND<>	D2 K0
18	OR>	D1 K1
21	OUT	MO
22	LDI	MO
23	BASIN	D0 K4Y40
26	END	

[Operations involved when X20 to X33 designates value of 0.4753]



# 7.12.33 BCD type COS<sup>-1</sup> operation (BACOS(P))

Basic
High performance
Process
Redundant
Universal



Ⓢ : Number of the device where data of which the COS<sup>-1</sup> (inverse cosine) value is obtained is stored (BCD 4 digits)  
 Ⓣ : Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		—	
Ⓣ	○	○				○		—	

## ★ Function

- Returns the COS<sup>-1</sup> (inverse cosine) value of the value designated by Ⓢ, and stores operation results at device designated by Ⓣ.

$$\text{COS}^{-1} \left( \begin{array}{|c|} \hline \text{Ⓢ} \\ \hline \text{Sign} \\ \hline \end{array} \begin{array}{|c|} \hline \text{Ⓢ}+1 \\ \hline \text{Integer part} \\ \hline \end{array} \begin{array}{|c|} \hline \text{Ⓢ}+2 \\ \hline \text{Decimal fraction part} \\ \hline \end{array} \right) = \text{Ⓣ}$$

- A sign for the operation data is set at Ⓢ.  
If the operation data is a positive value, this is set at "0", and if it is a negative value, it is set at "1".
- The part before the decimal point and fraction part are stored at Ⓢ + 1 and Ⓢ + 2 respectively, as BCD values.  
(Settings can be between 0 and 1.0000.)
- The operation results stored at Ⓣ will be a BCD value in the range of between 0 and 180° (degree units).
- Calculation results are a value from which the decimal fraction part has been rounded.

## ! Operation Error

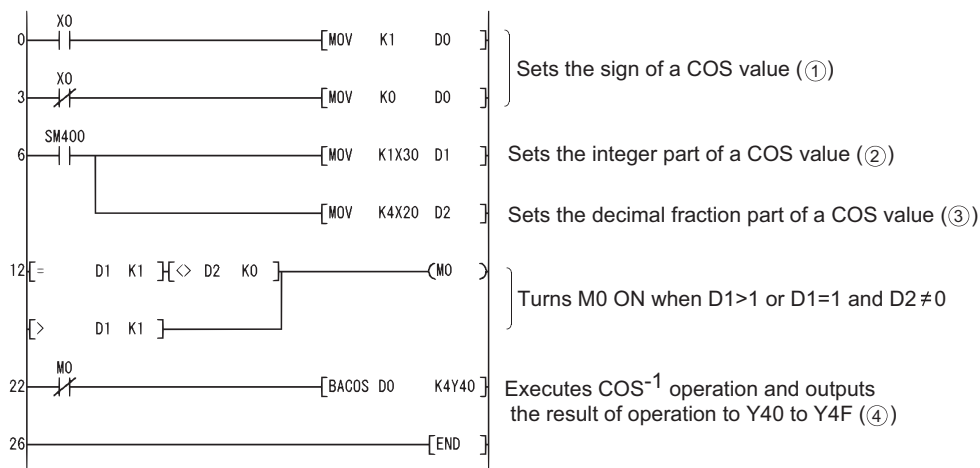
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The operation data designated by Ⓢ is not a BCD value. (Error code: 4100)
  - The operation data designated by Ⓢ is not in the range of from -1.0000 to 1.0000. (Error code: 4100)
  - The device specified by Ⓢ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

7.12 Special function instructions  
 7.12.33 BCD type COS<sup>-1</sup> operation (BACOS(P))

## Program Example

- (1) The following program performs a  $\text{COS}^{-1}$  operation on the sign (positive when X0 is OFF, and negative when X0 is ON), the BCD 1-digit integer part from X30 to X33 and the BCD 4-digit decimal fraction part from X20 to X2F, and outputs the calculated angle in 4 BCD digits from Y40 to Y4F.

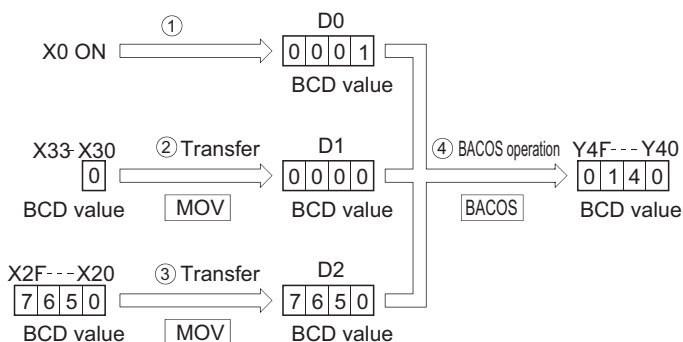
[Ladder Mode]



[List Mode]

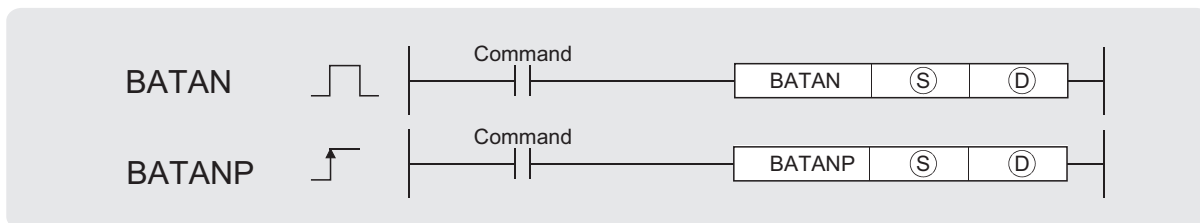
Step	Instruction	Device
0	LD	X0
1	MOV	K1 D0
3	LDI	X0
4	MOV	K0 D0
6	LD	SM400
7	MOV	K1X30 D1
10	MOV	K4X20 D2
12	LD=	D1 K1
15	AND<>	D2 K0
18	OR>	D1 K1
21	OUT	M0
22	LDI	M0
23	BACOS	D0 K4Y40
26	END	

[Operations involved if X0 and X20 to X33 designate a value of -0.7650]



# 7.12.34 BCD type TAN<sup>-1</sup> operations (BATAN(P))

Basic
High performance
Process
Redundant
Universal



(S) : Number of the device where data of which the TAN-1 (inverse tangent) value is obtained is stored (BCD 4 digits)  
 (D) : Head number of the devices where the operation result will be stored (BCD 4 digits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
(S)	—	○				—		—	
(D)	○	○				○		—	

## ★ Function

- Performs TAN<sup>-1</sup> (inverse tangent) on value designated by (S) and stores operation results (angles) at device designated by (D).

$$\text{TAN}^{-1} \left( \begin{array}{|c|c|c|} \hline \text{(S)} & \text{(S)+1} & \text{(S)+2} \\ \hline \text{Sign} & \text{Integer part} & \text{Decimal fraction part} \\ \hline \end{array} \right) = \text{(D)}$$

- A sign for the operation data is set at (S).  
If the operation data is a positive value, this is set at "0", and if it is a negative value, it is set at "1".
- The part before the decimal point and fraction part are stored at (S) + 1 and (S) + 2 respectively, as BCD values.  
(Values from 0 to 9999.9999 can be set.)
- Operation results stored at (D) are BCD values between 0 and 90 degrees, and 270 and 360 degrees (degree units).
- Calculation results are a value from which the decimal fraction part has been rounded.

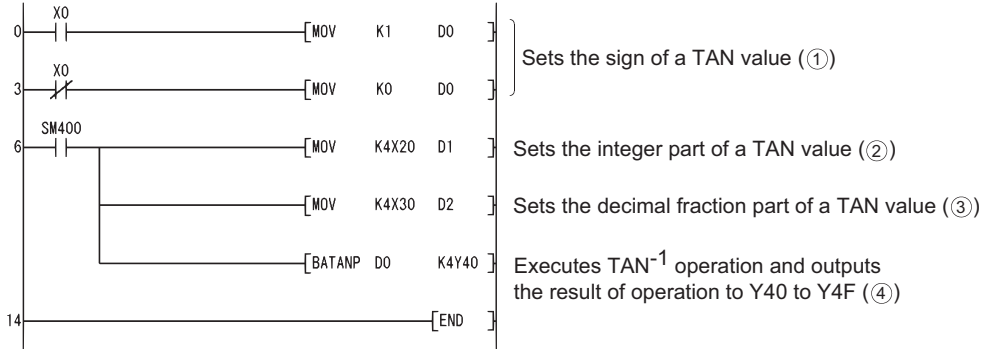
## ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The operation data designated by (S) is not a BCD value. (Error code: 4100)
  - The device specified by (S) exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program performs a TAN<sup>-1</sup> operation on the sign (positive when X0 is OFF, and negative when X0 is ON), the BCD 4-digit integer part from X20 to X2F and the BCD 4-digit decimal fraction part from X30 to X3F, and outputs the calculated angle in 4 BCD digits from Y40 to Y4F.

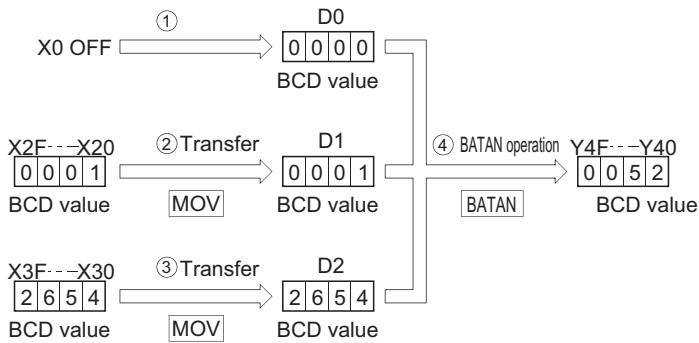
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	MOV	K1 D0
3	LDI	X0
4	MOV	K0 D0
6	LD	SM400
7	MOV	K4X20 D1
9	MOV	K4X30 D2
11	BATANP	D0 K4Y40
14	END	

[Operations involved when X0 and X20 to X2F designate a value of 1.2654]

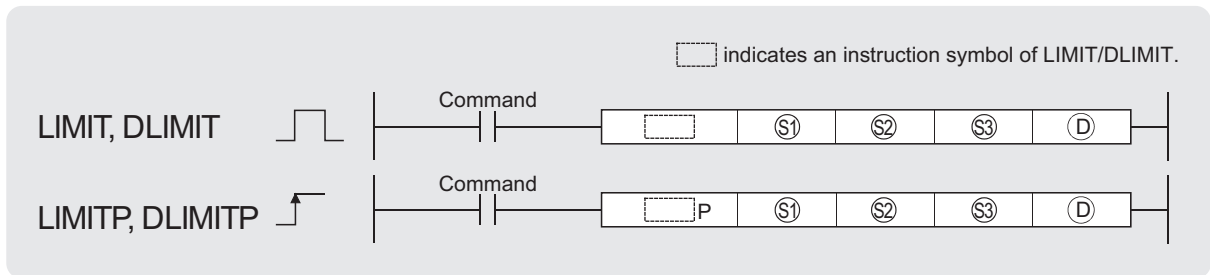




# 7.13 Data Control Instructions

## 7.13.1 Upper and lower limit controls for BIN 16-bit and BIN 32-bit data (LIMIT(P),DLIMIT(P))

Basic High performance Process Redundant Universal



- Ⓢ1 : Lower limit value (minimum output threshold value) (BIN 16/32 bits)
- Ⓢ2 : Upper limit value (maximum output threshold value) (BIN 16/32 bits)
- Ⓢ3 : Input value to be controlled by the upper and lower limit control (BIN 16/32 bits)
- Ⓧ : Head number of the devices where the output value controlled by the upper and lower limit control will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1					○			○	—
Ⓢ2					○			○	—
Ⓢ3					○			○	—
Ⓧ					○			—	—

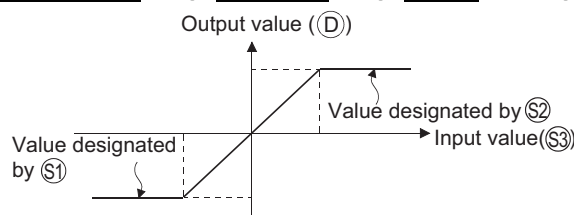
### ★ Function

#### LIMIT

- (1) Controls the output value to be stored at the device designated by Ⓧ by checking whether the input value (BIN 16 bits) designated by Ⓢ3 is within the range of upper and lower limit values specified by Ⓢ1 and Ⓢ2 or not.

Output value is controlled in the way shown below:

- When Ⓢ1 Lower limit value > Ⓢ3 Input value ..... Ⓢ1 Lower limit value → Ⓧ Output value
- When Ⓢ2 Upper limit value < Ⓢ3 Input value ..... Ⓢ2 Upper limit value → Ⓧ Output value
- When Ⓢ1 Lower limit value ≤ Ⓢ3 Input value ≤ Ⓢ2 Upper ..... Ⓢ3 Input value Ⓧ → Output value



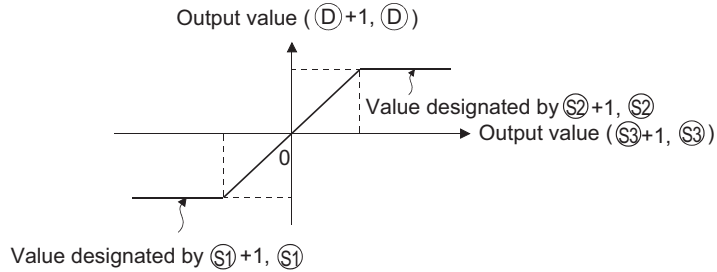
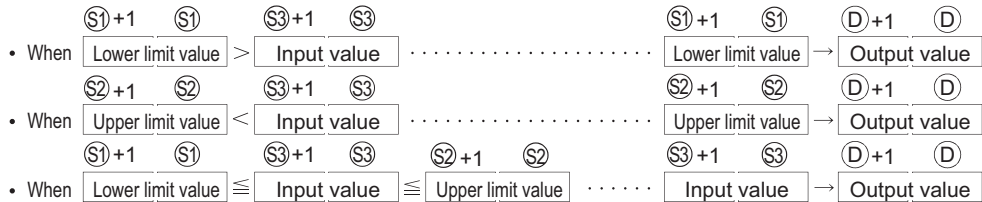
- (2) Values in the range from -32768 and 32767 can be designated at Ⓢ1, Ⓢ2, and Ⓢ3.

7  
7.13 Data Control Instructions  
7.13.1 Upper and lower limit controls for BIN 16-bit and BIN 32-bit data (LIMIT(P),DLIMIT(P))

- (3) When control based only on upper limit values is performed, the lower limit value designated at  $S1$  is set at "-32678".
- (4) When control based only on lower limit values is performed, the upper limit value designated at  $S2$  is set at "32767".

**DLIMIT**

- (1) The function controls the output value to be stored at the device designated by  $(D, D+1)$  by checking whether the input value (BIN 32 bits) designated by  $(S3, S3+1)$  is within the range of upper and lower limit values specified by  $(S1, S1+1)$  and  $(S2, S2+1)$  or not.



- (2) The values designated by  $(S1, S1+1)$ ,  $(S2, S2+1)$ , or  $(S3, S3+1)$  are within the range of -2147483648 to 2147483647.
- (3) To perform controls based only on the upper limit value, set the lower limit value designated by  $(S1, S1+1)$  to "-2147483648".
- (4) To perform controls based only on the lower limit value, set the upper limit value designated by  $(S2, S2+1)$  to "2147483647".



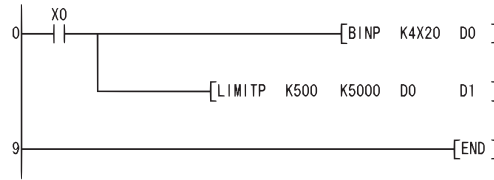
**Operation Error**

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The lower limit value designated by  $S1$  is larger than the upper limit value designated by  $S2$ . (Error code: 4100)

## Program Example

- (1) The following program conducts limit controls from 500 to 5000 on the data set as BCD values from X20 to X2F, and stores the result at D1 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	BINP	K4X20 D0
4	LIMITP	K500 K5000 D0 D1
9	END	

[Operation]

- D1 becomes 500 if  $D0 < 500$ .

**Example**  $D0=400 \rightarrow D1=500$

- D1 becomes the value of D0 when  $500 \leq D0 \leq 5000$ .

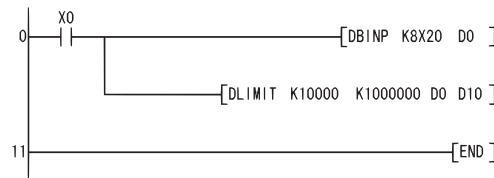
**Example**  $D0=1300 \rightarrow D1=1300$

- D1 becomes 5000 when  $5000 < D0$ .

**Example**  $D0=9600 \rightarrow D1=5000$

- (2) The following program conducts limit value controls from 10000 to 1000000 on the data set as BCD values from X20 to X3F when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DBINP	K8X20 D0
4	DLIMIT	K10000 K1000000 D0 D10
11	END	

[Operation]

- (D11, D10) become 10000 if (D1, D0) are less than 10000.

**Example**  $(D1,D0)=400 \rightarrow (D11,D10)=10000$

- (D11, D10) become the value of (D1, D0) if  $10000 \leq (D1, D0) \leq 1000000$ .

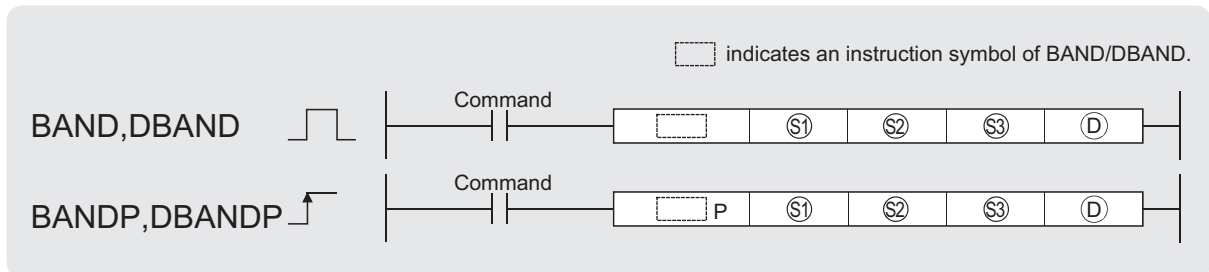
**Example**  $(D1,D0)=345678 \rightarrow (D11,D10)=345678$

- (D11, D10) become 1000000 if  $1000000 < (D1, D0)$ .

**Example**  $(D1,D0)=9876543 \rightarrow (D11,D10)=1000000$

## 7.13.2 BIN 16-bit and 32-bit dead band controls (BAND(P),DBAND(P))

Basic High performance Process Redundant Universal



- Ⓢ1 : Lower limit value of dead band (no output band) (BIN 16/32 bits)
- Ⓢ2 : Upper limit value of dead band (no output band) (BIN 16/32 bits)
- Ⓢ3 : Input value to be controlled by a dead band control (BIN 16/32 bits)
- Ⓣ : Head number of the devices where the output value controlled by the dead band control will be stored (BIN 16/32 bits)

Setting Data	Internal Devices		R, ZR	JDO		U:GO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1					○			○	—
Ⓢ2					○			○	—
Ⓢ3					○			○	—
Ⓣ					○			—	—

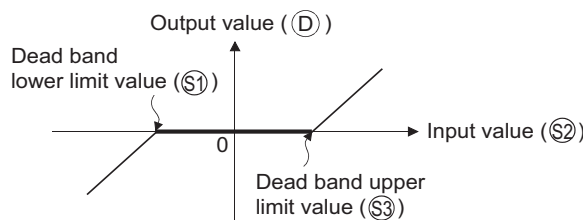
### ★ Function

#### BAND

- (1) Controls the output value to be stored at the device designated by Ⓣ by checking whether the input value (BIN 16 bits) designated by Ⓢ3 is within the range of dead band upper and lower limit values specified by Ⓢ1 and Ⓢ2 or not.

Output value is controlled in the way shown below:

- When Ⓢ1 Lower limit value > Ⓢ3 Input value ..... Ⓢ3 Input value - Ⓢ1 Lower → Ⓣ Output value
- When Ⓢ2 Upper limit value < Ⓢ3 Input value ..... Ⓢ3 Input value - Ⓢ2 Upper → Ⓣ Output value
- When Ⓢ1 Lower limit value ≤ Ⓢ3 Input value ≤ Ⓢ2 Upper ..... 0 → Ⓣ Output value



- (2) The values that can be designated by Ⓢ1, Ⓢ2, and Ⓢ3 are in the range of from -32768 to 32767.

- (3) The output value stored at (D) is a signed 16-bit BIN value. Therefore, if the operation results exceed the range of from -32768 to 32767, the following will take place:

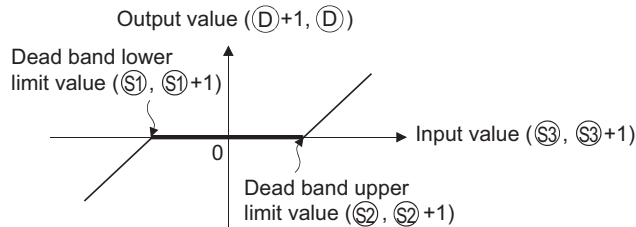
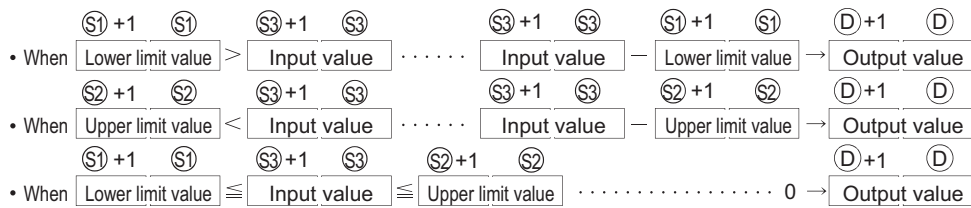
$$\text{When : } \left\{ \begin{array}{l} \text{Dead band lower limit value } (S1) \dots\dots\dots 10 \\ \text{Input value } (S3) \dots\dots\dots -32768 \end{array} \right.$$

$$\text{Output value} = -32768 - 10 = 8000_H - A_H = 7FF6_H = 32758$$

**DBAND**

- (1) Controls the output value to be stored at the device designated by (D) by checking whether the input value (BIN 32 bits) designated by (S3, S3 + 1) is within the range of dead band upper and lower limit values specified by (S1, S1 + 1) and (S2, S2 + 1) or not.

Output value is controlled in the way shown below:



- (2) The values designated by (S1, S1 + 1), (S2, S2 + 1), or (S3, S3 + 1) are within the range of from -2147483648 to 2147483647.

- (3) The output value stored at (D), (D + 1) is a signed 32-bit BIN value. Therefore, if the operation results exceed the range of from -2147483648 to 2147483647, the following takes place:

$$\text{When : } \left\{ \begin{array}{l} \text{Dead band lower limit value } (S1, S1+1) \dots\dots 1000 \\ \text{Input value } (S3, S3+1) \dots\dots\dots -2147483648 \end{array} \right.$$

$$\text{Output value} = -2147483648 - 1000 = 80000000_H - 000003E8_H$$

$$= 7FFFC18_H = 2147482648$$

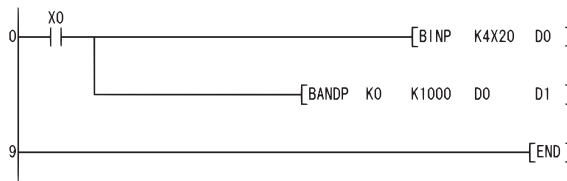
**! Operation Error**

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The lower limit value designated by (S1) is greater than the upper limit value designated by (S2). (Error code: 4100)

## Program Example

- (1) The following program performs the dead band control by applying the lower and upper limits of 0 and 1000 for the data set in BCD at X20 to X2F and stores the result of control at D1 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	BINP	K4X20 D0
4	BANDP	K0 K1000 D0 D1
9	END	

[Operation]

- "0" is stored at D1 if  $0 \leq D0 \leq 1000$ .

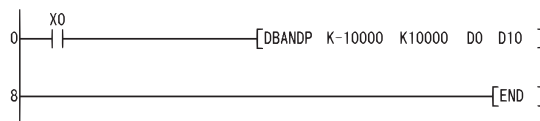
**Example**  $D0=500 \rightarrow D1=0$

- The value of  $(D0) - 1000$  is stored at D1 if  $1000 < D0$ .

**Example**  $D0=7000 \rightarrow D1=6000$

- (2) The following program performs the dead band control by applying the lower and upper limits of  $-10000$  and  $10000$  for the data set at D0 and D1 and stores the result of control at D10 and D11 when X0 is turned ON

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DBANDP	K-10000 K10000 D0 D10
8	END	

[Operation]

- The value  $(D1, D0) - (-10000)$  is stored at  $(D11, D10)$  if  $(D1, D0) < (-10000)$ .

**Example**  $(D1, D0) = -12345 \rightarrow (D11, D10) = -2345$

- The value 0 is stored at  $(D11, D10)$  if  $-10000 \leq (D1, D0) \leq 10000$ .

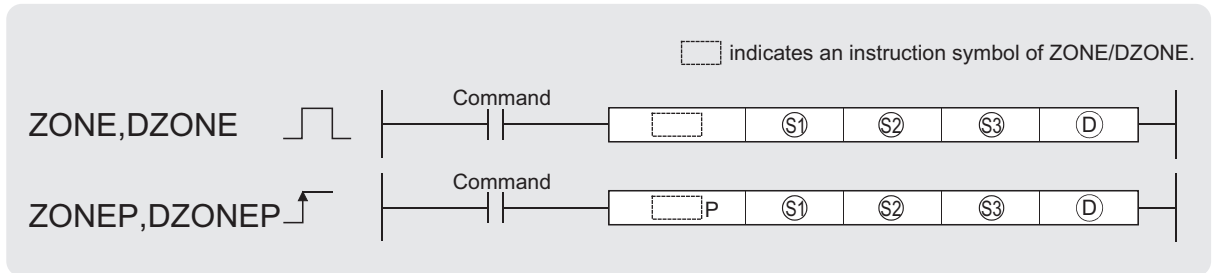
**Example**  $(D1, D0) = 6789 \rightarrow (D11, D10) = 0$

- The value  $(D1, D0) - 10000$  is stored at  $(D11, D10)$  if  $10000 < (D1, D0)$ .

**Example**  $(D1, D0) = 50000 \rightarrow (D11, D10) = 40000$

### 7.13.3 Zone control for BIN 16-bit and BIN 32-bit data (ZONE(P),DZONE(P))

Basic High performance Process Redundant Universal



- Ⓢ1 : Negative bias value to be added to an input value (BIN 16/32 bits)
- Ⓢ2 : Positive bias value to be added to an input value (BIN 16/32 bits)
- Ⓢ3 : Input value used for a zone control (BIN 16/32 bits)
- Ⓧ : Head number of the devices where the output value controlled by the zone control will be stored (BIN 16/32 bits).

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1					○			○	—
Ⓢ2					○			○	—
Ⓢ3					○			○	—
Ⓧ					○			—	—

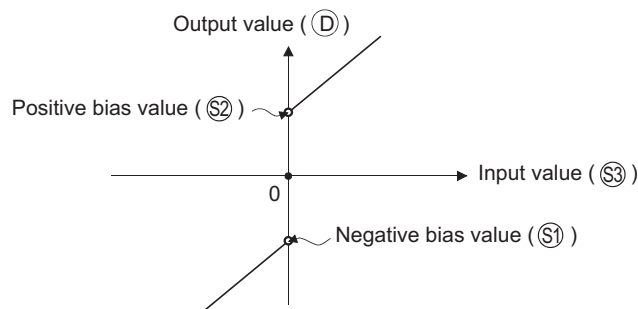
## ★ Function

### ZONE

- (1) Adds bias value designated by Ⓢ1 or Ⓢ2 to input value designated by Ⓢ3, and stores at device number designated by Ⓧ.

Bias values are calculated in the following manner:

- When Ⓢ3 Input value < 0..... Ⓢ3 Input value + Ⓢ1 Negative bias value → Ⓧ Output value
- When Ⓢ3 Input value = 0..... 0 → Ⓧ Output value
- When Ⓢ3 Input value > 0..... Ⓢ3 Input value + Ⓢ2 Positive bias value → Ⓧ Output value



- (2) The values that can be designated by (S1), (S2), and (S3) are in the range of from -32768 to 32767.
- (3) The output value stored at (D) is a signed 16-bit BIN value. Therefore, if the operation results exceed the range of -32768 to 32767, the following will take place:

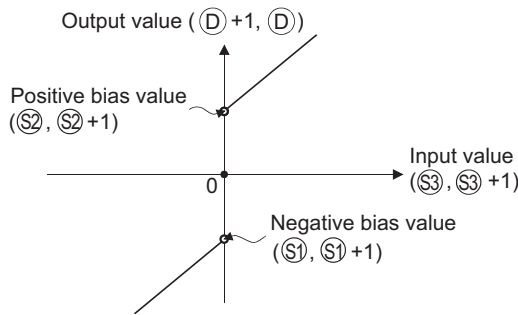
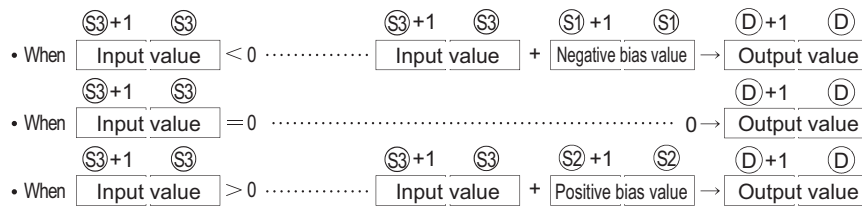
When :  $\left\{ \begin{array}{l} \text{Negative bias value (S1) ..... -100} \\ \text{Input value (S3) ..... -32768} \end{array} \right.$

Output value = -32768 + (-100) = 8000<sub>H</sub> + FF9C = 7F9C<sub>H</sub> = 32668

**DZONE**

- (1) Adds bias value designated by ((S1), (S1)+1) or ((S2), (S2)+1) to input value designated by ((S3), (S3)+1), and stores the result at device number designated by ((D), (D)+1).

Addition of the bias value is performed as follows:



- (2) The values designated by ((S1), (S1)+1), ((S2), (S2)+1), or ((S3), (S3)+1) are within the range of from -2147483648 to 2147483647.
- (3) The value stored at ((D), (D)+1) is a signed 32-bit BIN value. Therefore, if the operation results exceed the range of from -2147483648 to 2147483647, the following takes place:

When :  $\left\{ \begin{array}{l} \text{Negative bias value ((S1), (S1)+1)..... -1000} \\ \text{Input value ((S3), (S3)+1) ..... -2147483648} \end{array} \right.$

Output value = -2147483648 + (-1000) = 80000000<sub>H</sub> + FFFFC18<sub>H</sub>  
 = 7FFFC18 = 2147482648.

**Operation Error**

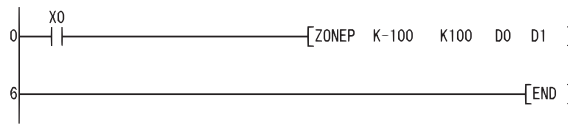
- (1) There are no operation errors associated with the ZONE(P) or DZONE(P) instructions.



## Program Example

- (1) The following program performs zone control by applying negative and positive bias values of  $-100$  to  $100$  for the data set at  $D0$  and stores the result of control at  $D1$  when  $X0$  is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ZONEP	K-100 K100 D0 D1
6	END	

[Operation]

- The value  $(D0) + (-100)$  is stored at  $D1$  if  $D0 < 0$ .

**Example**  $D0 = -200 \rightarrow D1 = -300$

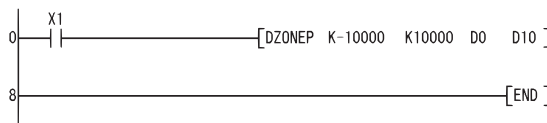
- The value  $0$  is stored at  $D1$  if  $D0 = 0$ .

- The value of  $(D0) + 100$  is stored at  $D1$  if  $0 < D0$ .

**Example**  $D0 = 700 \rightarrow D1 = 800$

- (2) The following program performs zone control by applying negative and positive bias values of  $-10000$  to  $10000$  for the data set at  $D0$  and  $D1$  and stores the result of control at  $D10$  and  $D11$  when  $X1$  is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1
1	DZONEP	K-10000 K10000 D0 D10
8	END	

[Operation]

- The value  $(D1, D0) + (-10000)$  is stored at  $(D11, D10)$  if  $(D1, D0) < 0$ .

**Example**  $(D1, D0) = -12345 \rightarrow (D11, D10) = -22345$

- The value  $0$  is stored at  $(D11, D10)$  if  $(D1, D0) = 0$ .

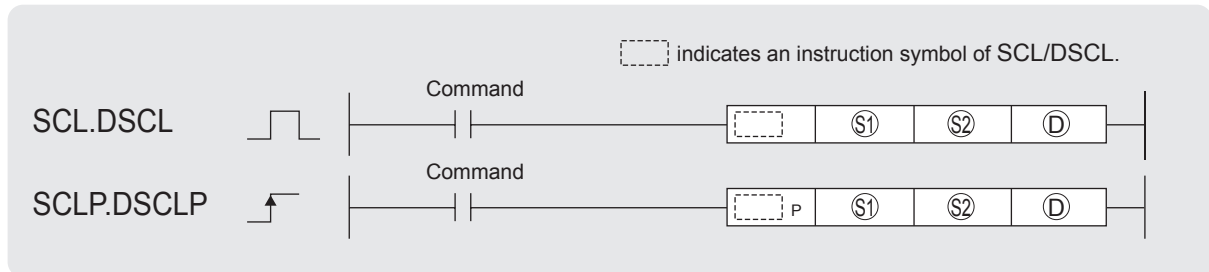
- The value  $(D1, D0) + 10000$  is stored at  $(D11, D10)$  if  $0 < (D1, D0)$ .

**Example**  $(D1, D0) = 50000 \rightarrow (D11, D10) = 60000$

## 7.13.4 Scaling (Point-by-point coordinate data) (SCL(P),DSCL(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.  
QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ<sub>1</sub> : Input values for scaling or head number of the device where input values are stored(BIN 16/32 bits)

Ⓢ<sub>2</sub> : Head number of the devices where scaling conversion data are stored(BIN 16/32 bits)

Ⓧ : Head number of the devices where output values depending on scaling are stored(BIN 16/32 bits).

Setting Data	Internal Devices		R, ZR	J:GO		U:GO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○	○			○		○	—
Ⓢ <sub>2</sub>	—	○	○			—		—	—
Ⓧ	—	○	○			○		—	—

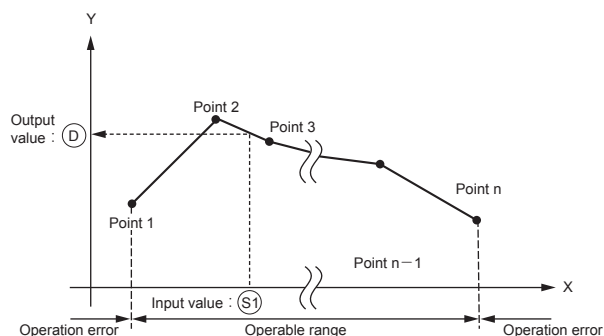
### ★ Function

#### SCL(P)

- This instruction executes scaling for the scaling conversion data (16-bit data units) specified by Ⓢ<sub>2</sub> with the input value specified by Ⓢ<sub>1</sub>, and then stores the operation result into the devices specified by Ⓧ. The scaling conversion is executed based on the scaling conversion data stored in the device specified by Ⓢ<sub>2</sub> and up.

Setting item	Device assignment	
Number of coordinate points	Ⓢ <sub>2</sub>	
Point 1	X coordinate	Ⓢ <sub>2</sub> +1
	Y coordinate	Ⓢ <sub>2</sub> +2
Point 2	X coordinate	Ⓢ <sub>2</sub> +3
	Y coordinate	Ⓢ <sub>2</sub> +4
⋮		
Point n	X coordinate	Ⓢ <sub>2</sub> +2n-1
	Y coordinate	Ⓢ <sub>2</sub> +2n

※n indicates the number of coordinates specified by (S2).



- If the value does not result in an integer, this instruction rounds the value to the whole number.
- Set the X coordinate of the scaling conversion data in ascending order.
- Set the input value Ⓢ<sub>1</sub> within the range of the scaling conversion data (within the range of Ⓢ<sub>2</sub> devices).

- (5) If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.
- (6) Specify the number of coordinate points of scaling conversion data from 1 to 32767.

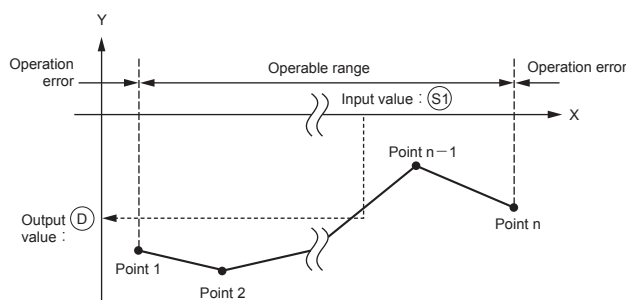
## DSCL(P)

- (1) This instruction executes scaling for the scaling conversion data (32-bit data units) specified by  $\text{S}2$  with the input value specified  $\text{S}1$ , and then stores the operation result into the devices specified by  $\text{D}$ .  
The scaling conversion is executed based on the scaling conversion data stored in the device specified by  $\text{S}2$  and up.

Scaling conversion data component

Setting item	Device assignment
Number of coordinate points	$\text{S}2 + 1$ , $\text{S}2$
Point 1	X coordinate $\text{S}2 + 3$ , $\text{S}2 + 2$
	Y coordinate $\text{S}2 + 5$ , $\text{S}2 + 4$
Point 2	X coordinate $\text{S}2 + 7$ , $\text{S}2 + 6$
	Y coordinate $\text{S}2 + 9$ , $\text{S}2 + 8$
Point n	X coordinate $\text{S}2 + 4n - 1$ , $\text{S}2 + 4n - 2$
	Y coordinate $\text{S}2 + 4n + 1$ , $\text{S}2 + 4n$

\*n indicates the number of coordinates specified by (S2).



- (2) If the value does not result in an integer, this instruction rounds the value to the whole number.
- (3) Set the X coordinate of the scaling conversion data in ascending order.
- (4) Set the input value  $\text{S}1$  within the range of the scaling conversion data (within the range of  $\text{S}2$  and  $\text{S}2 + 1$  devices).
- (5) If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.
- (6) Specify the number of coordinate points of scaling conversion data from 1 to 32767.

**POINT**

(1) There are two searching methods that depend on whether SM750 is on or off.

SM750	Searching method	Range of number of searches
OFF	Sequential search	1 ≦ Number of times ≦ 32767
ON	Binary search	1 ≦ Number of times ≦ 15

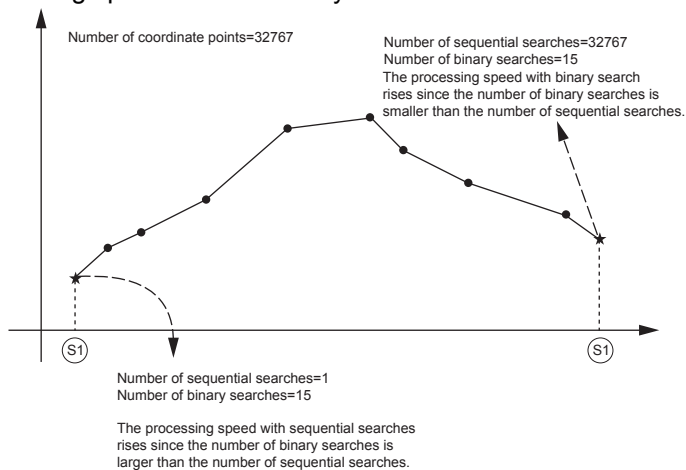
(2) When the scaling conversion data are set in ascending order, the searching methods change from one to the other depending on the SM750 status. Therefore, the processing speed also changes. The number of searches determines the processing speed. Fewer number of searches make the processing run faster.

(a) If the data processing speed with the sequential search rises:

If the number of coordinates is highest and the input value (S1) is within the coordinate range from 1 to 15 point, the number of sequential searches will be 15 or smaller. Therefore, the data processing speed with the sequential search will rise.

(b) If the data processing speed with the binary search rises:

If the maximum number of searches is 15 and the input value (S1) is out of the coordinate range, 16 or over, the number of binary searches will be equal to the number of sequential numbers or smaller. Therefore, the data processing speed with the binary search will rise.



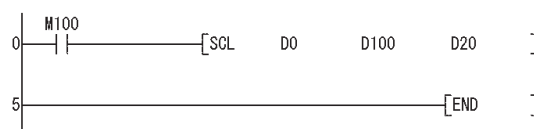
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The X coordinates of the scaling conversion data positioned before the point specified by (S1) are not set in ascending order. (However, this error is not detected when SM750 is on.) (Error code: 4100)
  - The input value specified by (S1) is out of the range of the scaling conversion data set. (Error code: 4100)
  - The number of X and Y coordinates of the device specified by (S2) is out of the range from 1 to 32767. (Error code: 4100)
  - The number of X and Y coordinates of the device specified by (S2) is out of the specified range. (Error code: 4101)

## Program Example

- (1) The following program executes scaling for the scaling conversion data of which the devices specified at D100 and up are set with the input value specified at D0, and then outputs the data at D20.

[Ladder Mode]



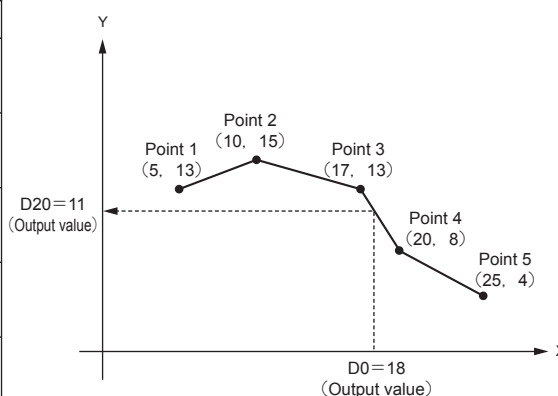
[List Mode]

Step	Instruction	Device
0	LD	M100
1	SCL	D0      D100    D20
5	END	

[Operation]

Scaling conversion data component

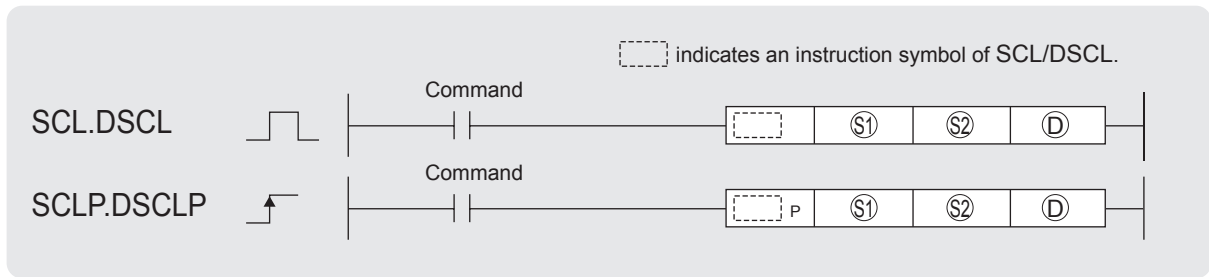
Setting item	Device	Setting contents
Number of coordinate points	D100	K5
Point 1	X coordinate	D101    K5
	Y coordinate	D102    K13
Point 2	X coordinate	D103    K10
	Y coordinate	D104    K15
Point 3	X coordinate	D105    K17
	Y coordinate	D106    K13
Point 4	X coordinate	D107    K20
	Y coordinate	D108    K8
Point 5	X coordinate	D109    K25
	Y coordinate	D110    K22



# 7.13.5 Scaling (Point-by-point coordinate data) (SCL2(P),DSCL2(P))



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.  
QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



- Ⓢ1 : Input values for scaling or head number of the device where input values are stored(BIN 16/32 bits)
- Ⓢ2 : Head number of the devices where scaling conversion data are stored(BIN 16/32 bits)
- Ⓧ : Head number of the devices where output values depending on scaling are stored(BIN 16/32 bits).

Setting Data	Internal Devices		R, ZR	J:DO		U:GO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○	○			○		○	—
Ⓢ2	—	○	○			—		—	—
Ⓧ	—	○	○			○		—	—

## ★ Function

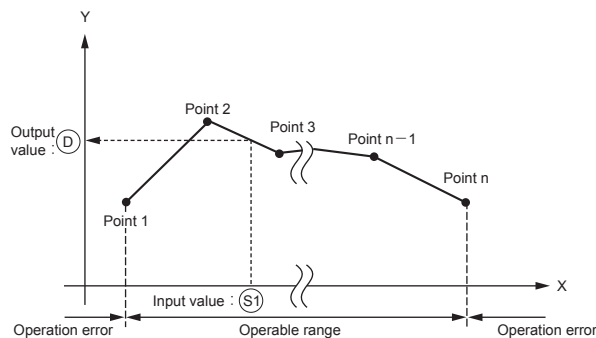
### SCL2

- (1) This instruction executes scaling for the scaling conversion data (16-bit data units) specified by Ⓢ2 with the input value specified by Ⓢ1, and then stores the operation result into the devices specified by Ⓧ. The scaling conversion is executed based on the scaling conversion data stored in the device specified by Ⓢ2 and up.

Scaling conversion data component

Setting item	Device assignment
Number of coordinate points	Ⓢ2
X coordinate	Point 1 Ⓢ2 +1
	Point 2 Ⓢ2 +2
	⋮
	Point n Ⓢ2 +n
Y coordinate	Point 1 Ⓢ2 +n+1
	Point 2 Ⓢ2 +n+2
	⋮
	Point n Ⓢ2 +2n

\*n indicates the number of coordinates specified by (S2).



- (2) If the value does not result in an integer, this instruction rounds the value to the whole number.
- (3) Set the X coordinate of the scaling conversion data in ascending order.

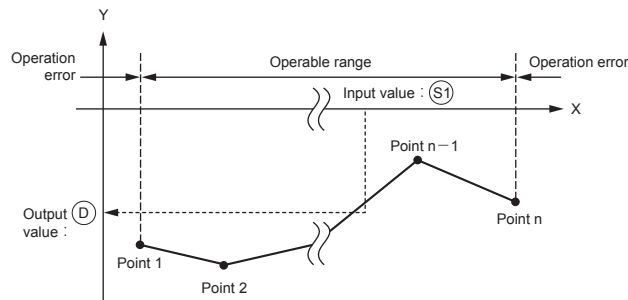
- (4) Set the input value (S1) within the range of the scaling conversion data (within the range of (S2) devices).
- (5) If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.

## DSCL2(P)

- (1) This instruction executes scaling for the scaling conversion data (32-bit data units) specified by (S2) with the input value specified (S1), and then stores the operation result into the devices specified by (D).  
The scaling conversion is executed based on the scaling conversion data stored in the device specified by (S2) and up.

Scaling conversion data component	
Setting item	Device assignment
Number of coordinate points	(S2) +1, (S2)
X coordinate	Point 1 (S2) +3, (S2) +2
	Point 2 (S2) +5, (S2) +4
	Point n (S2) +2n+1, (S2) +2n
Y coordinate	Point 1 (S2) +2n+3, (S2) +2n+2
	Point 2 (S2) +2n+5, (S2) +2n+4
	Point n (S2) +4n+1, (S2) +4n

※n indicates the number of coordinates specified by (S2).



- (2) If the value does not result in an integer, this instruction rounds the value to the whole number.
- (3) Set the X coordinate of the scaling conversion data in ascending order.
- (4) Set the input value (S1) within the range of the scaling conversion data (within the range of (S2) and (S2) +1 devices).
- (5) If some specified points have same X coordinates, the Y coordinate data of the highest point number will be output.
- (6) Specify the number of coordinate points of scaling conversion data from 1 to 32767.

### POINT

When the coordinates of the scaling conversion data are set in ascending order, the searching methods change from one to the other depending on the SM750 status. Therefore, the processing speed also change. The number of searches determines the processing speed. Fewer number of searches make the processing run faster.

For details, refer to Section 7.13.4.

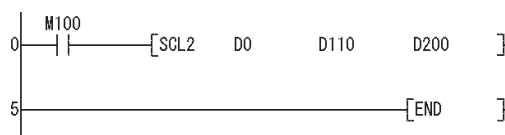
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The X coordinates are not set in ascending order. (Error code: 4100)
  - The input value specified by  $\text{S1}$  is out of the range of the scaling conversion data set. (Error code: 4100)
  - The number of X and Y coordinates of the device specified by  $\text{S2}$  is out of the range from 1 to 32767. (Error code: 4100)
  - The number of X and Y coordinates of the device specified by  $\text{S2}$  is out of the specified range. (Error code: 4101)

## Program Example

- (1) The following program executes scaling for the scaling conversion data of which the devices specified at D100 and up are set with the input value specified at D0, and then outputs the data at D20.

[Ladder Mode]



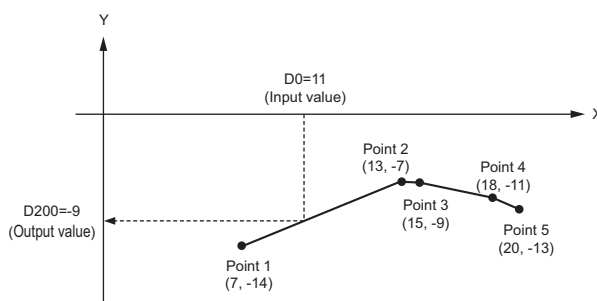
[List Mode]

Step	Instruction	Device
0	LD	M100
1	SCL2	D0            D110            D200
5	END	

[Operation]

Scaling conversion data component

Setting item	Device	Setting contents
Number of coordinate points	D110	K5
X coordinate	Point 1	D111    K7
	Point 2	D112    K13
	Point 3	D113    K15
	Point 4	D114    K18
	Point 5	D115    K20
Y coordinate	Point 1	D116    K-14
	Point 2	D117    K-7
	Point 3	D118    K-15
	Point 4	D119    K-11
	Point 5	D120    K-18

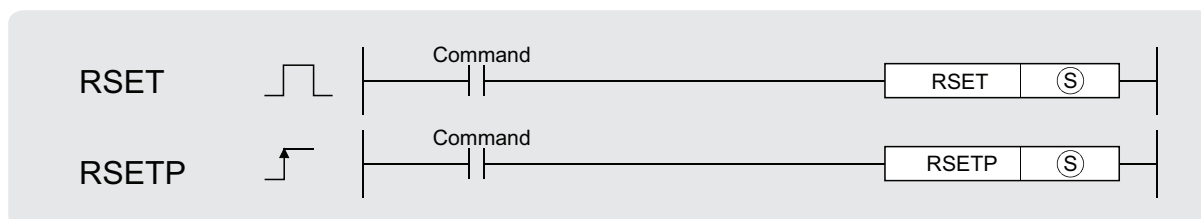




## 7.14 File register switching instructions

### 7.14.1 Switching file register numbers (RSET(P))

Basic High performance Process Redundant Universal



Ⓢ : Block number data used to change the block number or the number of the device where the block number data is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ									—

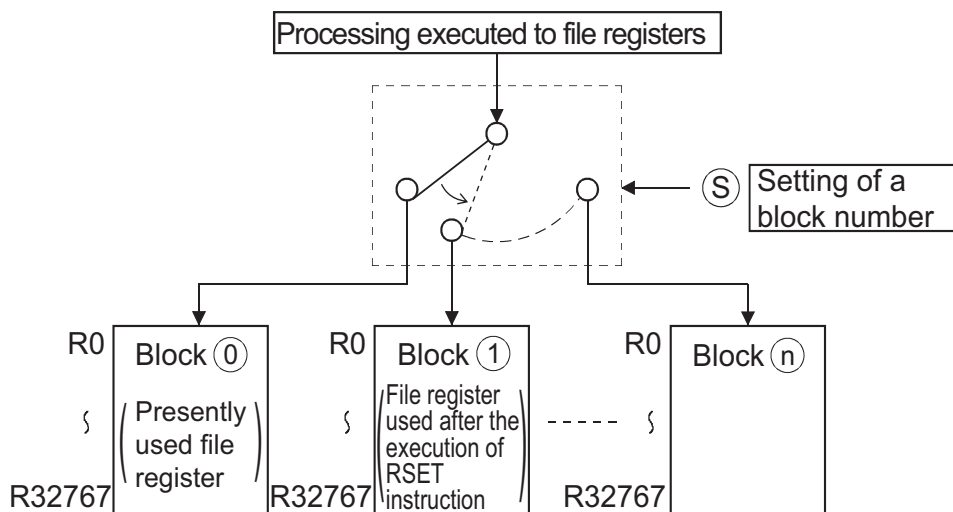
#### ★ Function

- Changes the file register block number used in the program to the block number stored in the device designated at Ⓢ.

Following the block number change, all file registers used in the sequence program are processed to the file register of the block number after the change.

#### Example

When switching block number from block No. 0 to block No. 1



#### ☒ POINT

When a file register (R) is refreshed and the block No. of the file register is switched with the RSET instruction, follow restrictions.  
For the restrictions on file registers, refer to Section 3.10.

## Operation Error

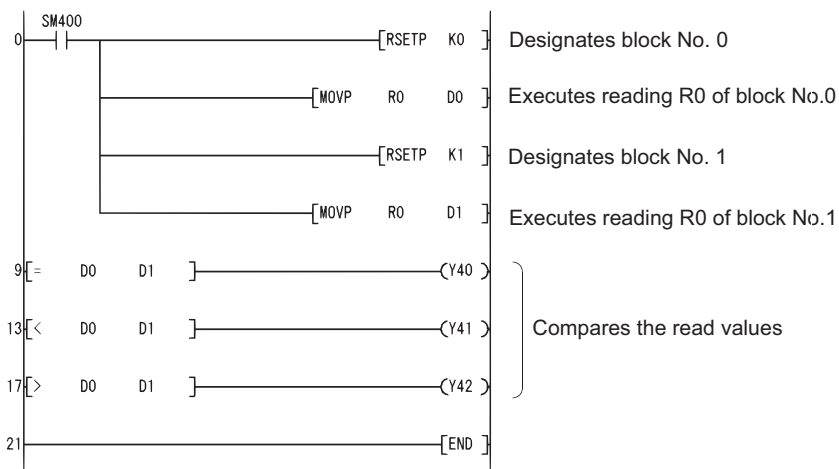
(1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- The block number designated by ⑤ does not exist. (Error code: 4100)
- There is no file register for the specified block No. (Error code: 4101)

## Program Example

(1) The following program compares R0 of block No. 0 and block No. 1.

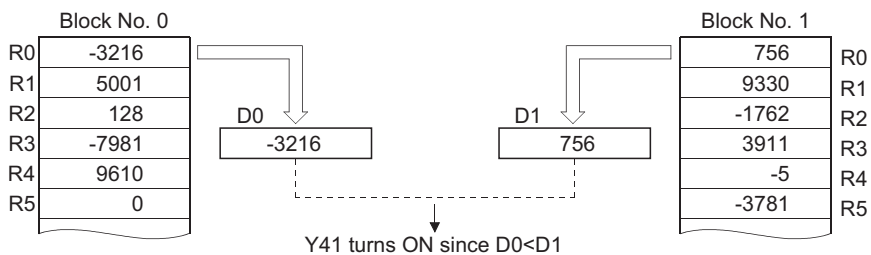
[Ladder Mode]



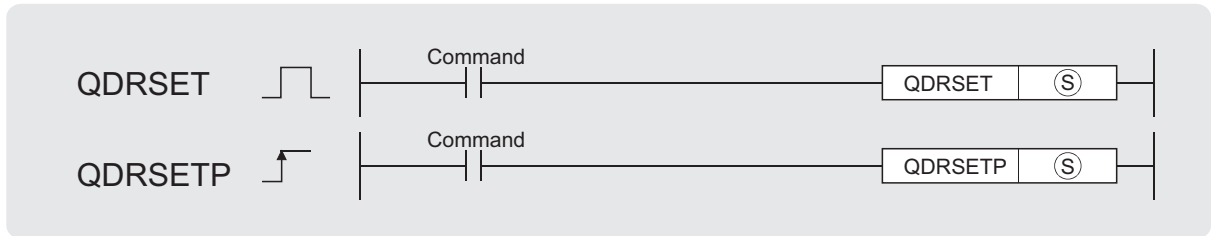
[List Mode]

Step	Instruction	Device
0	LD	SM400
1	RSETP	K0
3	MOV P	R0 D0
5	RSETP	K1
7	MOV P	R0 D1
9	LD=	D0 D1
12	OUT	Y40
13	LD<	D0 D1
16	OUT	Y41
17	LD>	D0 D1
20	OUT	Y42
21	END	

[Operation]



## 7.14.2 Setting files for file register use (QDRSET(P))



Ⓢ : Character string data of the drive No./file name in which the file register is set, or head number of the devices where the character string data is stored (character string)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—

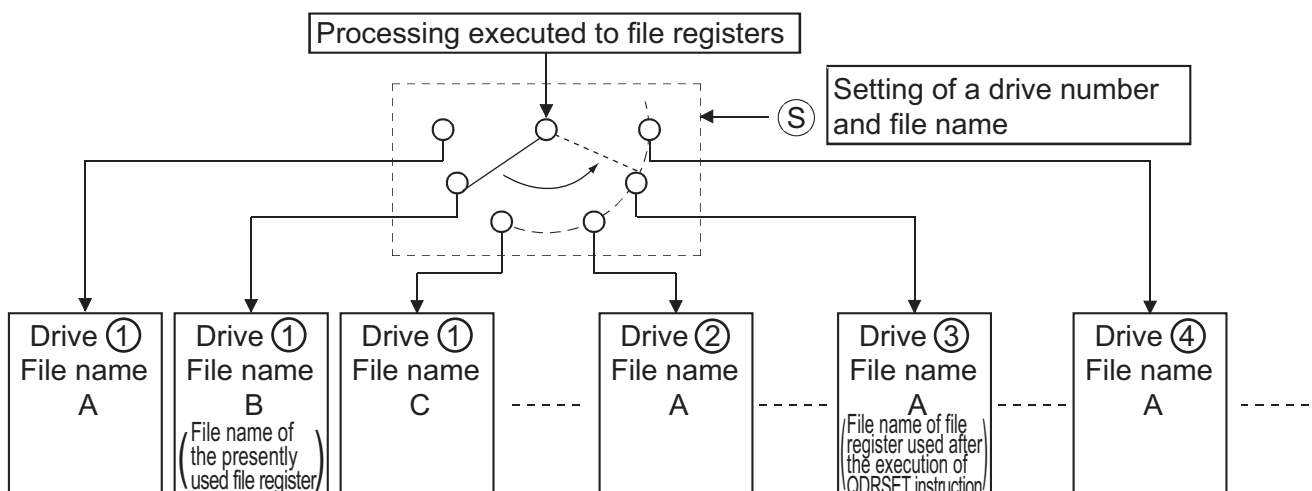
### ★ Function

- Changes the file register file name used in the program to the file name being stored at the device designated by Ⓢ.

After the file names have been changed, all the file registers being used by the sequence program process the file register of the block No. 0 of the renamed file. Block number switches are performed by the RSET instruction.

#### Example

When switching from Drive No. 1/File name B to Drive No. 3/File name A



- (2) Drive number can be designated from 1 to 4.  
(The drive number cannot be designated as drive 0 (program memory/internal memory).)  
Note that available drives vary depending on the CPU module used.  
Refer to the manual of the CPU module and check the drives that can be specified.
- (3) It is not necessary to designate the extension (.QDR) with the file name.
- (4) A file name setting can be deleted by designating the NULL character (00<sub>H</sub>) for the file name.
- (5) File names designated with this instruction will be given priority even if a drive number and file name have been designated in the parameters.

---

**POINT**

1. If the file name is changed with the QDRSET instruction, the file name returns to the name specified by the parameter when the CPU module is switched from STOP to RUN. To maintain the file name even after the CPU mode is changed from STOP to RUN, execute the QDRSET instruction with the SM402 special relay, which turns ON during one scan when the CPU enters from STOP to RUN mode.
  2. For refreshing a file register, do not change the file name of the file register with the QDRSET instruction. For restrictions on file registers, refer to Section 3.10.
- 

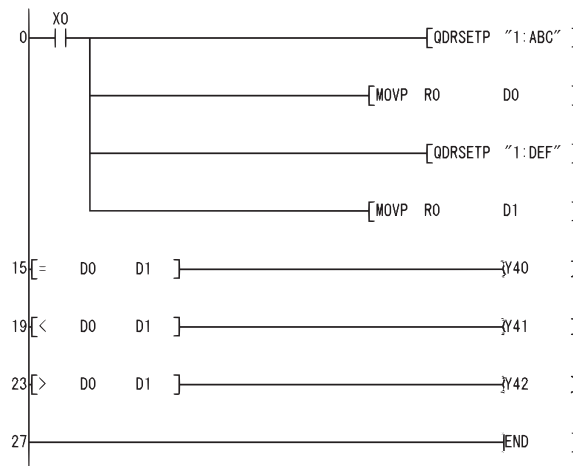


## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - File name does not exist at the drive number designated by ⑤. (Error code: 2410)

## Program Example

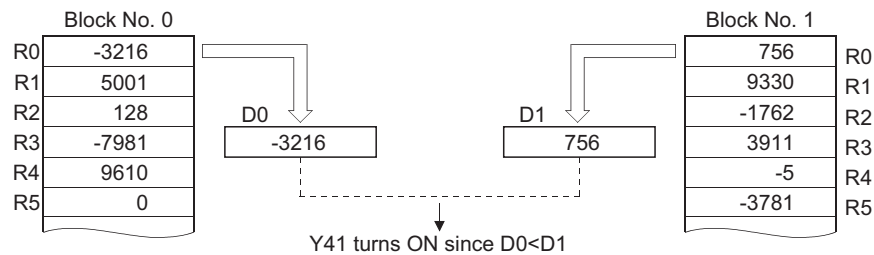
- (1) The following program compares R0 of ABC in block No. 1 and R0 of DEF in block No. 1.  
[Ladder Mode]



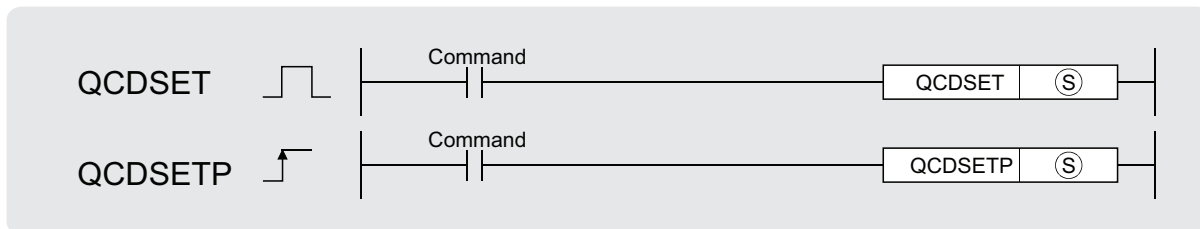
[List Mode]

Step	Instruction	Device
0	LD	X0
1	QDRSETP	"1:ABC"
6	MOV P	R0 D0
8	QDRSETP	"1:DEF"
13	MOV P	R0 D1
15	LD =	D0 D1
18	OUT	Y40
19	LD <	D0 D1
22	OUT	Y41
23	LD >	D0 D1
26	OUT	Y42
27	END	

[Operation]



## 7.14.3 File setting for comments (QCDSET(P))



Ⓢ : Character string data of the drive No./file name in which the comment file is set, or head number of the devices where the character string data is stored (character string)

Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—

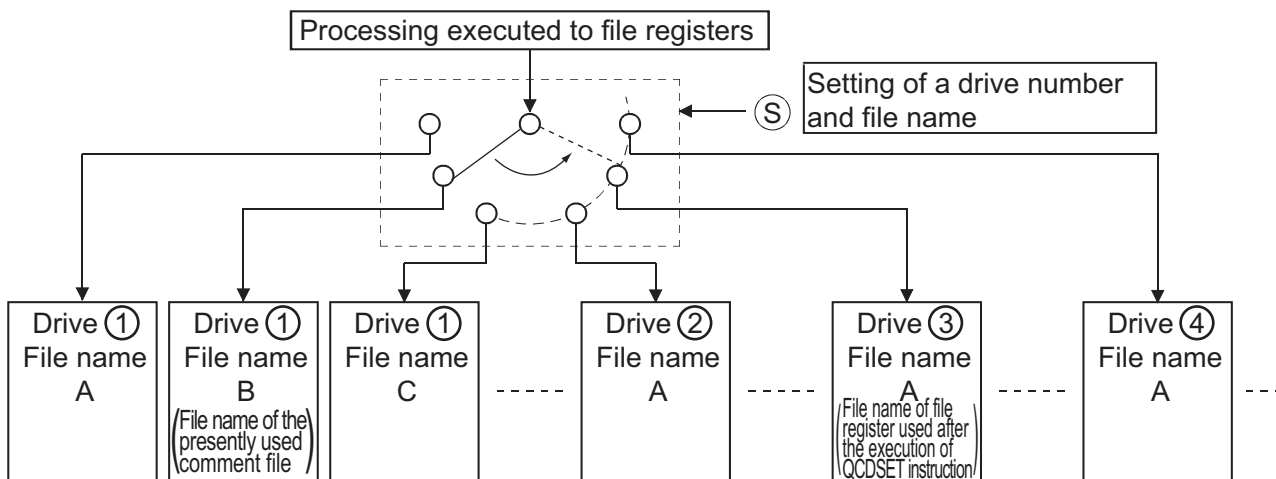
### ★ Function

- (1) Changes the file register file name used in the program to the file name being stored at the device designated by Ⓢ.

After the file name change, comment data being used by the sequence program perform processing in relation to the comment data of the file name after the change.

#### Example

When switching from Drive No. 1/File name B to Drive No. 3/File name A



- (2) Drive number can be designated from 1 to 4.  
(The drive number cannot be designated as drive 0 (program memory/internal memory).)  
Note that available drives vary depending on the CPU module used.  
Refer to the manual of the CPU module and check the drives that can be specified.
- (3) It is not necessary to designate the extension (.QCD) with the file name.

- (4) A file name setting can be deleted by designating the NULL character (00<sub>H</sub>) for the file name.
- (5) File names designated with this instruction will be given priority even if a drive number and file name have been designated in the parameters.
- (6) This instruction cannot be executed while SM721 is ON for the Universal model QCPU. No operation if executed.

### POINT

If the file name is changed with the QCDSET instruction, the file name returns to the name specified by the parameter when the CPU module is switched from STOP to RUN.

To maintain the file name even after the CPU mode is changed from STOP to RUN, execute the QCDSET instruction with the SM402 special relay, which turns ON during one scan when the CPU enters from STOP to RUN mode.

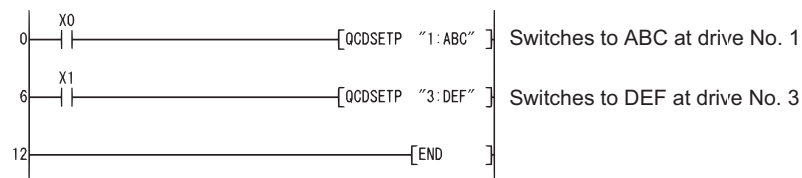
## Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - File name does not exist at the drive number designated by ⑤. (Error code: 2410)

## Program Example

- (1) The following program switches object file to file name ABC. QCD at drive No. 0 when X0 is ON, and to DEF. QCD at drive No. 1 when X1 is ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	QCDSETP	"1:ABC"
6	LD	X1
7	QCDSETP	"3:DEF"
12	END	

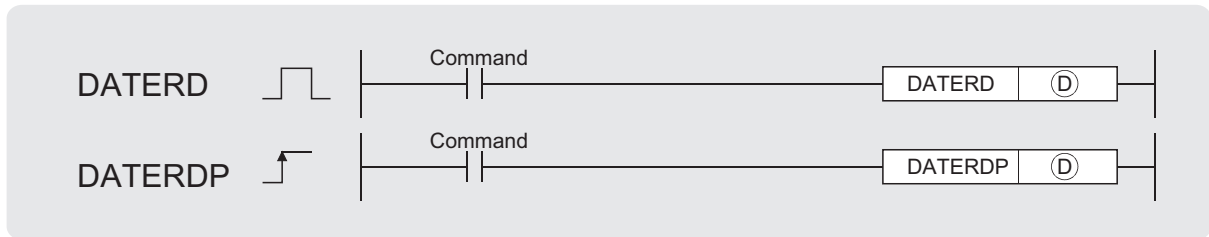
## Caution

- (1) This instruction will not be executed even when the execution command of this instruction is ON while SM721 (file access in execution) is ON for the Universal model QCPU only. Execute this instruction when SM721 is OFF.

# 7.15 Clock instructions

## 7.15.1 Reading clock data (DATERD(P))

Basic High performance Process Redundant Universal

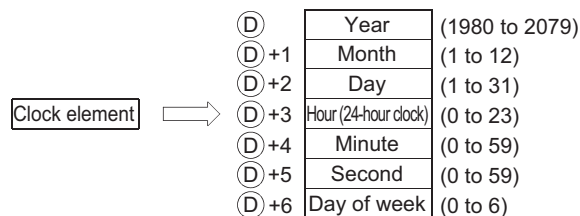


Ⓧ : Head number of the devices where the read clock data will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ	—	○					—		

### ★ Function

- (1) Reads "year, month, day, hour, minute, second, and day of week" from the clock element of the CPU module and stores it as BIN value to the device designated by Ⓧ or later device.



- (2) The "year" at Ⓧ is stored as 4-digit year indication.
- (3) The "day of week" at Ⓧ+6 is stored as 0 to 6 to represent the days Sunday to Saturday.

Day of week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
Stored data	0	1	2	3	4	5	6

- (4) Compensation is made automatically for leap years.

### ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by Ⓧ exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

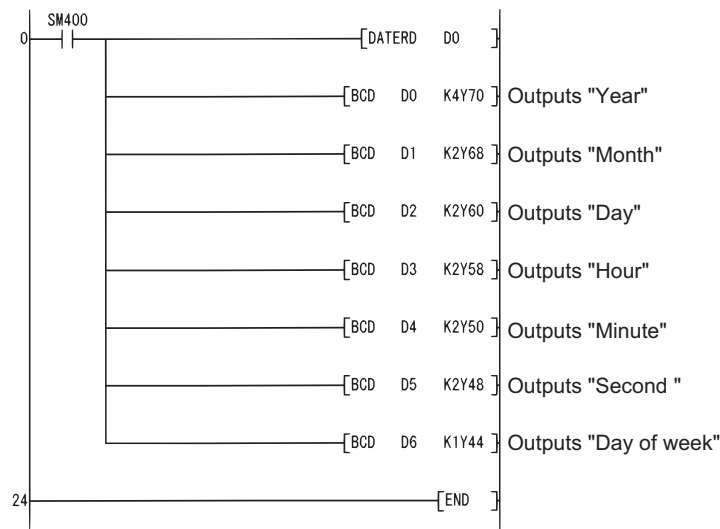


# Program Example

(1) The following program outputs the following clock data as BCD values:

- Year ..... Y70 to Y7F
- Month ..... Y68 to Y6F
- Day ..... Y60 to Y67
- Hour..... Y58 to Y5F
- Minute..... Y50 to Y57
- Second ..... Y48 to Y4F
- Week ..... Y44 to Y47

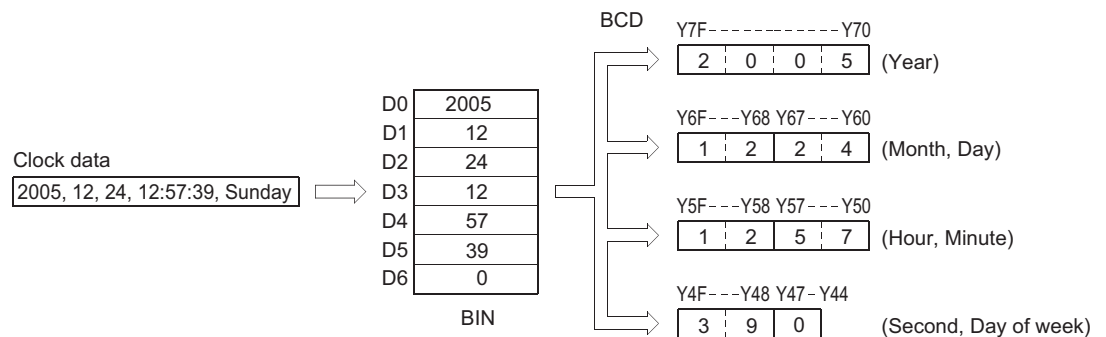
[Ladder Mode]



[List Mode]

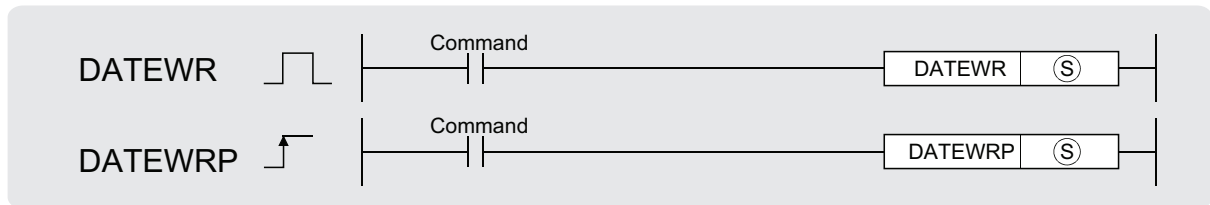
Step	Instruction	Device
0	LD	SM400
1	DATERD	D0
3	BCD	D0 K4Y70
6	BCD	D1 K2Y68
9	BCD	D2 K2Y60
12	BCD	D3 K2Y58
15	BCD	D4 K2Y50
18	BCD	D5 K2Y48
21	BCD	D6 K1Y44
24	END	

[Operation]



## 7.15.2 Writing clock data (DATEWR(P))

Basic High performance Process Redundant Universal

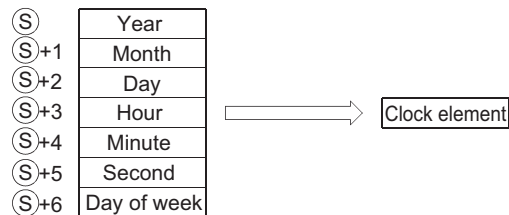


Ⓢ : Head number of the devices where clock data to be written into the clock device is stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○					—		

### ★ Function

- (1) Writes clock data stored in the device number designated by Ⓢ or later device number to the clock element of the CPU module.



- (2) Each item is set as a BIN value.
- (3) The "year" at Ⓢ is designated by using four-digit year indication between 1980 to 2079.
- (4) Ⓢ +1 designates the "month" in values of from 1 to 12 (January to December).
- (5) Ⓢ +2 designates the "day" in values of from 1 to 31.
- (6) Ⓢ +3 designates the "hour" in values of from 0 to 23 (using 24-hour clock, from 0 hours to 23 hundred hours). (Uses the 24-hour clock.)
- (7) Ⓢ +4 designates the "minute" in values of from 0 to 59.
- (8) Ⓢ +5 designates the "second" in values of from 0 to 59.
- (9) Ⓢ +6 designates the "day of week" in values of from 0 to 6 (Sunday to Saturday).

Day of week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
Stored data	0	1	2	3	4	5	6

## ! Operation Error

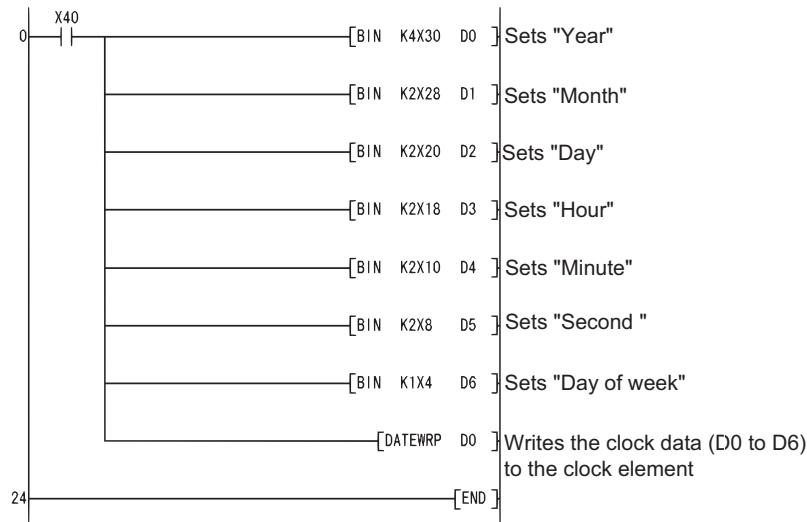
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Individual items of data have been set outside the setting range. (Error code: 4100)
  - The device specified by Ⓢ exceeds the range of the corresponding device. (Error code: 4101)  
(For the Universal model QCPU only.)

## Program Example

- (1) The following program writes the following clock data to the clock element as BCD values when X40 is turned ON.

Year ..... X30 to X3F                      Hour..... X18 to X1F  
 Month ..... X28 to X2F                    Minute..... X10 to X17  
 Day ..... X20 to X27                      Second ..... X8 to XF  
 Week ..... X4 to X7

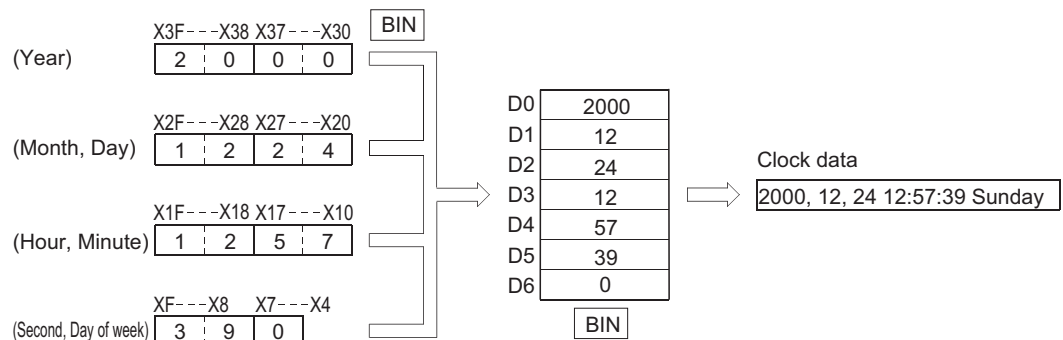
[Ladder Mode]



[List Mode]

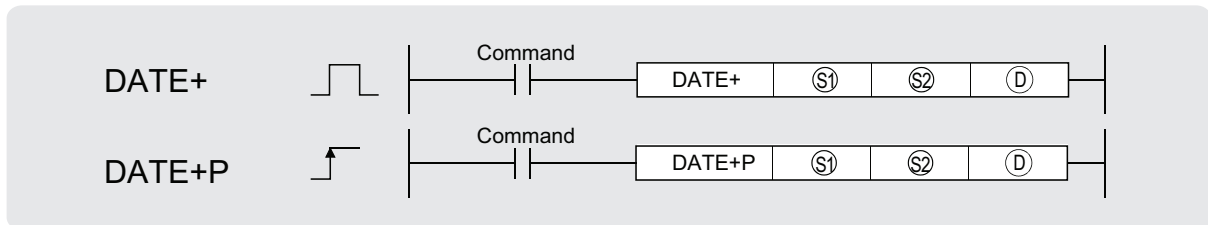
Step	Instruction	Device
0	LD	X40
1	BIN	K4X30 D0
4	BIN	K2X28 D1
7	BIN	K2X20 D2
10	BIN	K2X18 D3
13	BIN	K2X10 D4
16	BIN	K2X8 D5
19	BIN	K1X4 D6
22	DATEWRP	D0
24	END	

[Operation]



## 7.15.3 Clock data addition operation (DATE+(P))

Basic High performance Process Redundant Universal



Ⓢ1 : Head number of the devices where the clock data to be adjusted by addition is stored (BIN 16 bits)

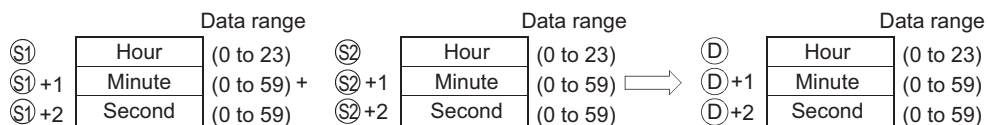
Ⓢ2 : Head number of the devices where the time data to be added for adjustment is stored (BIN 16 bits)

ⓓ : Head number of the devices where the result of addition of clock (time) data will be stored (BIN 16 bits)

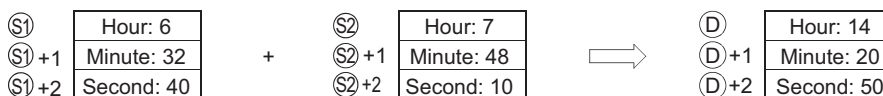
Setting Data	Internal Devices		R, ZR	J20		U2G0	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○					—		
Ⓢ2	—	○					—		
ⓓ	—	○					—		

### ★ Function

- (1) Adds the time data designated by Ⓢ2 to the clock data designated by Ⓢ1, and stores the result into the area starting from the device designated by ⓓ.

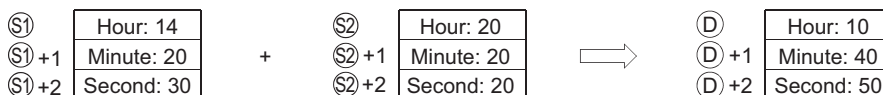


For example, adding the time 7:48:10 to 6:32:40 would result in the following operation:



- (2) If the results of the addition of time exceed 24 hours, 24 hours will be subtracted from the sum to make the final operation result.

For example, if the time 20:20:20 were added to 14:20:30, the result would not be 34:40:50, but would instead be 10:40:50.



#### Remark

See 7.15.2 for further information regarding the data that can be set for hours, minutes, and seconds.

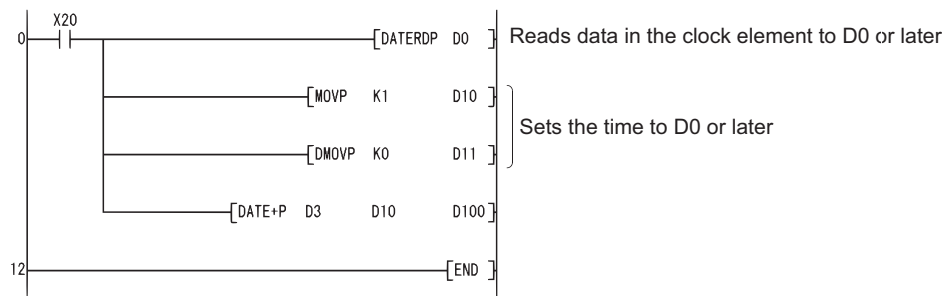
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data set by ① and ② is outside the setting range. (Error code: 4100)
  - The device specified by ① or ② or ③ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program adds 1 hour to the clock data read from the clock element, and stores the results in the area starting from D100 when X20 is ON.

[Ladder Mode]

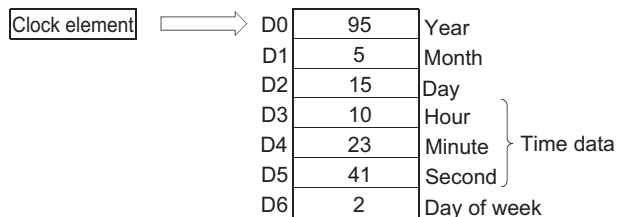


[List Mode]

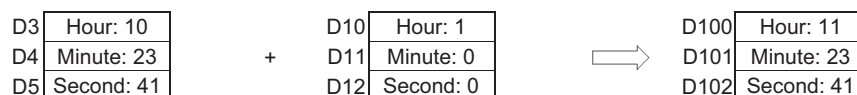
Step	Instruction	Device
0	LD	X20
1	DATERDP	D0
3	MOV P	K1 D10
5	DMOV P	K0 D11
8	DATE+P	D3 D10 D100
12	END	

[Operation]

- Time data read operation triggered by DATERDP instruction.

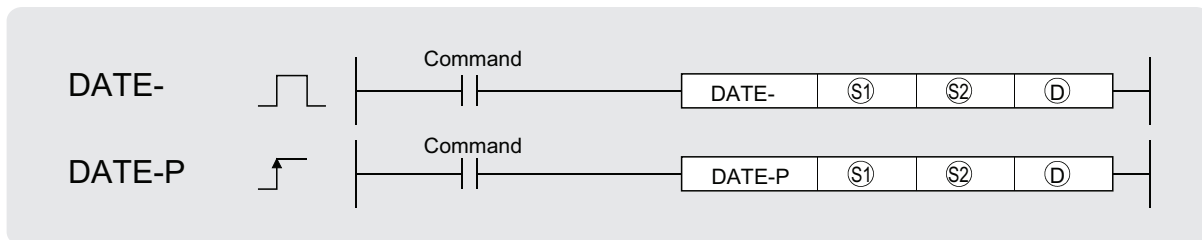


- Addition triggered by DATE+P instruction.



# 7.15.4 Clock data subtraction operation (DATE-(P))

Basic High performance Process Redundant Universal

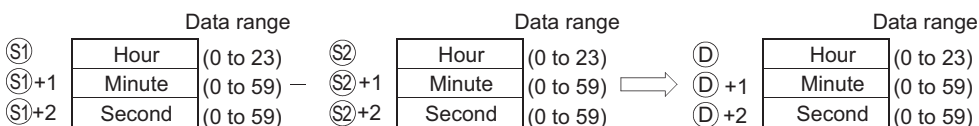


- Ⓢ1 : Head number of the devices where the clock time data to be adjusted by subtraction is stored (BIN 16 bits)
- Ⓢ2 : Head number of the devices where time data to be subtracted for adjustment is stored (BIN 16 bits)
- Ⓣ : Head number of the devices where the result of subtraction of clock (time) data will be stored (BIN 16 bits)

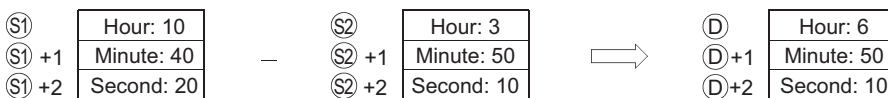
Setting Data	Internal Devices		R, ZR	I/O		U/I/O	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○					—		
Ⓢ2	—	○					—		
Ⓣ	—	○					—		

## ★ Function

- (1) Subtracts the time data designated by Ⓢ2 from the clock data designated by Ⓢ1, and stores the result into the area starting from the device designated by Ⓣ.

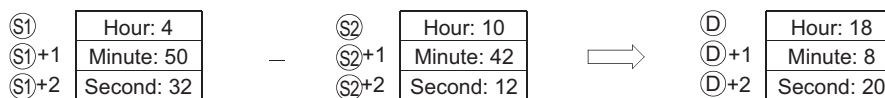


For example, if the clock time 3:50:10 were subtracted from the clock time 10:40:20, the operation would be performed as follows:



- (2) If the subtraction results in a negative number, 24 will be added to the result to make a final operation result.

For example, if the clock time 10:42:12 were subtracted from 4:50:32, the result would not be -6:8:20, but rather would be 18:8:20.



### Remark

See 7.15.2 for further information regarding the data that can be set for hours, minutes, and seconds.

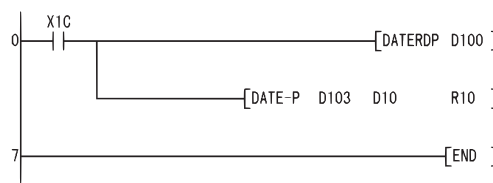
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data set by ⑤<sub>1</sub> and ⑤<sub>2</sub> is outside the setting range. (Error code: 4100)
  - The device specified by ⑤<sub>1</sub> or ⑤<sub>2</sub> or ④ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

- (1) The following program subtracts the time data stored in devices starting from D10 from the clock data read from the clock element when X1C is turned ON, and stores the result at devices starting from R10.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X1C
1	DATERDP	D100
3	DATE-P	D103 D10 R10
7	END	

[Operation]

- Time data read operation triggered by DATERDP instruction.

Clock device	Device	Value	Unit
	D100	95	Year
	D101	4	Month
	D102	20	Day
	D103	3	Hour
	D104	21	Minute
	D105	20	Second
	D106	1	Day of week

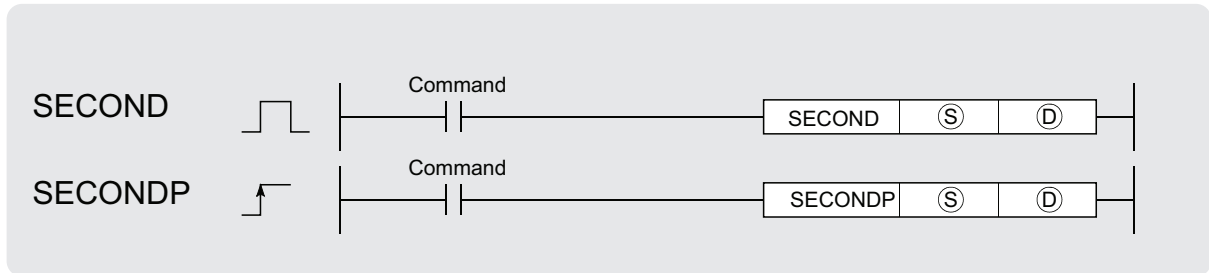
} Time data

- Subtraction as triggered by DATE-P instruction (when 10 hours, 40 minutes, and 10 seconds have been designated by D10 to D12).

D103	Hour: 3		D10	Hour: 10		R10	Hour: 16
D104	Minute: 21		D11	Minute: 40		R11	Minute: 41
D105	Second: 20		D12	Second: 10		R12	Second: 10
3:21:20 - 10:40:10				-8:41:10		16:41:10	
				↓			
				24 is added to this value			

# 7.15.5 Time data conversion (from Hour/Minute/Second to Second) (SECOND(P))

Basic High performance Process Redundant Universal



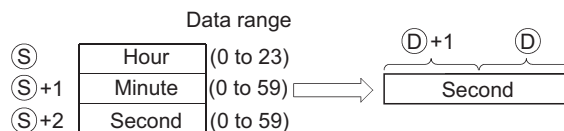
Ⓢ : Head number of the devices where the clock data before conversion is stored (BIN 16 bits)

Ⓣ : Head number of the devices where the clock data after conversion will be stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		—	
Ⓣ	○	○				○		—	

## ★ Function

- Converts the time data stored in the area starting from the device designated by Ⓢ to seconds and stores the conversion result into the device designated by Ⓣ.



For example, if the value were 4 hours, 29 minutes, and 31 seconds, the conversion would be made as follows:



## ! Operation Error

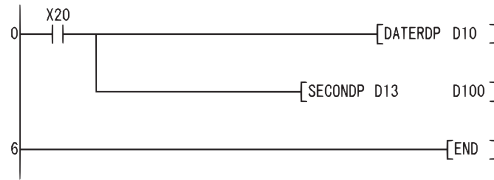
- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The data set by Ⓢ and is outside the setting range. (Error code: 4100)
  - The device specified by Ⓢ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)



# Program Example

- (1) The following program converts the clock time data read from the clock element into second when X20 is turned ON, and stores the result at D100 and D101.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X20
1	DATERDP	D10
3	SECONDP	D13 D100
6	END	

[Operation]

- Time data read operation triggered by DATERDP instruction.

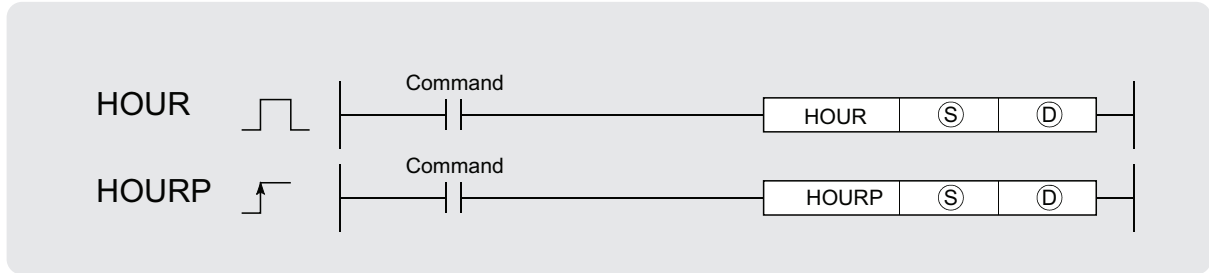
Clock device	→	D10	95	Year	} Time data
		D11	4	Month	
		D12	20	Day	
		D13	20	Hour	
		D14	21	Minute	
		D15	23	Second	
		D16	5	Day of week	

- Conversion to seconds as triggered by the SECONDP instruction.

D13	20	→ D101, D100	73283
D14	21		
D15	23		

## 7.15.6 Time data conversion (from Second to Hour/Minute/Second) (HOUR(P))

Basic High performance Process Redundant Universal



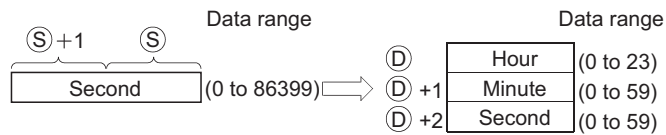
Ⓢ : Head number of the devices where clock data before conversion is stored (BIN 32 bits)

Ⓣ : Head number of the devices where the clock data after conversion will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JAG		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○	○				○		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- Converts the data in seconds stored in the device number designated by Ⓢ to an hour/minute/second format, and stores the conversion result into the area starting from the device designated by Ⓣ.



For example, if 45325 seconds were the value designated, the conversion operation would be conducted as follows:



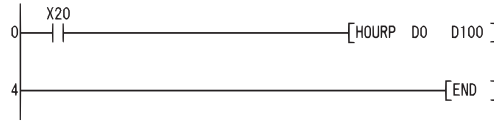
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The data set by Ⓢ and is outside the setting range. (Error code: 4100)
  - The device specified by Ⓣ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

# Program Example

- (1) The following program converts the seconds stored at D0 and D1 into an hour, minute, second format, and stores the result at devices starting from D100 when X20 is turned ON.

[Ladder Mode]

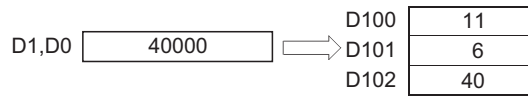


[List Mode]

Step	Instruction	Device
0	LD	X20
1	HOURP	D0 D100
4	END	

[Operation]

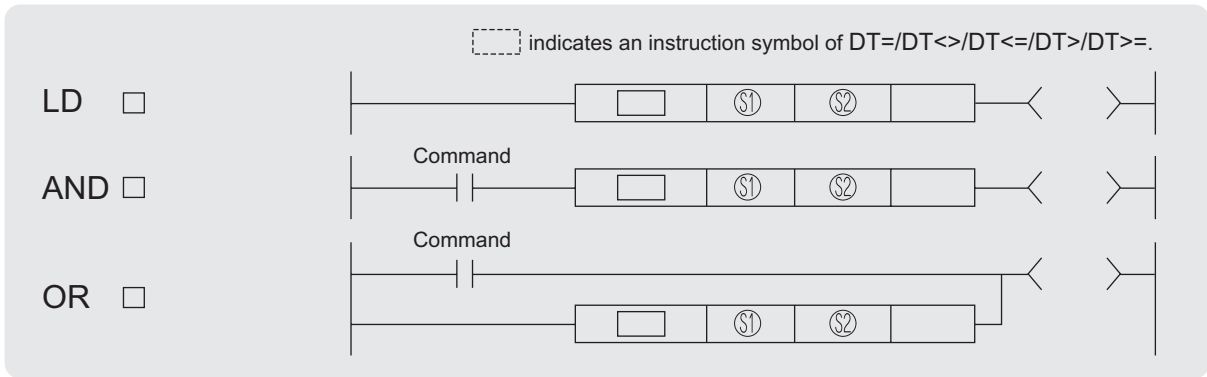
- Conversion to hour minute, and second format by the HOURP instruction (when the value 40000 seconds has been designated by D1 and D0).



# 7.15.7 Date comparison (DT=,DT<>,DT>,DT<=,DT<,DT>=)



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.  
QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ1 : Head number of the devices where the data to be compared are stored (BIN 16 bits)  
Ⓢ2 : Head number of the devices where the data to be compared are stored (BIN 16 bits)  
n : Value of the data to be compared or the number of the stored data to be compared (BIN 16 bits)

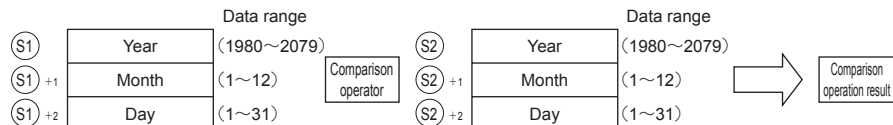
Setting Data	Internal Devices		R, ZR	Jm		UaGo	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		—	—
Ⓢ2	—	○				—		—	—
n	—	○				○		○	—

## ★ Function

(1) This instruction compares the date data specified by Ⓢ1 with those specified by Ⓢ2, or the date data specified by Ⓢ1 with current date data. Setting n can determine the data to be compared.

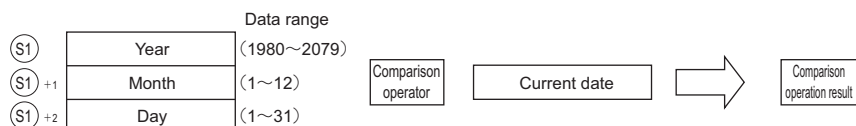
(a) Comparison of given date data

- This instruction treats the date data specified by Ⓢ1 and Ⓢ2 as a normally open contact, and then compares the data in accordance with the value of n.



(b) Comparison of current date data

- This instruction treats the date data specified by Ⓢ1 and the current date data as a normally open contact, and then compares the data in accordance with the value of n.



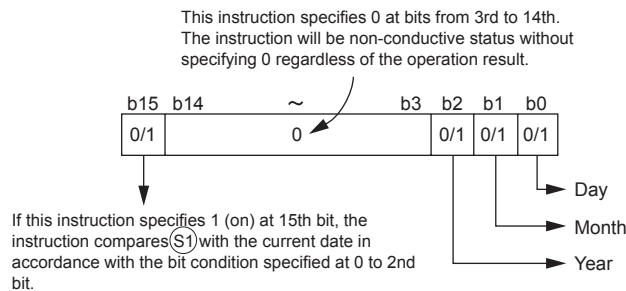
**POINT**

When either  $\textcircled{S1}$  or  $\textcircled{S2}$  corresponds to any of the following in comparing given or current date data with given date data, the operation error (error code: 4101) or a malfunction may occurs.

- The range of the devices to be used for the index modification is specified over the range of the device specified by  $\textcircled{S1}$  or  $\textcircled{S2}$ .
- File registers are specified by  $\textcircled{S1}$  or  $\textcircled{S2}$  without a register set.

- (2) This instruction sets BIN values for each item.
- (3) This instruction sets the year of four digits selected from 1980 to 2079 with the BIN value specified by  $\textcircled{S1}$  or  $\textcircled{S2}$ .
- (4) This instruction sets the month selected from 1 to 12 (January to December) with the BIN value specified by  $\textcircled{S1} + 1$  or  $\textcircled{S2} + 1$ .
- (5) This instruction sets the day selected from 1 to 31 (1st to 31st) for with the BIN value specified by  $\textcircled{S1} + 2$  or  $\textcircled{S2} + 2$ .
- (6) This instruction specifies the following values at n so that the data to be compared can be specified.

The bit configuration specified at n is as follows.



- (a) Date data to be compared (from 0 to 2nd bit)
  - 0: Does not compare specified date data (year/month/day).
  - 1: Compares specified date data (year/month/day).
- (b) Operation data to be compared (15th bit)
  - 0: Compares the date data specified by  $\textcircled{S1}$  with the date data specified by  $\textcircled{S2}$ .
  - 1: Compares the date data specified by  $\textcircled{S1}$  with the current date data.
  - Ignores the date data specified by  $\textcircled{S2}$ .

(c) The following table shows processing details of bits to be compared.

n value for comparison of specified date data with given date data	n value for comparison of specified date data with current date data	Date to be compared	Processing details
0001H	8001H	Day	Comparison of days ( $S_1+2$ )
0002H	8002H	Month	Comparison of months ( $S_1+1$ )
0003H	8003H	Month, day	Comparison of months ( $S_1+1$ ) and days ( $S_1+2$ )
0004H	8004H	Year	Comparison of years ( $S_1$ )
0005H	8005H	Year, day	Comparison of years ( $S_1$ ) and days ( $S_1+2$ )
0006H	8006H	Year, month	Comparison of years ( $S_1$ ) and months ( $S_1+1$ )
0007H	8007H	Year, month, day	Comparison of years ( $S_1$ ), months ( $S_1+1$ ), and days ( $S_1+2$ )
Other than 0001H to 0007H, 8001H to 8007H		No objects	No comparison of years ( $S_1$ ), months ( $S_1+1$ ), and days ( $S_1+2$ ) (Non-conductive)

(7) If the data stored in the devices to be compared are not recognized as date data, SM709 will be turned on after the instruction execution and no-conductive status will be made. Even if they are not recognized as date data but the range of the devices is within the setting range, SM709 will not be turned on.

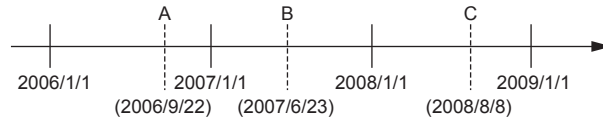
Moreover, if the range of devices specified by  $S_1$  to  $S_1+2$  or  $S_2$  to  $S_2+2$  exceeds the range of specified devices, SM709 will be turned on after the instruction execution and no-conductive status will be made.

Once SM709 is turned on, on-status will be retained till when the CPU modules are reset or powered off. Therefore, turn off SM709 if necessary.

(8) The following table shows the comparison operation results for each instruction.

Instruction symbols in $\square$	Condition	Comparison operation result	Instruction symbols in $\square$	Condition	Comparison operation result
DT=	$S_1 = S_2$	Conductive status	DT=	$S_1 \neq S_2$	No-conductive status
DT<>	$S_1 \neq S_2$		DT<>	$S_2 = S_1$	
DT>	$S_1 > S_2$		DT>	$S_1 \leq S_2$	
DT<=	$S_1 \leq S_2$		DT<=	$S_1 > S_2$	
DT<	$S_1 < S_2$		DT<	$S_1 \geq S_2$	
DT>=	$S_1 \geq S_2$		DT>=	$S_1 < S_2$	

(a) The following figure shows the comparison example of dates.



The following table shows the conductive states resulting from performing the comparison operation of the dates A, B, and C shown above.

Even if the objects to be compared are under the same condition, the comparison operation results vary depending on the objects selected.

Comparison objects	Comparison condition		
	A<B	B<C	A<C
Day	○	×	×
Month	×	○	×
Month, day	×	○	×
Year	○	○	○
Month, day	○	○	○
Year, month	○	○	○
Year, month, day	○	○	○
No objects	×	×	×

○: Conductive ×: No-conductive

(b) Even if the dates to be compared do not exist practically, this instruction executes the comparison operation for the objects with the settable dates in accordance with the following condition.

- Date A: 2006/02/30 (This date is settable, though it does not exist.)
- Date B: 2007/03/29
- Date C: 2008/02/31 (This date is settable, though it does not exist.)

Comparison objects	Comparison condition		
	A<B	B<C	A<C
Day	×	×	○
Month	×	×	×
Month, day	○	×	○
Year	○	○	○
Month, day	○	○	○
Year, month	○	○	○
Year, month, day	○	○	○
No objects	×	×	×

○: Conductive ×: No-conductive



## Operation Error

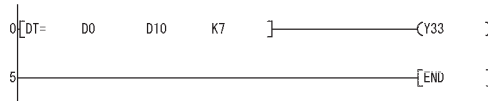
- (1) Any operation errors do not occur in DT=,DT<>,DT>,DT<=,DT<,DT>= instruction.



## Program Example

- (1) The following program compares the data stored in D0 with the data (year, month, and day) stored in D10, and makes Y33 be conductive status when the data stored in D0 meet the data stored in D10.

[Ladder Mode]

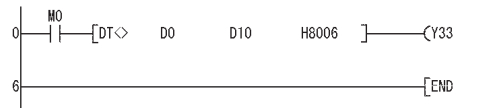


[List Mode]

Step	Instruction	Device
0	LDDT=	D0 D10 K7
4	OUT	Y33
5	END	

- (2) The following program compares the data stored in D0 with the current date data (year and month), and makes Y33 be conductive status when the data stored in D0 do not meet the current date data, when M0 is turned on.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	ANDDT<>	D0 D10 H8006
5	OUT	Y33
6	END	

- (3) The following program compares the data stored in D0 with the data (year and day) stored in D10, and makes Y33 be conductive status when the data value stored in D10 is smaller than the data value stored in D0, when M0 is turned on.

[Ladder Mode]

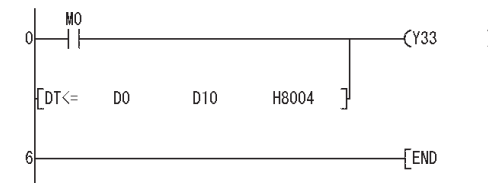


[List Mode]

Step	Instruction	Device
0	LD	M0
1	ANDDT>	D0 D10 K5
5	OUT	Y33
6	END	

- (4) The following program compares the data stored in D0 with the current date data (year), and makes Y33 be conductive status when the value of the current date data is the data value stored in D0 or larger.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	ORDT<=	D0 D10 H8004
5	OUT	Y33
6	END	

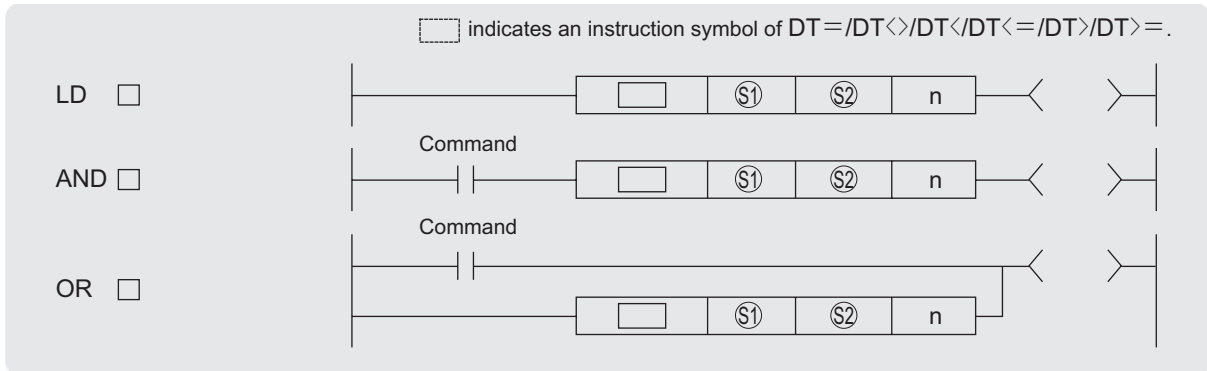


## 7.15.8 Clock comparison (TM=,TM<>,TM>,TM<=,TM<,TM>=)



QnU(D)(H)CPU: The serial number (first five digits) is "10102" or later.

QnUDE(H)CPU: The serial number (first five digits) is "10102" or later.



Ⓢ1 : Head number of the devices where the data to be compared are stored (BIN 16 bits)

Ⓢ2 : Head number of the devices where the data to be compared are stored (BIN 16 bits)

n : Value of the data to be compared or the number of the stored data to be compared (BIN 16 bits)

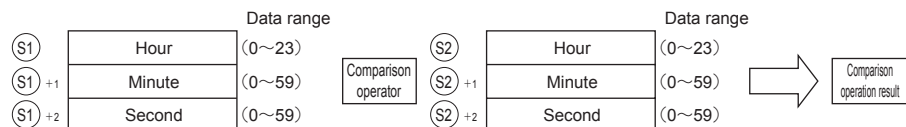
Setting Data	Internal Devices		R, ZR	J:G		U:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		—	—
Ⓢ2	—	○				—		—	—
n	—	○				○		○	—

### ★ Function

- (1) This instruction compares the clock data specified by Ⓢ1 with those specified by Ⓢ1, or the clock data specified by Ⓢ1 with the current time data. Setting n determines the data to be compared.

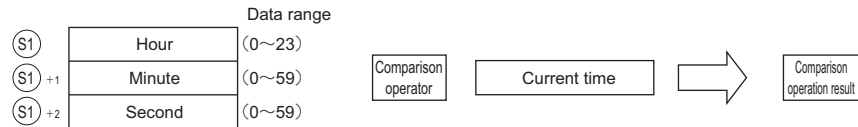
(a) Comparison of given clock data

- This instruction treats the clock data specified by Ⓢ1 and the clock data specified by Ⓢ1 as a normally open contact, and compares the data in accordance with the value of n.



## (b) Comparison of current time data

- This instruction treats the clock data specified by  $\text{S1}$  and the current time data as a normally open contact, and compares the data in accordance with the value of n.
- This instruction treats the clock data specified by  $\text{S1}$  as dummy data and ignores the data.

**POINT**

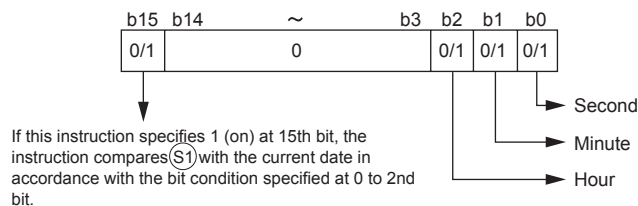
When either  $\text{S1}$  or  $\text{S1}$  corresponds to any of the following conditions in comparing given or current time data with specified clock data, the operation error (error code: 4101) or a malfunction may occur.

- The range of the devices to be used for the index modification is specified over the range of the device specified by  $\text{S1}$  or  $\text{S1}$ .
- File registers are specified by  $\text{S1}$  or  $\text{S1}$  without a register set.

- (2) This instructions set BIN values for each item.
- (3) This instructions sets the time selected from 0 to 23 (midnight to 23 o'clock) with the BIN value specified by  $\text{S1}$  or  $\text{S1}$ . (Uses the 24-hour clock.)
- (4) This instructions sets the minute selected from 0 to 59 (0 to 59 minutes) with BIN value specified by  $\text{S1}+1$  or  $\text{S1}+1$ .
- (5) This instructions sets the second selected from 0 to 59 (0 to 59 seconds) with BIN value specified by  $\text{S1}+2$  or  $\text{S1}+2$ .
- (6) This instructions specifies the following values at n so that the data to be compared can be specified.

The bit configuration specified at n is as follows.

This instruction specifies 0 at bits from 3rd to 14th.  
The instruction will be non-conductive status without specifying 0 regardless of the operation result.



## (a) Clock data to be compared (from 0 to 2nd bit)

- 0: Does not compare specified clock data (hour/minute/second).
- 1: Compares specified clock data (hour/minute/second).

## (b) Operation data to be compared (15th bit)

- 0: Compares the clock data specified by  $\text{S1}$  with the clock data specified by  $\text{S1}$ .
- 1: Compares the clock data specified by  $\text{S1}$  with the current time data.  
Ignores the clock data specified by  $\text{S1}$ .



(c) The following table shows processing details of bits to be compared.

n value for comparison of specified clock data with given clock data	n value for comparison of specified clock data with current time data	Time to be compared	Processing details
0001H	8001H	Second	Comparison of seconds (S <sub>1</sub> +2)
0002H	8002H	Minute	Comparison of minutes (S <sub>1</sub> +1)
0003H	8003H	Minute, second	Comparison of minutes (S <sub>1</sub> +1) and seconds days (S <sub>1</sub> +2)
0004H	8004H	Hour	Comparison of hours (S <sub>1</sub> )
0005H	8005H	Hour, second	Comparison of hours (S <sub>1</sub> ) and seconds (S <sub>1</sub> +2)
0006H	8006H	Hour, minute	Comparison of hours (S <sub>1</sub> ) and minutes (S <sub>1</sub> +1)
0007H	8007H	Hour, minute, second	Comparison of hours (S <sub>1</sub> ), minutes (S <sub>1</sub> +1), and seconds (S <sub>1</sub> +2)
Other than 0001H to 0007H, 8001H to 8007H		No objects	No comparison of hours (S <sub>1</sub> ), minutes (S <sub>1</sub> +1), and seconds (S <sub>1</sub> +2) (Non-conductive)

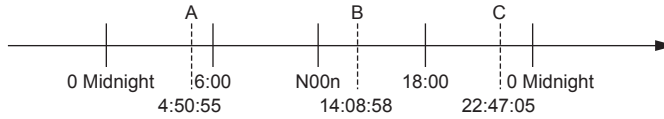
- (7) If the data stored in the devices to be compared are not recognized as date data, SM709 will be turned on after the instruction execution and no-conductive status will be made. Once SM709 is turned on, on-status will be retained till when the CPU modules are reset or powered off. Therefore, turn off SM709 if necessary.

Moreover, if the range of devices specified by S<sub>1</sub> to S<sub>1</sub>+2 or S<sub>1</sub> to S<sub>1</sub>+2 exceeds the range of specified devices, SM709 will be turned on and no-conductive status will be made.

- (8) The following table shows the comparison operation results for each instruction.

Instruction symbols in 	Condition	Comparison operation result	Instruction symbols in 	Condition	Comparison operation result
TM=	S <sub>1</sub> = S <sub>2</sub>	Conductive status	TM=	S <sub>1</sub> ≠ S <sub>2</sub>	No-conductive status
TM<>	S <sub>1</sub> ≠ S <sub>2</sub>		TM<>	S <sub>2</sub> = S <sub>1</sub>	
TM>	S <sub>1</sub> > S <sub>2</sub>		TM>	S <sub>1</sub> ≦ S <sub>2</sub>	
TM<=	S <sub>1</sub> ≧ S <sub>2</sub>		TM<=	S <sub>1</sub> > S <sub>2</sub>	
TM<	S <sub>1</sub> < S <sub>2</sub>		TM<	S <sub>1</sub> ≧ S <sub>2</sub>	
TM>=	S <sub>1</sub> ≧ S <sub>2</sub>		TM>=	S <sub>1</sub> < S <sub>2</sub>	

(a) The following figure shows the comparison example of time.



The following table shows the conductive states resulting from performing the comparison operation of the dates A, B, and C shown above.

Even if the objects to be compared are under the same condition, the comparison operation results vary depending on the objects selected.

Comparison objects	Comparison condition		
	A<B	B<C	A<C
Second	○	×	×
Month	×	○	×
Month, day	×	○	×
Hour	○	○	○
Hour, second	○	○	○
Hour, minute	○	○	○
Hour, minute, second	○	○	○
No objects	×	×	×

○: Conductive × : No-conductive



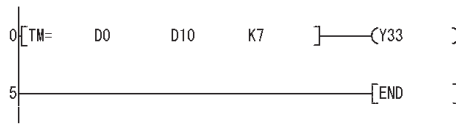
## Operation Error

(1) Any operation errors do not occur in TM=, TM<>, TM>, TM<=, TM<, TM>=instruction.

## Program Example

- (1) The following program compares the data stored in D0 with the data (hour, minute, and second) stored in D10, and makes Y33 be conductive status when the data stored in D0 meet the data stored in D10.

[Ladder Mode]

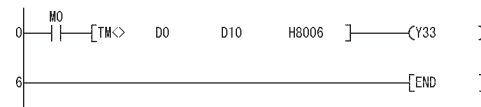


[List Mode]

Step	Instruction	Device
0	LDTM=	D0 D10 K7
4	OUT	Y33
5	END	

- (2) The following program compares the data stored in D0 with the current time data (hour and minute), and makes Y33 be conductive status when the data stored in D0 do not meet the current date data, when M0 is turned on.

[Ladder Mode]

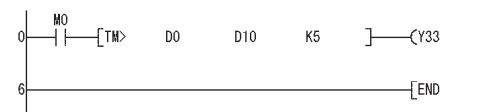


[List Mode]

Step	Instruction	Device
0	LD	M0
1	ANDTM<>	D0 D10 H8006
5	OUT	Y33
6	END	

- (3) The following program compares the data stored in D0 with the data (hour and second) stored in D10, and makes Y33 be conductive status when the data value stored in D10 is smaller than the data value stored in D0, when M0 is turned on.

[Ladder Mode]

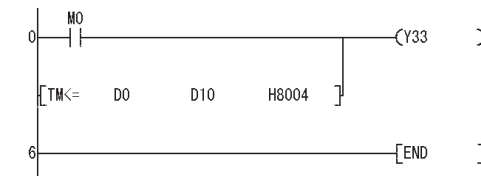


[List Mode]

Step	Instruction	Device
0	LD	M0
1	ANDTM>	D0 D10 K5
5	OUT	Y33
6	END	

- (4) The following program compares the data stored in D0 with the current time data (hour), and makes Y33 be conductive status when the value of the current time data is the data value stored in D0 or larger.

[Ladder Mode]



[List Mode]

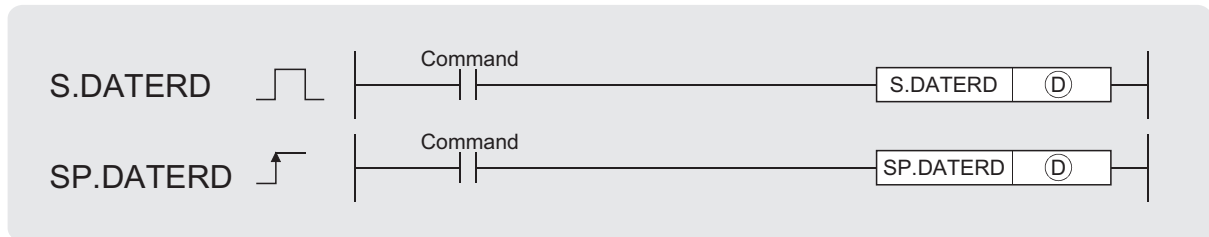
Step	Instruction	Device
0	LD	M0
1	ORTM<=	D0 D10 H8004
5	OUT	Y33
6	END	

## 7.16 Expansion Clock Instructions

### 7.16.1 Reading expansion clock data (S(P).DATERD)



The first 5 digits of the serial No. are "07032" or higher.



Ⓧ: Head number of the devices where the read clock data will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ	—	○					—		

### ★ Function

- (1) Reads "year, month, day, hour, minute, second, day of the week, and millisecond" from the clock element of the CPU module, and stores it as BIN value into the device specified by Ⓧ or later device.

Ⓧ	Element	Range
Ⓧ	Year	(1980 to 2079)
Ⓧ +1	Month	(1 to 12)
Ⓧ +2	Day	(1 to 31)
Ⓧ +3	Hour (24-hour clock)	(0 to 23)
Ⓧ +4	Minute	(0 to 59)
Ⓧ +5	Second	(0 to 59)
Ⓧ +6	Day of week	(0 to 6)
Ⓧ +7	Millisecond	(0 to 999)

- (2) The "year" at Ⓧ is stored as 4-digit year indication.
- (3) The "day of the week" at Ⓧ +6 is stored as 0 to 6 to represent the days Sunday to Saturday.

Day of week	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Stored data	0	1	2	3	4	5	6

- (4) Compensation is made automatically for leap years.

### ! Operation Error

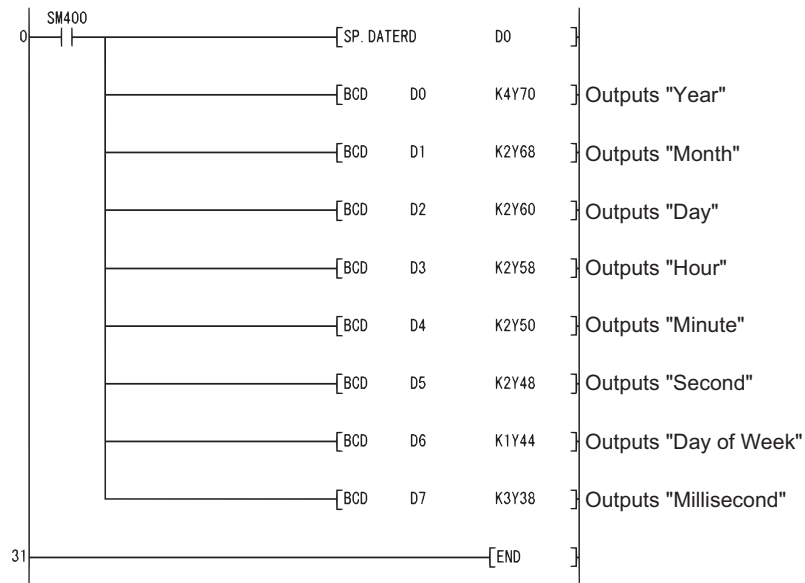
- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device specified by Ⓧ exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

## Program Example

(1) The following program outputs the following clock data as BCD values:

Year ..... Y70 to Y7F  
 Month ..... Y68 to Y6F  
 Day ..... Y60 to Y67  
 Hour ..... Y58 to Y5F  
 Minute ..... Y50 to Y57  
 Second ..... Y48 to Y4F  
 Week ..... Y44 to Y47  
 Millisecond ..... Y38 to Y43

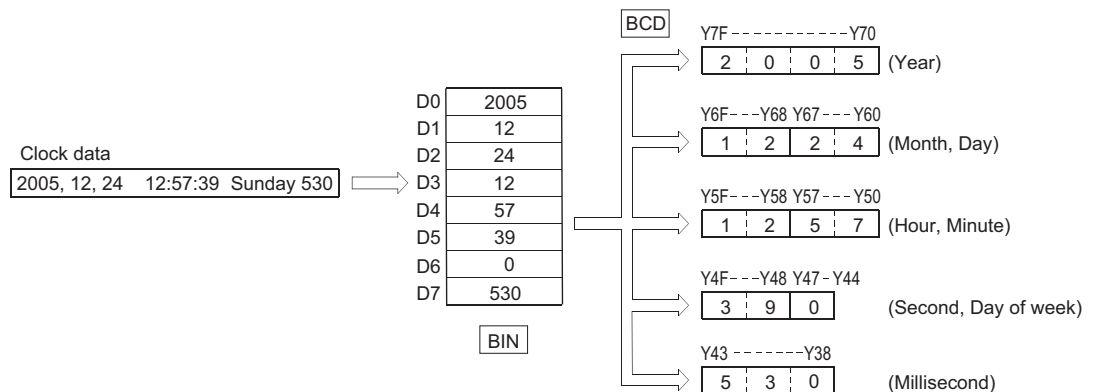
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM400
1	SP.DATERD	D0
7	BCD	K4Y70
10	BCD	K2Y68
13	BCD	K2Y60
16	BCD	K2Y58
19	BCD	K2Y50
22	BCD	K2Y48
25	BCD	K1Y44
28	BCD	K3Y38
31	END	

[Operation]



 **Caution**

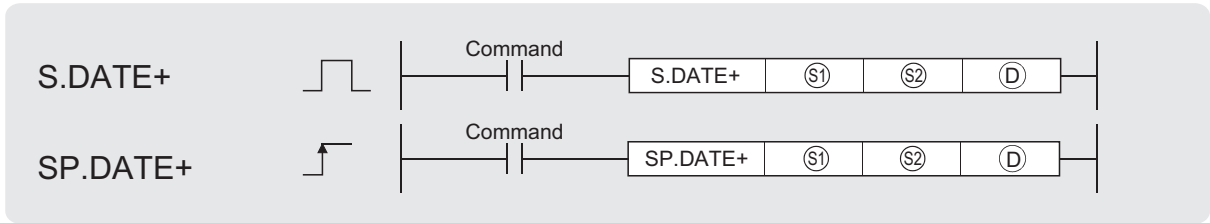
- (1) This instruction reads clock data and stores those to a specified device even if a wrong clock data is set to the CPU module. (example: Feb. 30th)  
When setting clock data with the DATEWR instruction or GX Developer, make sure to set a correct data.
- (2) Time error of reading a clock data of millisecond is a maximum of 2ms. (Difference between the data memorized by clock element inside of the CPU module and the data read by this function.)
- (3) Specifying digit for the bit device can be used only when the following conditions (a) and (b) are met.
  - (a) Digit specification: K4
  - (b) Head of device: multiple of 16When the above conditions (a) and (b) are not met, INSTRCT CODE ERR. (error code: 4004) will occur.



# 7.16.2 Expansion clock data addition operation (S(P).DATE+)



The first 5 digits of the serial No. are "07032" or higher.



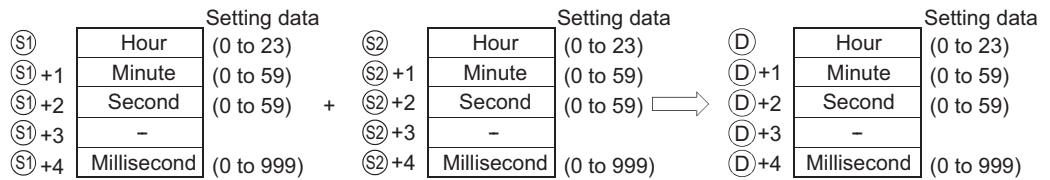
- Ⓢ1 : Head number of the devices where the clock data to be adjusted by addition is stored (BIN 16 bits)
- Ⓢ2 : Head number of the devices where the time data to be added for adjustment is stored (BIN 16 bits)
- ⓈD : Head number of the devices where the result of addition of clock (time) data will be stored (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J <small>0</small> : <small>0</small>		U <small>0</small> : <small>0</small> : <small>0</small>	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○					—		
Ⓢ2	—	○					—		
ⓈD	—	○					—		

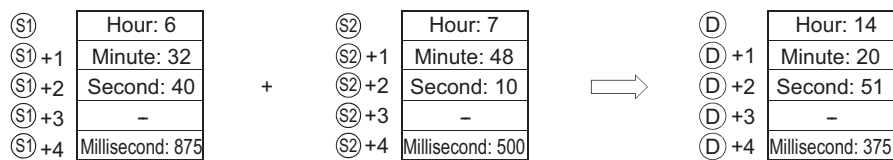
7

## ★ Function

- (1) Adds the time data designated by Ⓢ2 to the clock data designated by Ⓢ1, and stores the result into the area starting from the device designated by ⓈD.



For example, adding the time 7:48:10:500 to 6:32:40:875 would result in the following operation:



7.16 Expansion Clock Instructions  
7.16.2 Expansion clock data addition operation (S(P).DATE+)

- (2) If the results of the addition of time exceed 24 hours, 24 hours will be subtracted from the sum to make the final operation result.

For example, when the time 20:20:20:500 is added to 14:20:30:875, the result is not 34:40:51:375, but 10:40:51:375.

Ⓢ1	Hour: 14	+	Ⓢ2	Hour: 20	→	Ⓓ	Hour: 10
Ⓢ1+1	Minute: 20		Ⓢ2+1	Minute: 20		Ⓓ+1	Minute: 40
Ⓢ1+2	Second: 30		Ⓢ2+2	Second: 20		Ⓓ+2	Second: 51
Ⓢ1+3	-		Ⓢ2+3	-		Ⓓ+3	-
Ⓢ1+4	Millisecond: 875		Ⓢ2+4	Millisecond: 500		Ⓓ+4	Millisecond: 375

## POINT

Devices, Ⓢ1+3, Ⓢ2+3, and Ⓓ+3 are not used for operation.

A clock data read by the S(P).DATERD instruction can be directly added.

Ⓓ	Hour	
Ⓓ+1	Minute	
Ⓓ+2	Second	
Ⓓ+3	Day of week	←■■■■
Ⓓ+4	Millisecond	

When the clock data is read by the S(P).DATERD instruction, day of week is inserted between "second" and "millisecond".

If the S(P).DATE+ instruction is used to read the clock data, the data can be directly used for addition since it does not perform the calculation for the day of a week.

## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data set by Ⓢ1 and Ⓢ2 is outside the range. (See Function (1).) (Error code: 4100)
  - The device specified by Ⓢ1, Ⓢ2 or Ⓓ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

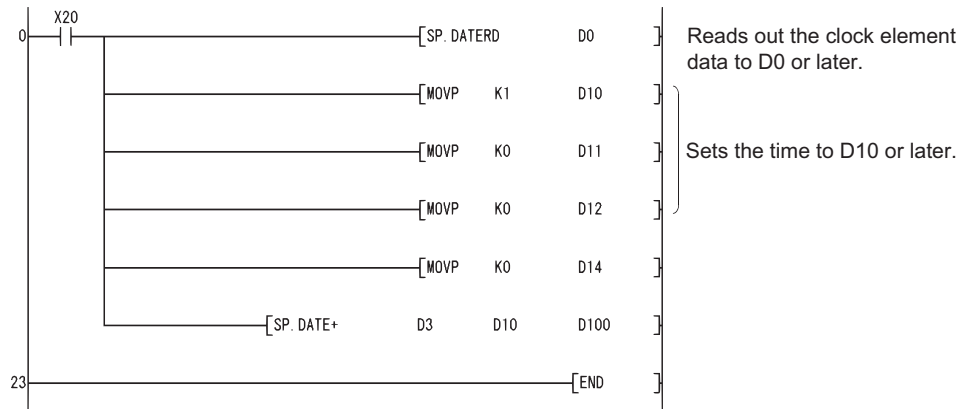
## ⚠ Caution

- (1) Specifying digit for the bit device can be used only when the following conditions (a) and (b) are met.
- (a) Digit specification: K4
  - (b) Head of device: multiple of 16
- When the above conditions (a) and (b) are not met, INSTRCT CODE ERR. (error code:4004) will occur.

## Program Example

- (1) The following program adds 1 hour to the clock data read from the clock element, and stores the results into the area starting from D100 when X20 is turned ON.

[Ladder Mode]

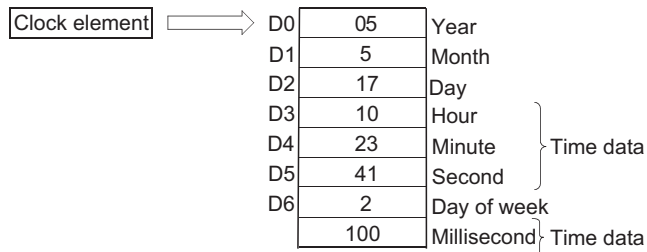


[List Mode]

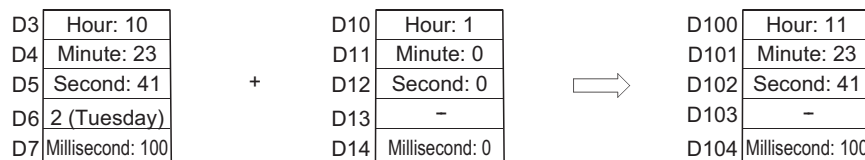
Step	Instruction	Device
0	LD	X20
1	SP.DATERD	D0
7	MOV P	K1 D10
9	MOV P	K0 D11
11	MOV P	K0 D12
13	MOV P	K0 D14
15	SP.DATE+	D3 D10 D100
23	END	

[Operation]

- Time data read operation by the SP.DATERD instruction



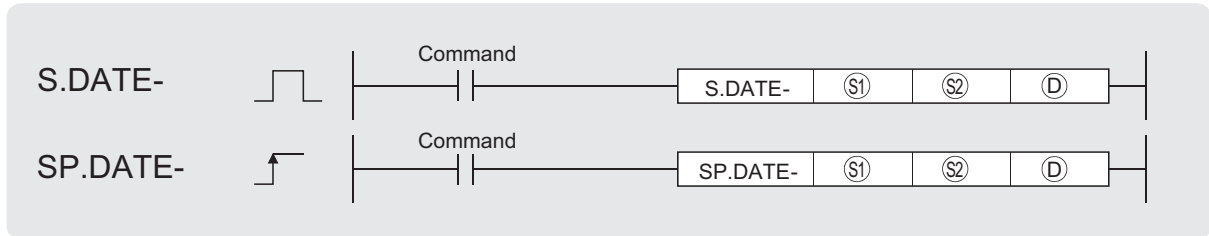
- Addition by the SP.DATE+ instruction



# 7.16.3 Expansion clock data subtraction operation (S(P).DATE-)



The first 5 digits of the serial No. are "07032" or higher.

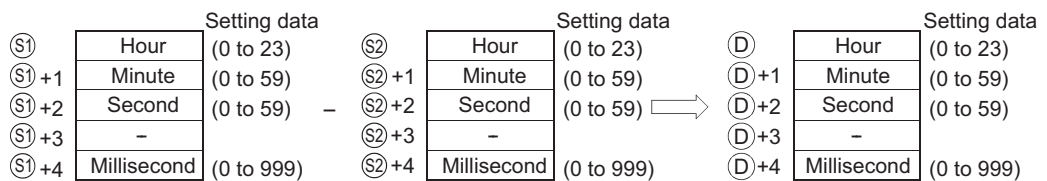


- Ⓢ1 : Head number of the devices where the clock time data to be adjusted by subtraction is stored (BIN 16 bits)
- Ⓢ2 : Head number of the devices where time data to be subtracted for adjustment is stored (BIN 16 bits)
- Ⓧ : Head number of the devices where the result of subtraction of clock (time) data will be stored (BIN 16 bits)

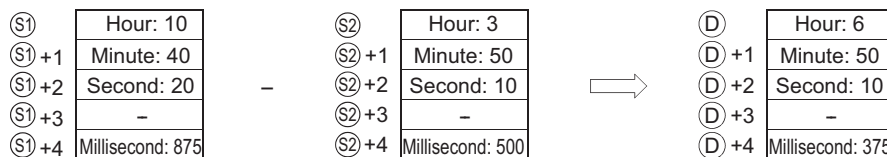
Setting Data	Internal Devices		R, ZR	Jm		UAGo	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○					—		
Ⓢ2	—	○					—		
Ⓧ	—	○					—		

## ★ Function

- (1) Subtracts the time data designated by Ⓢ2 from the clock data designated by Ⓢ1, and stores the result into the area starting from the device designated by Ⓧ.



For example, when the clock time 3:50:10:500 is subtracted from the clock time 10:40:20:875, the operation is performed as follows:



- (2) If the subtraction results in a negative number, 24 will be added to the result to make a final operation result.

For example, when the clock time 10:42:12:500 is subtracted from 4:50:32:875, the result is not 6:8:20:375, but 18:8:20:375.

Ⓢ1	Hour: 4	-	Ⓢ2	Hour: 10	→	Ⓣ	Hour: 18
Ⓢ1+1	Minute: 50		Ⓢ2+1	Minute: 42		Ⓣ+1	Minute: 8
Ⓢ1+2	Second: 32		Ⓢ2+2	Second: 12		Ⓣ+2	Second: 20
Ⓢ1+3	-		Ⓢ2+3	-		Ⓣ+3	-
Ⓢ1+4	Millisecond: 875		Ⓢ2+4	Millisecond: 500		Ⓣ+4	Millisecond: 375

### POINT

Devices, Ⓢ1+3, Ⓢ2+3, and Ⓣ+3 are not used for operation.

A clock data read by S(P).DATERD instruction can be directly subtracted.

Ⓣ	Hour	
Ⓣ+1	Minute	
Ⓣ+2	Second	
Ⓣ+3	Day of week	←■■■■
Ⓣ+4	Millisecond	

When the clock data is read by the S(P).DATERD instruction, day of week is inserted between "second" and "millisecond".

If the S(P).DATE- instruction is used to read the clock data, the data can be directly used for subtraction since it does not perform the calculation for the day of the week.

### Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The data set by Ⓢ1 and Ⓢ2 is outside the range. (See Function (1).) (Error code: 4100)
  - The device specified by Ⓢ1, Ⓢ2 or Ⓣ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

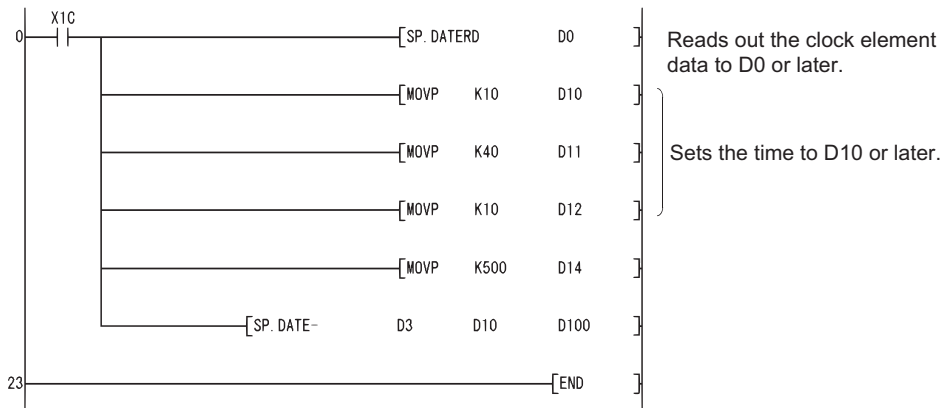
### Caution

- (1) Specifying digit for the bit device can be used only when the following conditions (a) and (b) are met.
- (a) Digit specification: K4
- (b) Head of device: multiple of 16
- When the above conditions (a) and (b) are not met, INSTRCT CODE ERR. (error code:4004) will occur.

# Program Example

- (1) The following program subtracts the time data stored in the area starting from D10 from the clock data read from the clock element when X1C is turned ON, and stores the result into the area starting from D100.

[Ladder Mode]

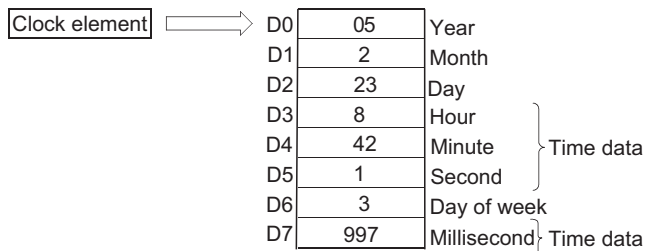


[List Mode]

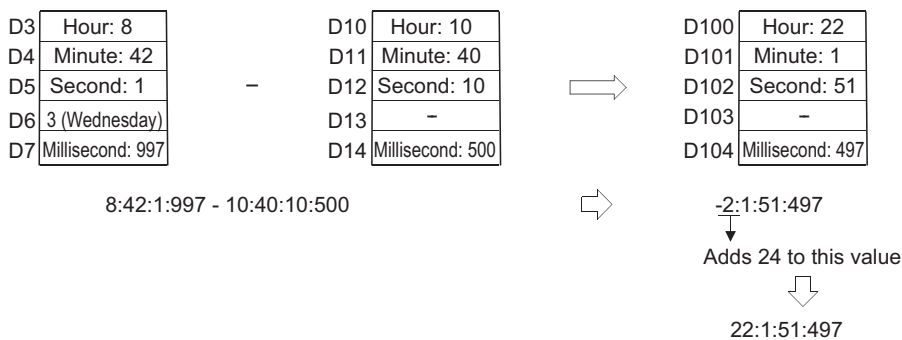
Step	Instruction	Device
0	LD	X1C
1	SP.DATERD	D0
7	MOV P	K10 D10
9	MOV P	K40 D11
11	MOV P	K10 D12
13	MOV P	K500 D14
15	SP.DATE-	D3 D10 D100
23	END	

[Operation]

- Time data read operation by the SP.DATERD instruction



- Subtraction by the SP.DATE- instruction



## 7.17 Program control instructions

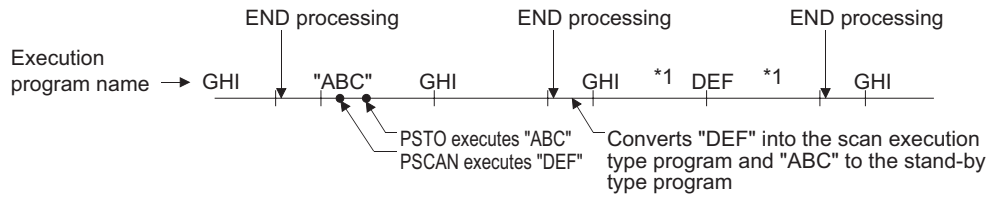
- (1) Processing when the execution type is converted with the program control instruction is as follows.

Execution type before change	Executed Instruction			
	PSCAN	PSTOP	POFF	PLOW
Scan execution type	No change-remains scan type execution.	Becomes stand-by type.	Output turned OFF in next scan.	Becomes low speed execution type.
Initial execution type	Becomes scan execution type.		Becomes stand-by type from the next scan after that.	
Stand-by type		No change-remains stand-by type	Ignored	
Low speed execution type	Low speed execution type execution is stopped, becomes scan execution type from the next scan. (Execution from step 0)	Low speed execution type execution is stopped, becomes stand-by type from next scan.	Low speed execution type execution is stopped, and output is turned OFF in the next scan. Becomes stand-by type from the next scan after that.	No change -remains low speed execution type.
Fixed scan execution type	Becomes scan execution type.	Becomes stand-by type.	Output turned OFF in next scan. Becomes stand-by type from the next scan after that.	Becomes low speed execution type.

### POINT

Once the fixed scan execution type program is changed to another execution type, it cannot be returned to the fixed scan execution type.

- (2) As program execution type conversions by PSCAN and PSTOP instructions occur at the END processing, such conversions are impossible during program execution. When different execution types have been set for the same program in the same scan, the execution type will be that specified by the execution switching command that was executed last.



\*1: The order of "GHI" and "DEF" program execution is determined by the program settings parameters. Switching from the fixed scan execution type program to the execution type program is performed in the following timing.

- (a) For the Universal model QCPU
    - The execution type is changed when the execution of the fixed scan execution type is stopped at the END processing after the program control instruction execution.
  - (b) For the CPU modules other than the Universal model QCPU
    - The execution of the fixed scan execution type is stopped at the execution of the program control instruction, and the execution type is changed at the END processing.
- (3) When the POFF instruction is executed, the output is turned OFF at the next scan, and the execution type will be the stand-by type at the second next scan and later. If executed prior to the output OFF processing, the program control instruction is ignored.



# 7.17.1 Program standby instruction (PSTOP(P))

Basic
High performance
Process
Redundant
Universal



Ⓢ : Character string for the name of the program file to be set in the stand-by status or head number of the devices where the character string data is stored (character string)

Setting Data	Internal Devices		R, ZR	JOG		U/G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○						○	—

## ★ Function

- (1) Places the file name program stored in the device designated by Ⓢ in the stand-by status.
- (2) Only the programs stored in the drive No. 0 (program memory/internal RAM) can be set as the stand-by type.
- (3) The specified program is placed in the stand-by status when END processing is performed.
- (4) This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- (5) It is not necessary to designate the extension (.QPG) with the file name.  
(Only .QPG files will be acted on.)

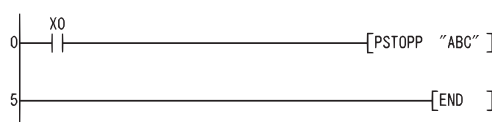
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The program with the file name specified by Ⓢ does not exist. (Error code: 2410)
  - The program type of the file name specified by Ⓢ is the SFC program. (Error code: 2412)
  - The file name storage destination device of Ⓢ exceeds the range of the corresponding device. (Error code: 4101)

## 📄 Program Example

- (1) The following program places the program with the file name ABC in the stand-by status when X0 goes ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	PSTOPP	"ABC"
5	END	

## 7.17.2 Program output OFF standby instruction (POFF(P))



Ⓢ : File name of the program to be set in the standby status by turning OFF the output, or the device where the file name is stored (character string)

Setting Data	Internal Devices		R, ZR	JGD		UJGD	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○						○	—

### ★ Function

- (1) Changes the execution type of the program with the file name stored in the device designated by Ⓢ.
  - Scan execution type: Turns OFF outputs at the next scan (Non-execution processing). Programs are set as the stand-by type after the subsequent scan.
  - Low speed execution type: Stops the execution of the low speed execution type program and turns OFF outputs at the next scan. Programs are set as the stand-by type after the subsequent scan.
- (2) Only the programs stored in the drive No. 0 (program memory) can be set as the stand-by type.
- (3) This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- (4) It is not necessary to designate the extension (.QPG) with the file name.  
(Only .QPG files will be acted on.)

### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The program with the file name specified by Ⓢ does not exist. (Error code: 2410)
  - The file name storage destination device of Ⓢ exceeds the range of the corresponding device. (Error code: 4101)

**Remark**

1. Non-execution processing is identical to the processing that is conducted when the condition contacts for the individual coil instructions are in the OFF state.

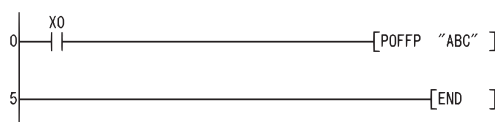
The operation results for the individual coil instructions following non-execution processing will be as follows, regardless of the ON/OFF status of the individual contacts:

OUT instruction	}	..... Forced OFF
SET instruction		
RST instruction	}	..... Maintains status
SFT instruction		
Basic instruction		
Application instruction		
PLS instruction	}	Processing identical to
Pulse generation instruction (□ P)		..... when condition contacts are OFF
Current value of low speed/high speed timer		..... 0
Current value of retentive timer	}	..... Preserves
Current value of counter		

**Program Example**

- (1) The following program makes the program with the file name ABC non-executionable and places it in the standby status when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	POFFP	"ABC"
5	END	

## 7.17.3 Program scan execution registration instruction (PSCAN(P))



Ⓢ : File name of the program to be set as a scan execution type, or head number of the devices where the file name is stored (character string)

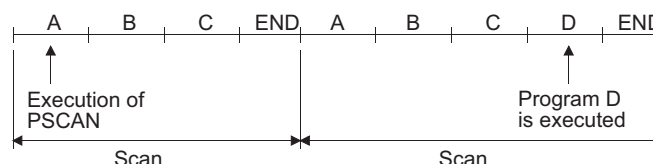
Setting Data	Internal Devices		R, ZR	JPG		UNPG	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○						○	—

### ★ Function

- (1) Sets the program whose file name is being stored at the device designated by Ⓢ in the scan execution type.
- (2) Only the programs stored in the drive No. 0 (program memory/internal RAM) can be set as the scan execution type.
- (3) Designated programs assume the scan execution type with END processing.

#### Example

When programs A, B, and C exist and program A performs "PSCAN" of program D.



- (4) This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- (5) It is not necessary to designate the extension (.QPG) with the file name.  
(Only .QPG files will be acted on.)

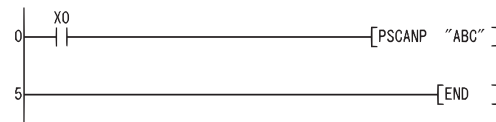
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The program with the file name specified by (S) does not exist. (Error code: 2410)
  - The file name storage destination device of (S) exceeds the range of the corresponding device. (Error code: 4101)
  - The specified file name is the SFC program, and the SFC program for the other file name has been already started. (Dual activation error of the SFC program)  
(For the Universal model QCPU) (Error code: 4131)  
(For the High Performance model QCPU, Process CPU, Redundant CPU) (Error code: 2504)

## Program Example

- (1) The following program sets the program with file name ABC as scan execution type when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	PSCANP	"ABC"
5	END	

# 7.17.4 Program low speed execution registration instruction (PLOW(P))



Ⓢ : File name of the program to be set as a low speed execution type, or head number of the devices where the file name is stored (character string)

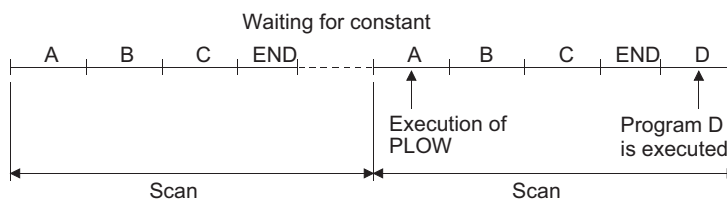
Setting Data	Internal Devices		R, ZR	JPG		UJGO	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○						○	—

## ★ Function

- (1) Sets the program whose file name is being stored at the device designated by Ⓢ in low-speed execution type.
- (2) Only the programs stored in the drive No. 0 (program memory/internal RAM) can be set as the low speed execution type.
- (3) Designated programs assume the low speed execution type with END processing.

### Example

When programs A, B, and C exist and program A performs "PLOW" of program D.  
(Assume that the constant scan has been set.)



- (4) This instruction will be given priority even in cases when a program execution type has been designated in the parameters.
- (5) It is not necessary to designate the extension (.QPG) with the file name.  
(Only .QPG files will be acted on.)

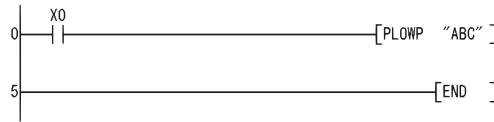
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The program with the designated file name does not exist. (Error code: 2410)
  - There is a CHK instruction contained within the program whose file name has been designated. (Error code: 4235)

## Program Example

- (1) The following program sets the program with file name ABC as low-speed execution type when X0 is turned ON.

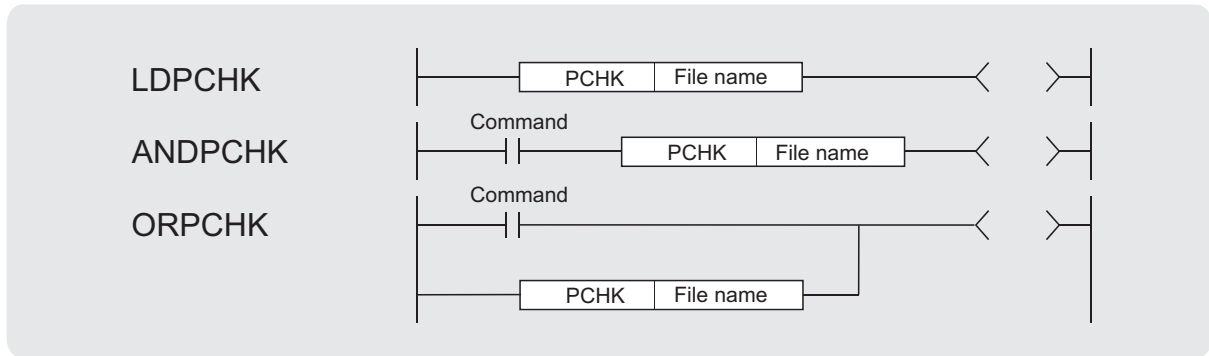
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	PLOWP	"ABC"
5	END	

## 7.17.5 Program execution status check instruction (PCHK)



Ⓢ : File name of the program whose execution status will be checked (character string)

Setting Data	Internal Devices		R, ZR	JOG		URGO	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ								○	—

### ★ Function

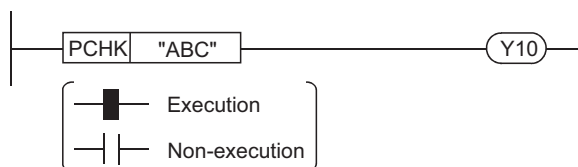
- (1) Checks whether the program of the specified file name is in execution or not (non-execution).
- (2) The instruction is in conduction when the program of the specified file name is in execution, and the instruction is in non-conduction when the program is in non-execution.
- (3) Specify the file name without an extension (.QPG).  
For example, specify "ABC" when the file name is ABC.QPG.

### ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The program with the designated file name does not exist. (Error code: 2410)

### 📄 Program Example

- (1) Program that keeps Y10 ON when the program file "ABC.QPG" is being executed.





**Remark**

Non-execution indicates that the program execution type is a stand-by type. Execution indicates that the program execution type is a scan execution type (including during output OFF (during non-execution processing)), low speed execution type or fixed scan execution type.

**POINT**

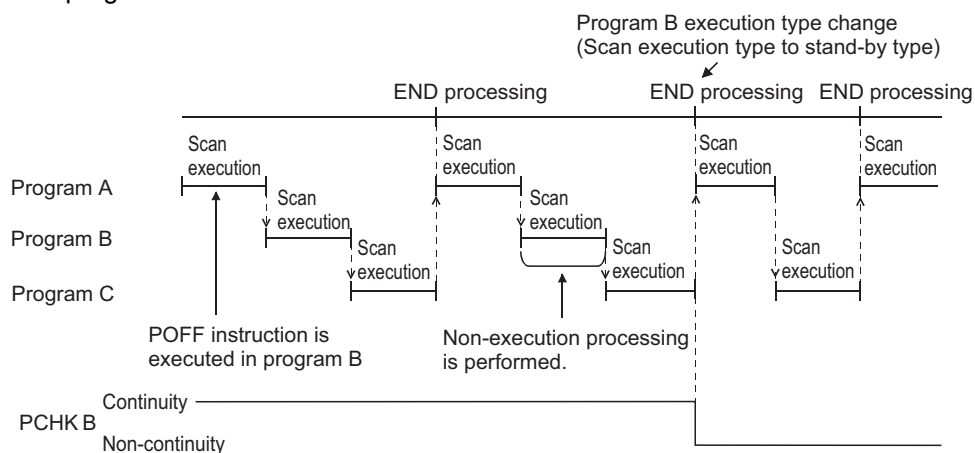
The PCHK instruction is in conduction when the program of the specified file name (target program) is in execution, and the instruction is in non-conduction when the program is in non-execution.

When the target program is set to non-execution (stand-by type) with the POFF instruction, the PCHK instruction is in conduction while the non-execution processing of the target program is being performed.

At the END processing of the scan where the non-execution processing is completed, the target program is put into non-execution (stand-by type), and the PCHK instruction is brought into non-conduction.

Therefore, note that if the PCHK instruction is executed for the program where the non-execution processing has been completed by the POFF instruction, the PCHK instruction may be brought into conduction.

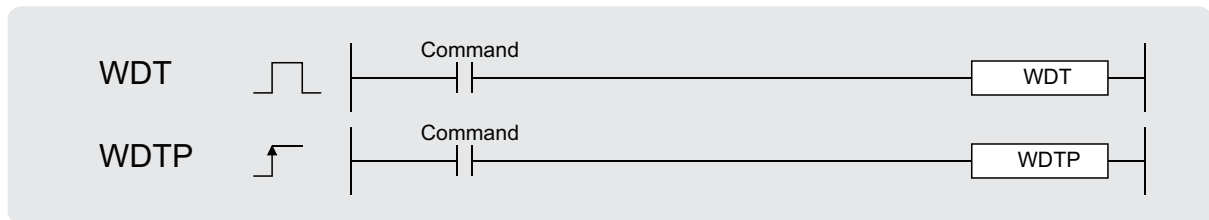
The following chart shows the operation performed when program A executes the POFF instruction of program B and program C executes the PCHK instruction of program B with the programs being executed in order of program A, program B and program C.



## 7.18 Other instructions

### 7.18.1 Resetting watchdog timer (WDT(P))

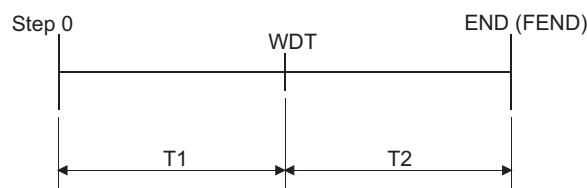
Basic High performance Process Redundant Universal



Setting Data	Internal Devices		R, ZR	JAD		UAGD	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

#### ★ Function

- (1) Resets watchdog timer during the execution of a sequence program.
- (2) Used in cases where the scan time exceeds the value set for the watchdog timer due to prevailing conditions.  
If the scan time exceeds the watchdog timer setting value on every scan, change the watchdog timer settings at the peripheral device parameter settings.
- (3) Make sure that the setting for t1 from step 0 to the WDT instruction and the setting for t2 from the WDT instruction to the END (FEND) instruction do not exceed the setting value of the watchdog timer.



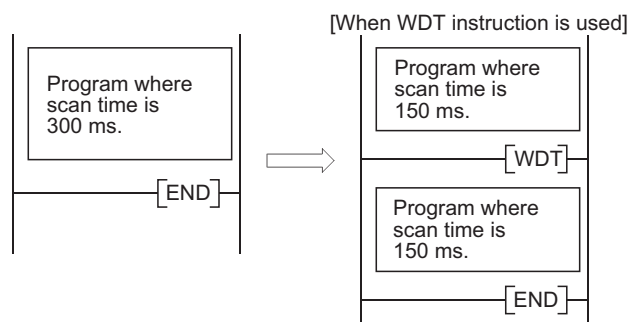
- (4) The WDT instruction can be used two or more times during a single scan, but care should be taken in such cases, because of the time required until the output goes OFF during the generation of an error.
- (5) Scan time values stored at the special register will not be cleared even if the WDT or WDTP instruction is executed.  
Accordingly, there are times when the value for the scan time for the special register is greater than the value of the watchdog timer set at the parameters.

## ! Operation Error

- (1) There are no operation errors associated with the WDT(P) instruction.

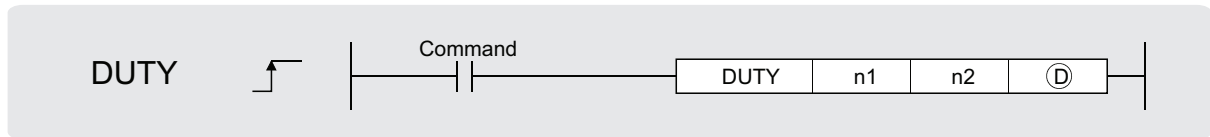
## Program Example

- (1) The following program has a watchdog timer setting of 200 ms, when due to the execution conditions program execution requires 300 ms from step 0 to the END (FEND) instruction.



## 7.18.2 Timing pulse generation (DUTY)

Basic High performance Process Redundant Universal



n1 : Number of scans for ON (BIN 16 bits)

n2 : Number of scans for OFF (BIN 16 bits)

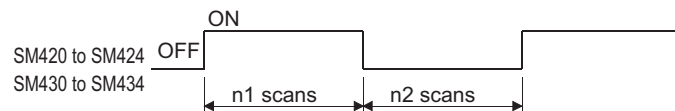
Ⓓ : User timing clock (SM420 to SM424, SM430 to M434) (bits)

Setting Data	Internal Devices		R, ZR	JOG		U <sub>GO</sub>	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n1	<input type="radio"/>					<input type="radio"/>			—
n2	<input type="radio"/>					<input type="radio"/>			—
Ⓓ	<input type="radio"/> *					—			—

\*1: Only SM420 to SM424, SM430 to SM434 can be used.

### ★ Function

- (1) Turns the user timing clock (SM420 to SM424, SM430 to M434), designated by Ⓓ, ON for the duration equivalent to the number of scans specified by n1, and OFF for the duration equivalent to the number of scans specified by n2.



- (2) Scan execution type programs use SM420 to SM424, and low speed execution type programs use SM430 to SM434.
- (3) The following will take place if both n1 and n2 have been set for 0:
  - (a) n1=0, n2≧0 SM420 to SM424 and SM430 to SM434 will stay OFF.
  - (b) n1>0, n2=0 SM420 to SM424 and SM430 to SM434 will stay ON.
- (4) The data designated by n1, n2, and Ⓓ is registered with the system when the DUTY instruction is executed, and the timing pulse is turned ON and OFF by END processing.

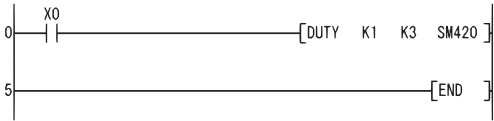
### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The device designated by Ⓓ is not from SM420 to SM424 or SM430 to SM434. (Error code: 4101)
  - The values of n1 and n2 are less than 0. (Error code: 4100)

# Program Example

(1) The following program turns SM420 ON for 1 scan, and OFF for 3 scans if X0 is ON.

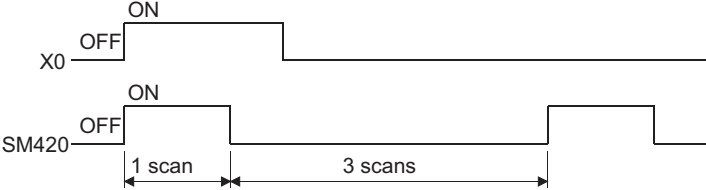
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DUTY	K1 K3 SM420
5	END	

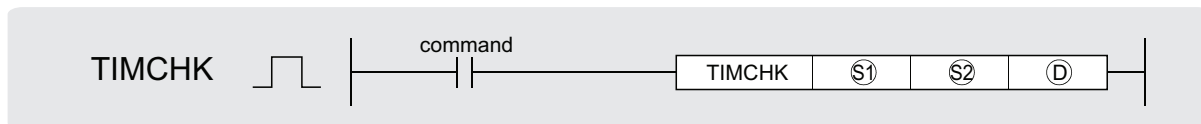
[Operation]



## 7.18.3 Time check instruction (TIMCHK)



Basic model QCPU: The upper five digits of the serial No. are "04122" or larger.



Ⓢ<sub>1</sub> : Device where the measured current value will be stored (BIN 16 bits)

Ⓢ<sub>2</sub> : Device where the set value of measurement is stored (BIN 16 bits)

Ⓧ : Device to be turned ON at time-out (bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ <sub>1</sub>	—	○				—			—
Ⓢ <sub>2</sub>	○	○				○			—
Ⓧ	○	—				—			—

### ★ Function

(1) Measures the ON time of the device used as a condition, and turns ON the device specified by Ⓢ<sub>2</sub> if the condition device remains ON for longer than the time set to the device specified by Ⓧ.

(2) The current value of the device specified by Ⓢ<sub>1</sub> is cleared to 0 and the device specified by Ⓧ is turned OFF at the leading edge of the execution command.

The current value of the device designated by Ⓢ<sub>1</sub> and the ON status of the device designated by Ⓧ are retained after the execution command turns OFF.

(3) Set the set value of measurement in units of 100ms.

### ! Operation Error

(1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

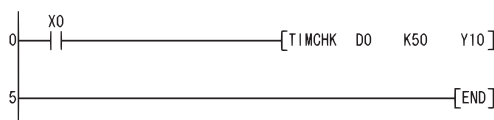
- The device that cannot be specified has been specified.

(Error code: 4100)

### 📄 Program Example

(1) Program where the ON time of X0 is set to 5s, the current value storage device to D0, and the device that will turn ON at time-out to Y10.

[Ladder Mode]

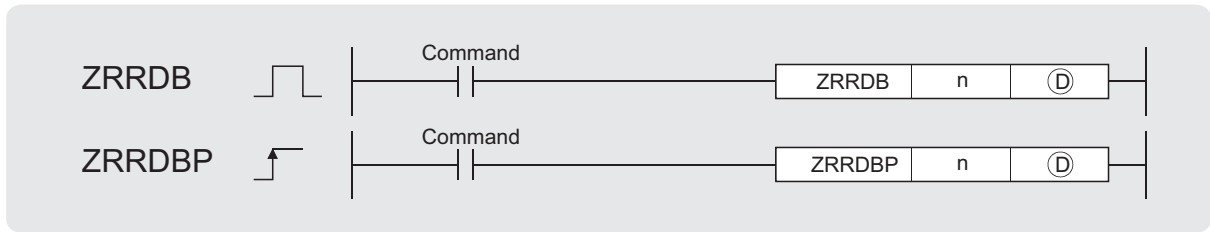


[List Mode]

Step	Instruction	Device
0	LD	X0
1	TIMCHK	D0 K50 Y10
5	END	

# 7.18.4 Direct 1-byte read from file register (ZRRDB(P))

Basic High performance Process Redundant Universal



n : Serial byte number for the file register to be read (BIN 32 bits)

(D) : Number of the device where the read data will be stored (BIN 16 bits)

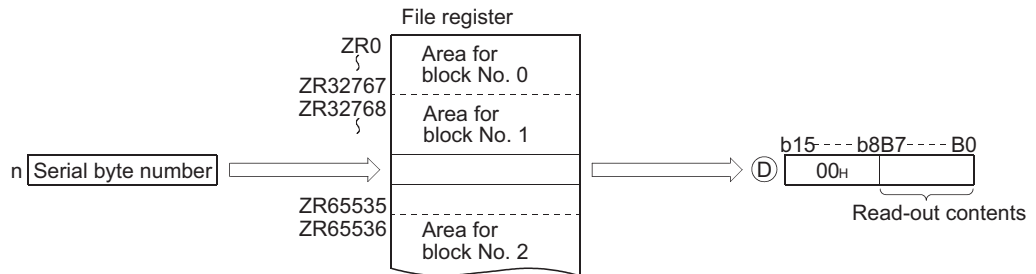
Setting Data	Internal Devices		R, ZR	JESD		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n					○			○	—
(D)					○			—	—

7

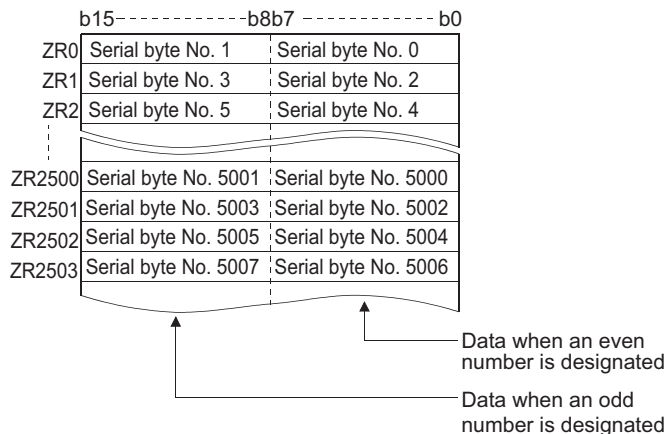
## ★ Function

- (1) Reads the serial byte number designated by n that does not signify a block number, and stores at the lower 8 bits of the device designated by (D).

The upper 8 bits designated by (D) will become 00H.

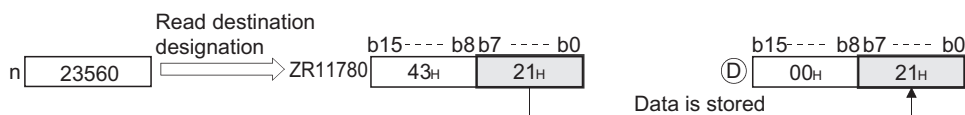


- (2) The correspondence between file register numbers and serial byte numbers is as indicated below:

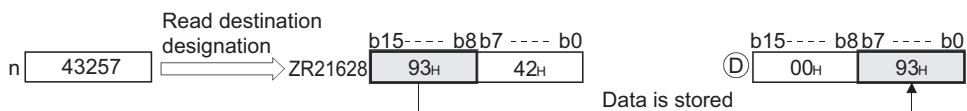


7.18 Other instructions  
7.18.4 Direct 1-byte read from file register (ZRRDB(P))

- (a) If the value of n has been designated as 23560, the data at the lower 8 bits of ZR11780 will be read.



- (b) If the value of n has been designated as 43257, the data at the upper 8 bits of ZR21628 will be read.



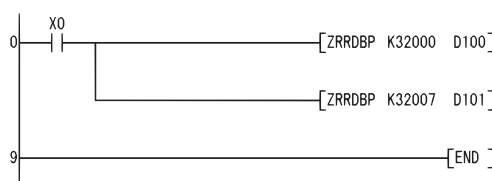
## ! Operation Error

- (1) In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- A device number (serial byte number) that exceeds the range of allowable designations has been designated. (Error code: 4101)

## Program Example

- (1) The following program reads the lower bits of ZR16000 and the upper bits of R16003, and stores results at D100 and D101 when X0 is ON.

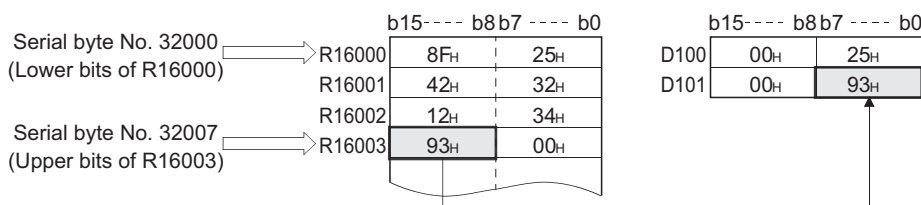
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	ZRRDBP	K32000 D100
5	ZRRDBP	K32007 D101
9	END	

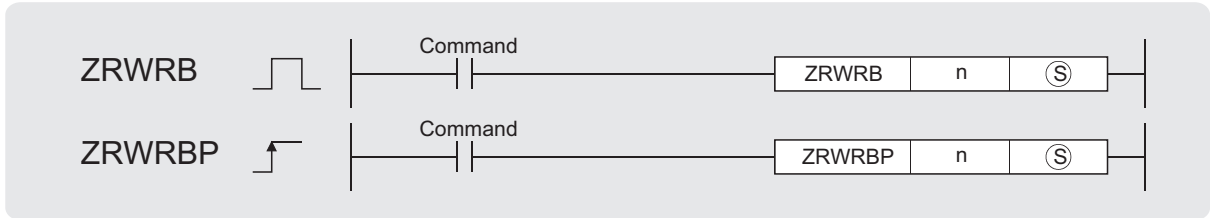
[Operation]





# 7.18.5 File register direct 1-byte write (ZRWRB(P))

Basic High performance Process Redundant Universal



n : Serial byte number for the file register to be written (BIN 32 bits)

S : Number of the device where the data to be written is stored (BIN 16 bits)

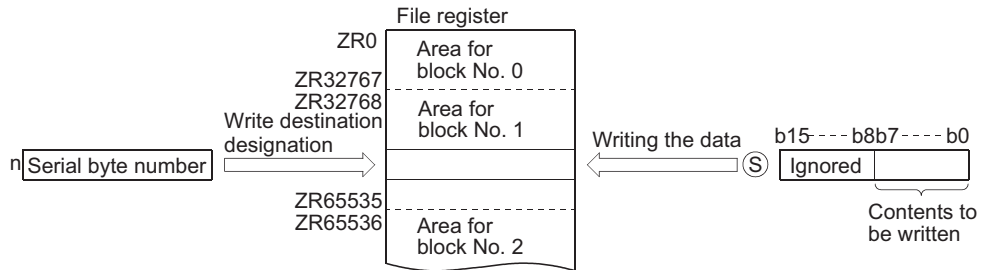
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n					○				—
S					○				—

7

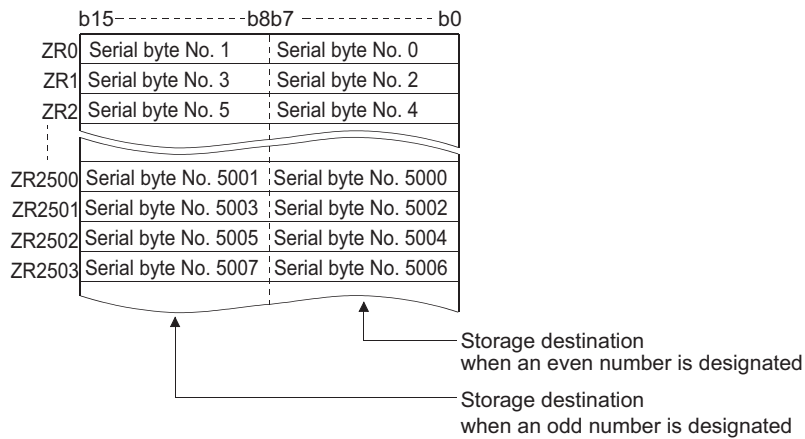
## ★ Function

- (1) Writes the lower bits of data stored in the device designated by S that does not signify a block number to the file register of the serial byte number designated by n.

The upper 8 bits of data in the device designated by S are ignored. S

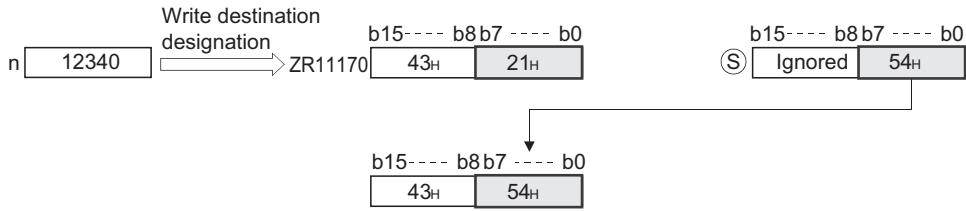


- (2) The correspondence between file register numbers and serial byte numbers is as indicated below:

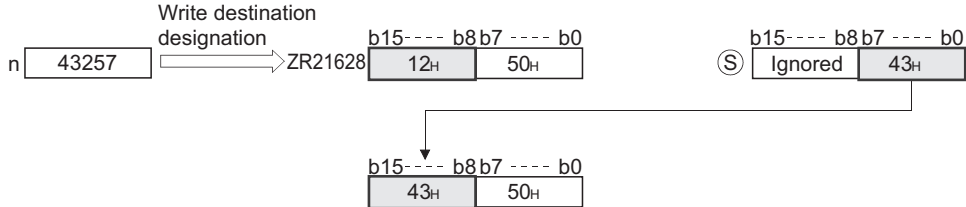


7.18 Other instructions  
7.18.5 File register direct 1-byte write (ZRWRB(P))

If n=12340 is specified, the data will be written to the lower 8 bits of ZR11170.



If n=43257 is specified, the data will be written to the upper 8 bits of ZR21628.



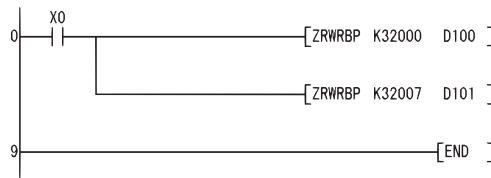
## ! Operation Error

- In the following case, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - A device number (serial byte number) that exceeds the range of allowable designations has been designated. (Error code: 4101)

## Program Example

- The following program writes the data at the lower bits of D100 and D101 to the lower bits of R16000 and the upper bits of R16003 when X0 is turned ON.

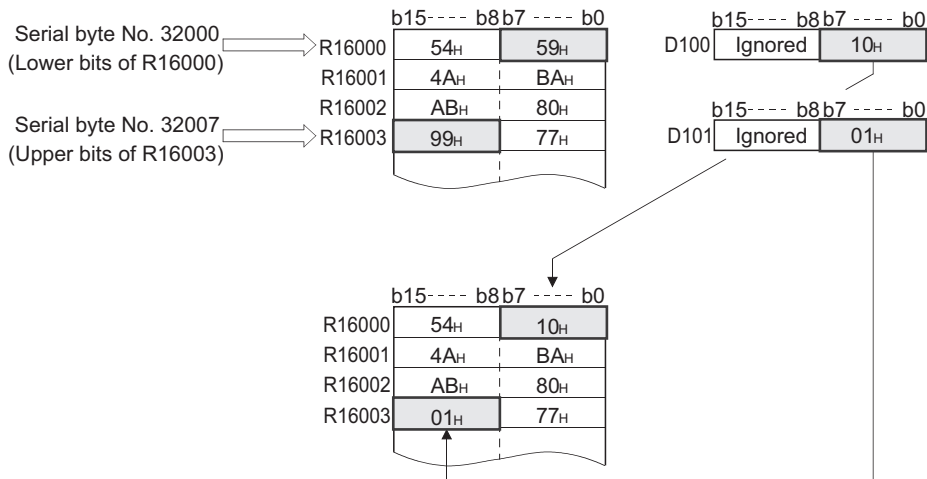
[Ladder Mode]



[List Mode]

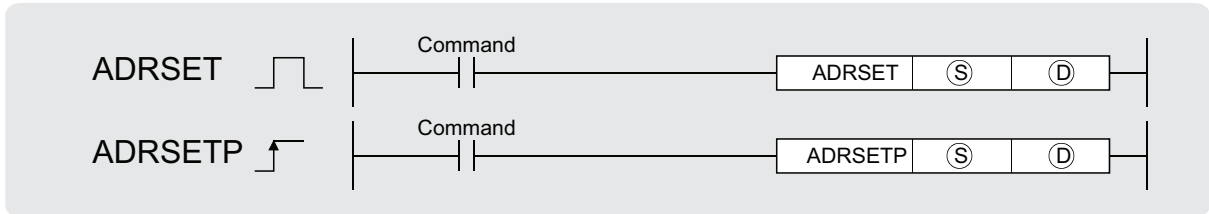
Step	Instruction	Device
0	LD	X0
1	ZRWRBP	K32000 D100
5	ZRWRBP	K32007 D101
9	END	

[Operation]



# 7.18.6 Indirect address read operations (ADRSET(P))

Basic High performance Process Redundant Universal



Ⓢ : Number of the device whose indirect address is read out (Device name)

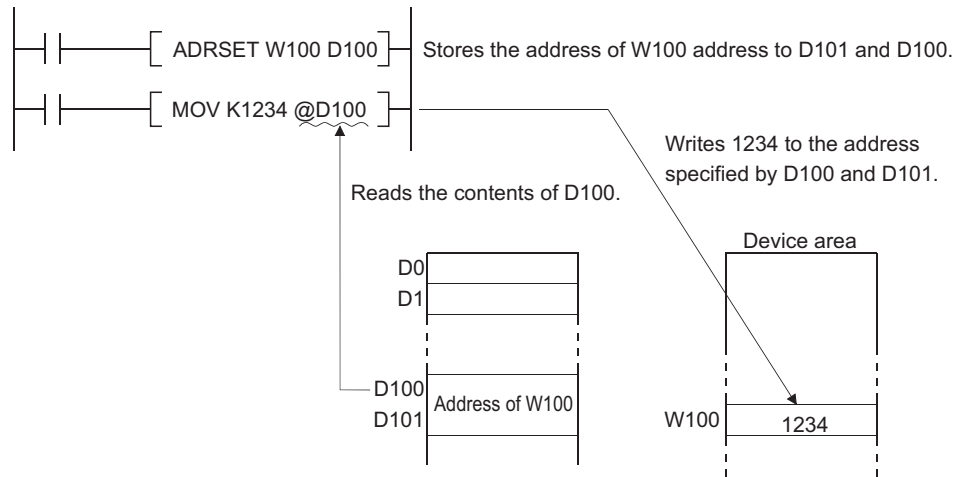
Ⓣ : Number of the device where the indirect address of the device designated by Ⓢ will be stored (BIN 32 bits)

Setting Data	Internal Devices		R, ZR	J		U/G	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓢ		○					—		
Ⓣ		○					—		

## ★ Function

- (1) Stores the indirect address of the device designated by Ⓢ at Ⓣ+1 and Ⓣ.

The address stored at the device designated by Ⓣ is used when an indirect device address is performed by the sequence program.



- (2) A bit device designation cannot be made at Ⓢ.

## ! Operation Error

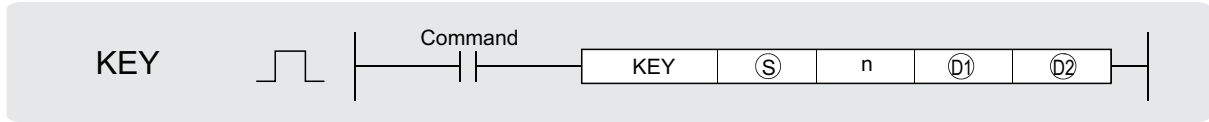
- (1) There are no operation errors associated with the ADRSET(P) instruction.

**Remark**

See Section 3.4 for further information on indirect designations.

7.18 Other instructions  
7.18.6 Indirect address read operations (ADRSET(P))

# 7.18.7 Numerical key input from keyboard (KEY)

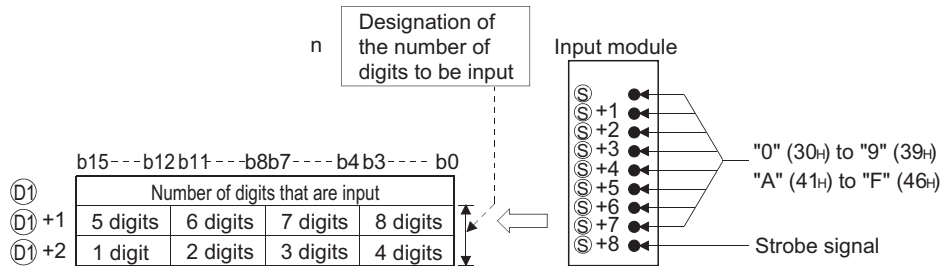


- Ⓢ : Head number of the devices (X) to which a numeral will be input (bits)
- n : Number of digits of the numeral to be input (BIN 16 bits)
- Ⓧ1 : Head number of the devices where the input numeral will be stored (BIN 16 bits)
- Ⓧ2 : Number of the bit device to turn ON at the completion of input (bits)

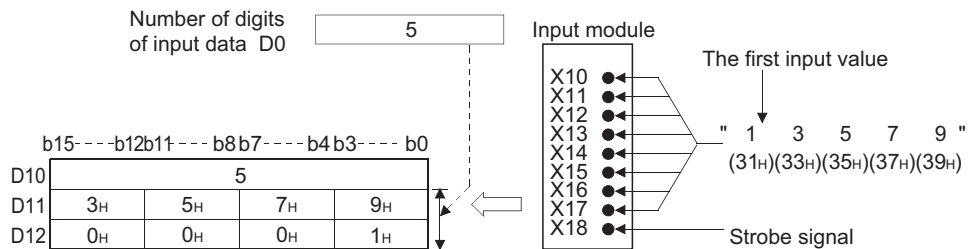
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	○ (Only X)	—			—		—		—
n	○	○			○		○		—
Ⓧ1	—	○			—		—		—
Ⓧ2	○	○			○		—		—

## ★ Function

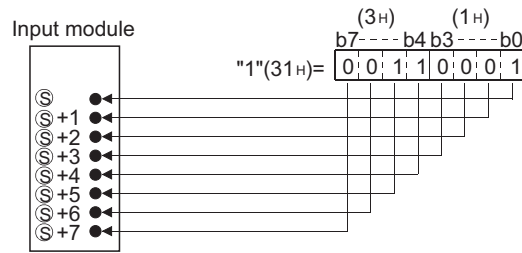
- (1) Fetches ASCII data from the 8 points of input (X) designated by Ⓢ, converts it to hexadecimal values and stores the result in the area starting from the device designated by Ⓧ1.



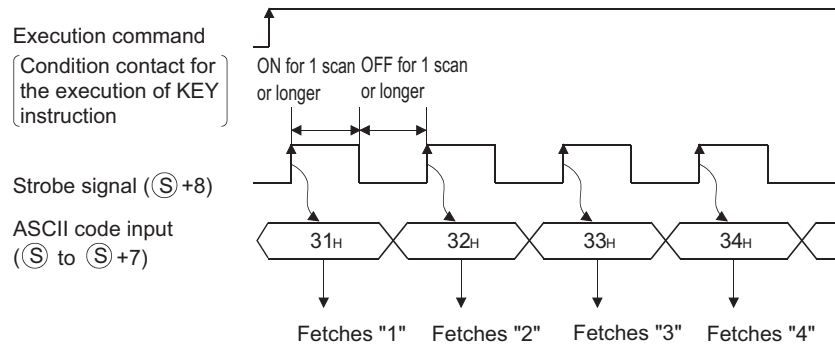
For example, in a case where the number of digits (n) has been set at 5, and the values "31", "33", "35", "37" and "39" have been input through X10 to X18 of the input module, the following will take place:



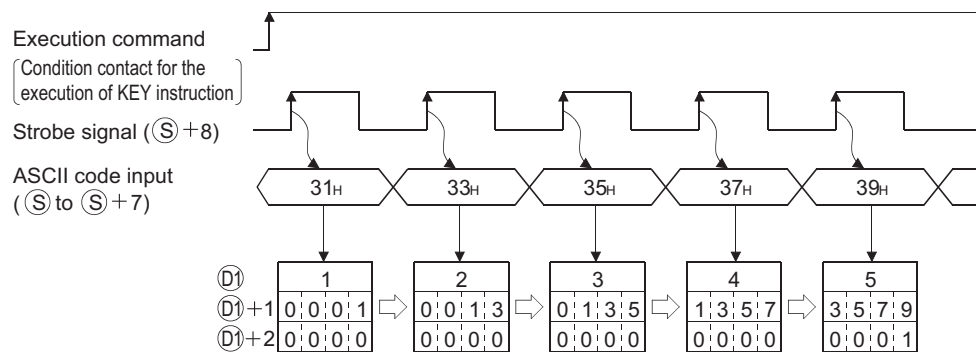
- (2) Numerical input to input (X) designated by (S) undergoes bit development at (S) through (S)+7 and is input as the ASCII code corresponding to the numbers.  
 ASCII code which can be input is from 30H (0) to 39H (9), and from 41H (A) to 46H (F).



- (3) After ASCII code is input to (S) to (S)+7, the strobe signal at (S)+8 goes ON to incorporate the designated numbers internally.  
 The strobe signal should be held at its ON or OFF status for more than one scan of the sequence program.  
 If this time is less than 1 scan, there will be cases when the data is correctly incorporated.



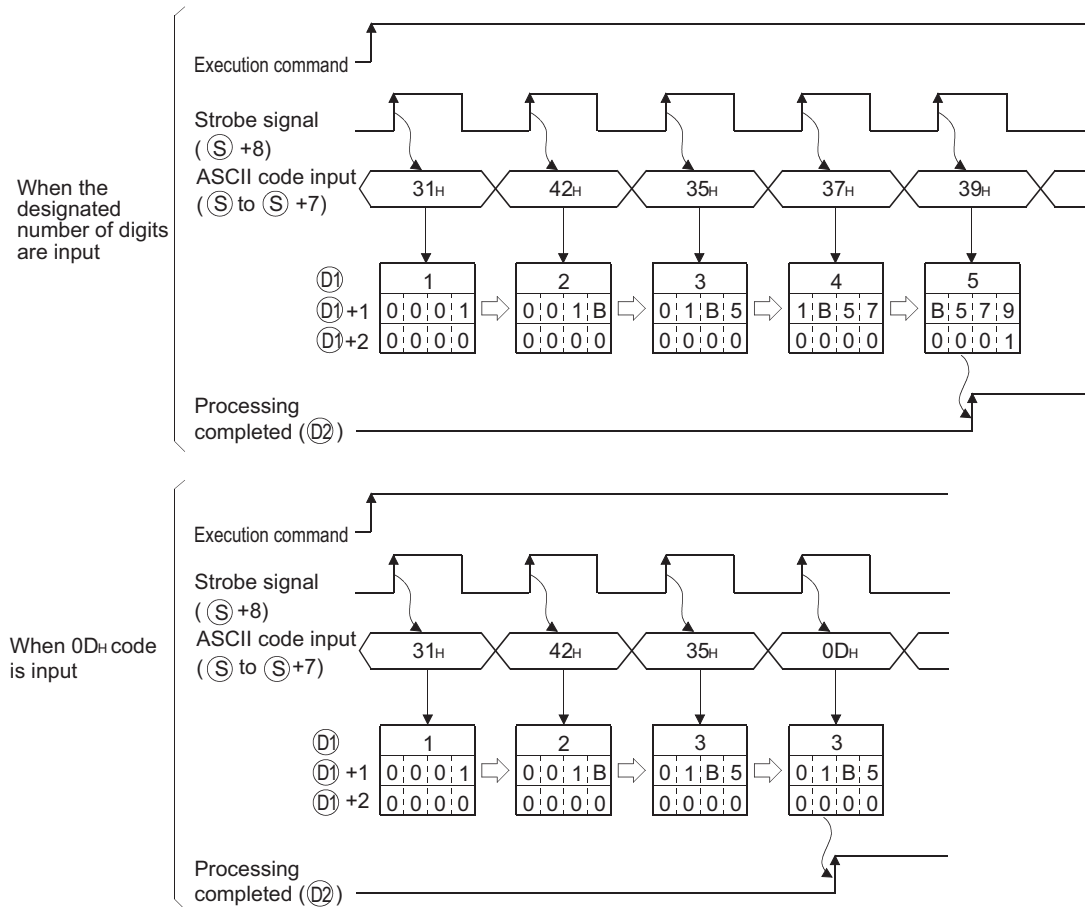
- (4) Be sure to keep the execution command (condition contact for the KEY instruction) ON until the specified number of digits has been input.  
 The KEY instruction cannot be executed if the execution command turns OFF.
- (5) The digits for the numbers actually fetched to (D) will be stored at the device designated by (D), and these will be converted to the ASCII codes input at (D)+1 and (D)+2, converted to hexadecimal BIN values, and stored.



- (6) The number of digits that can be designated by n is from 1 to 8.

- (7) Fetching of the input data is completed when any of the inputs shown below has been made.  
At the completion, the bit device designated by  $\textcircled{02}$  is turned ON.
- When the number of digits specified by  $n$  has been input
  - When the "0DH" code has been input

For example, the operations at the location designated if  $n = 5$  will be as indicated below:



If input processing is to be performed a second time, it is necessary to clear the number of digits input and the input data stored at  $\textcircled{D1}$ , and turn OFF the designated device at the user program.

If  $\textcircled{D1}$  is not cleared and  $\textcircled{D2}$  not turned OFF, the next input processing cannot be performed.



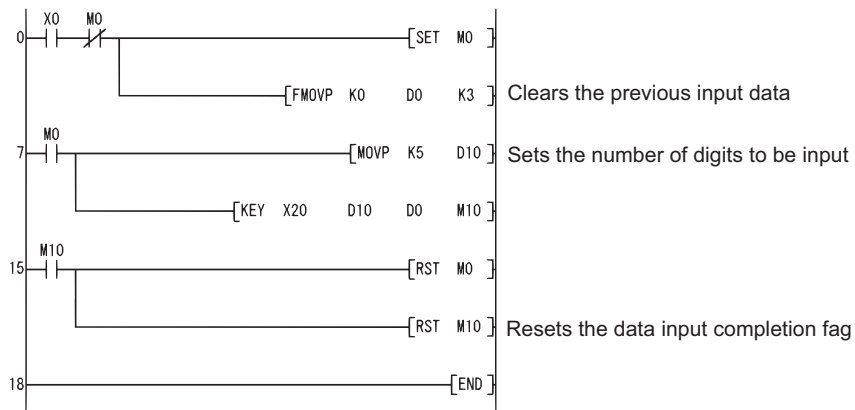
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device designated by  $\textcircled{D}$  is not an input (X) device. (Error code: 4100)
  - The number of digits designated by  $n$  are outside the range of from 1 to 8. (Error code: 4100)

## Program Example

- (1) The following program fetches data of the 5 or fewer digits from the numerical key pad connected to X20 to X28, and stores it to the area starting from D0 when X0 is turned ON.

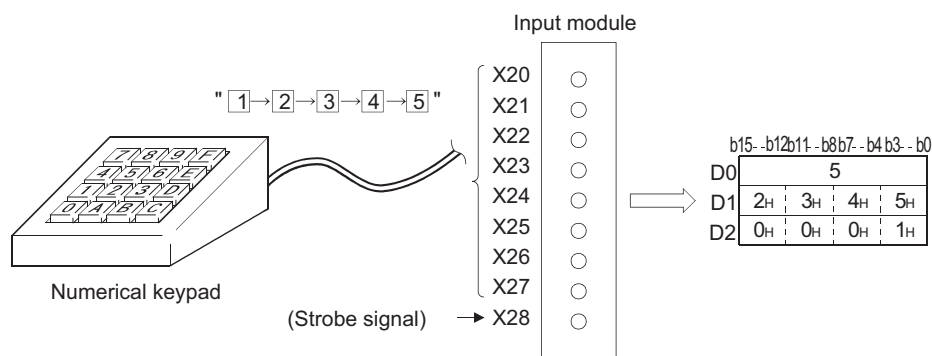
[Ladder Mode]



[List Mode]

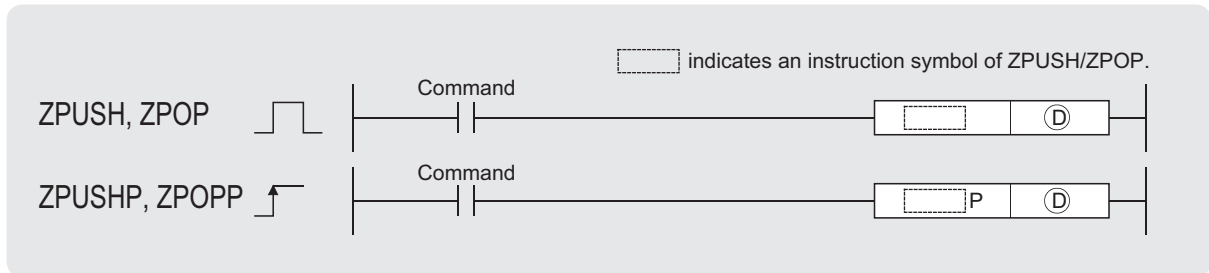
Step	Instruction	Device
0	LD	X0
1	ANI	MO
2	SET	MO
3	FMOVP	K0 D0 K3
7	LD	MO
8	MOV	K5 D10
10	KEY	X20 D10 D0 M10
15	LD	M10
16	RST	MO
17	RST	M10
18	END	

[Operation]



# 7.18.8 Batch save or recovery of index register (ZPUSH(P),ZPOP(P))

Basic High performance Process Redundant Universal



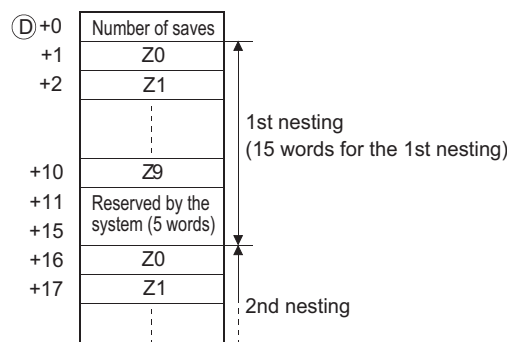
Ⓧ : Head number of the devices to/from which contents of an index register are saved/recovered (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	JAG		UAG	Zn	Constants	Other
	Bit	Word		Bit	Word				
Ⓧ	—	○					—		

## ★ Function

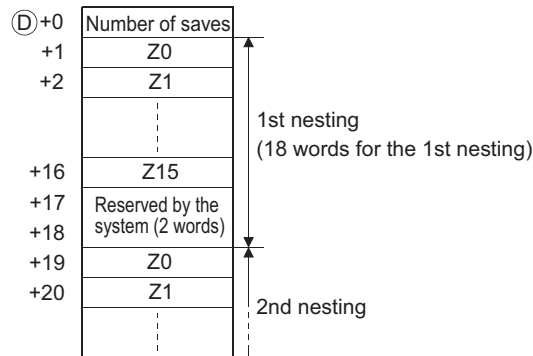
### ZPUSH

- Saves the contents of the following index registers to after the device specified by Ⓧ .  
(When contents of an index register are saved, Ⓧ + 0 (the number of saves made) is increased by 1.)
  - Basic model QCPU: Z0 to Z9
  - High Performance model QCPU/Process CPU/Redundant CPU: Z0 to Z15
  - Universal model QCPU: Z0 to Z19
- The ZPOP instruction is used for data recovery. Nesting is possible within the ZPUSH to ZPOP cycle.
- If nesting has been done, each time the ZPUSH instruction is executed, the field used following Ⓧ will be added to, so a field large enough to accommodate the number of times the instruction will be used should be maintained from the beginning.
- The composition of the field used following Ⓧ is as shown below:
  - When Basic model QCPU is used

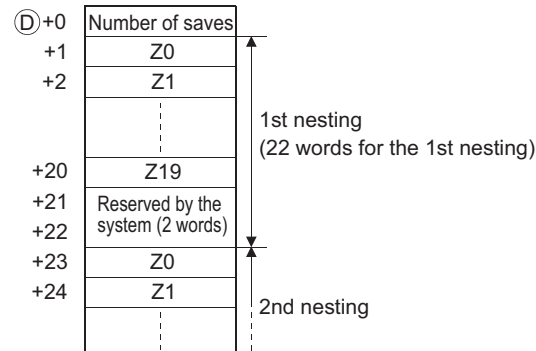




- When using a High Performance model QCPU/Process CPU/Redundant CPU



- When Universal model QCPU is used



## ZPOP

- (1) Recovers the contents saved in the area starting from the device designated by  $\textcircled{D}$  to the index register. (When the saved content is read out to the index register,  $\textcircled{D} + 0$  (the number of saves made) is decreased by 1.)

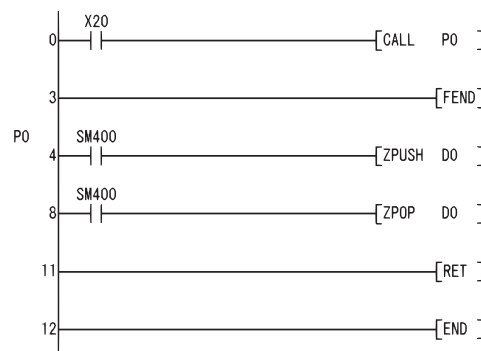
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The range for the number of points to be used at  $\textcircled{D}$  and later by the ZPUSH(P) instruction exceeds the corresponding device range. (Error code: 4101)
  - The contents of  $\textcircled{D} + 0$  (number of saves made) is 0 in the ZPOP(P) instruction. (Error code: 4100)

## Program Example

- (1) The following program saves the contents of the index register to the fields following D0 before calling the subroutine following P0 that uses the index register.

[Ladder Mode]

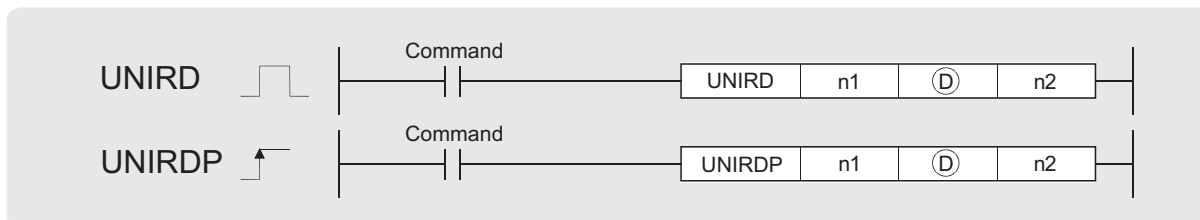


[List Mode]

Step	Instruction	Device
0	LD	X20
1	CALL	P0
3	FEND	
4		P0
5	LD	SM400
6	ZPUSH	D0
8	LD	SM400
9	ZPOP	D0
11	RET	
12	END	

# 7.18.9 Reading Module Information (UNIRD(P))

Basic High performance Process Redundant Universal



n1: Value obtained by dividing the head I/O number of the reading module information source by 16 (0 to FFn) (BIN 16 bits)

Ⓧ: Head number of the devices where the module information will be stored (device name)

n2: The number of points of read data (0 to 256) (BIN 16 bits)

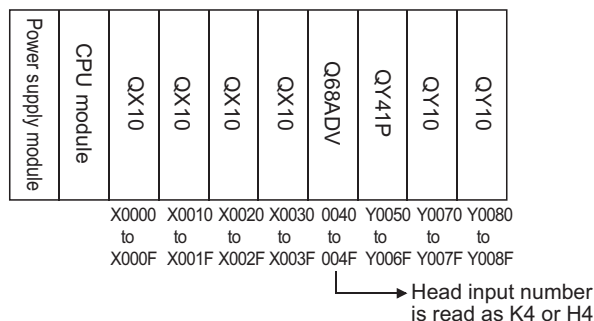
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n1	<input type="radio"/>	<input type="radio"/>						<input type="radio"/>	—
Ⓧ	—	<input type="radio"/>						—	—
n2	<input type="radio"/>	<input type="radio"/>						<input type="radio"/>	—

## ★ Function

- Reads the module information as much as designated by n2 from the module designated by n1 (value obtained by dividing the head I/O number by 16), and stores that information into the area starting from the device designated by Ⓧ. (Reads the status of the actually installed modules instead of the module type designated by I/O assignment.)

### Remark

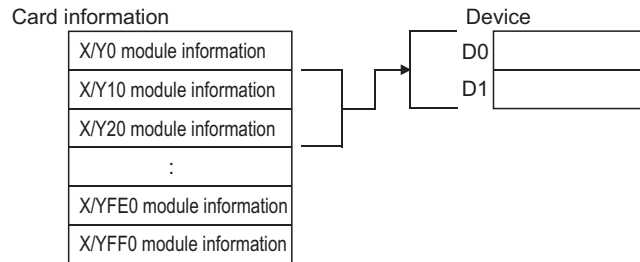
The value of n1 is designated by the higher 3 digits of the head I/O number of the slot from which the module information is read, when it is expressed in 4 digits in hexadecimal notation.



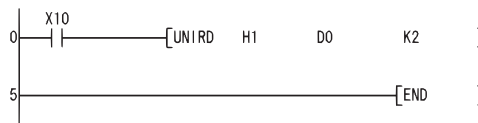


# Program Example

- (1) The following program stores the module information at I/O numbers 10H to 20H into the devices starting from D0 when X10 is turned ON.



[Ladder Mode]

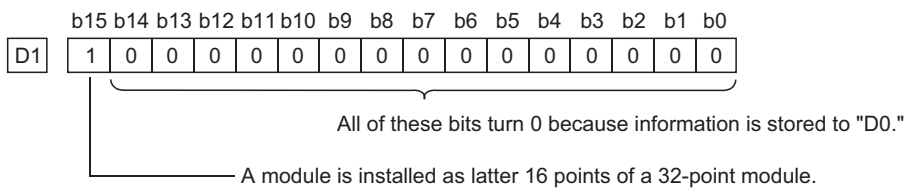
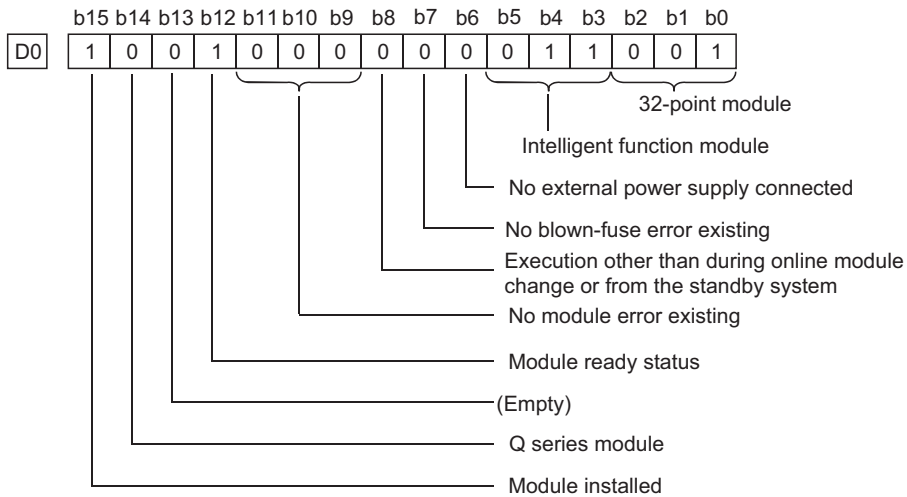


[List Mode]

Step	Instruction	Device
0	LD	X10
1	UNIRD	H1 D0 K2
5	END	

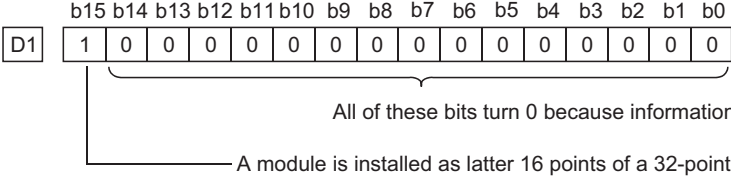
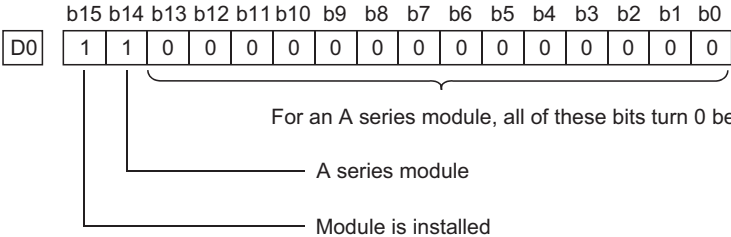
Readout result (When read to D0)

(a) 32-point intelligent function module for Q series



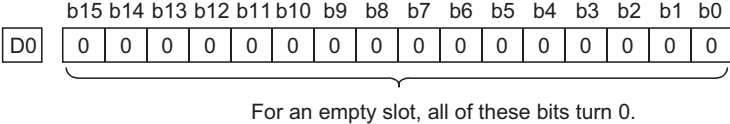
- With a 48- or 64-point module, the same contents as those of D1 are stored in D2 or D2 and D3 respectively.

(b) 32-point module for A series

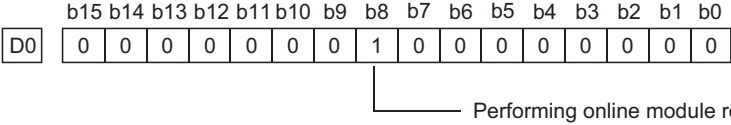


- With a 48- or 64-point module, the same contents as those of D1 are stored in D2 or D2 and D3 respectively.

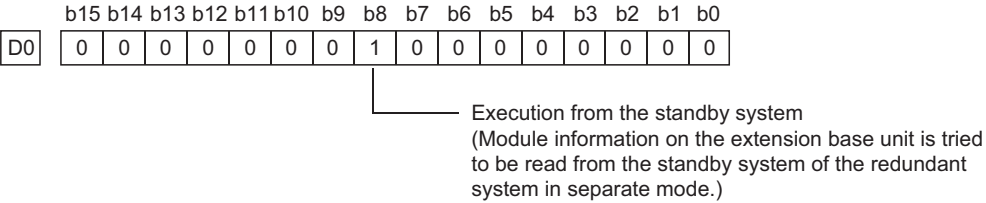
(c) Empty slot



(d) Performing online module replacement



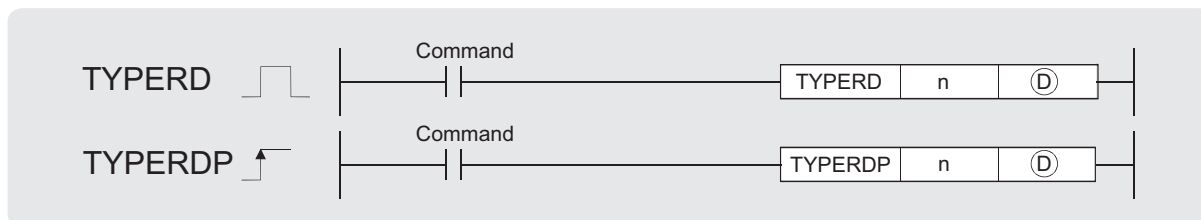
- (e) Module information on the extension base unit is tried to be read from the standby system of the redundant system in separate mode.



## 7.18.10 Reading module model name(TYPERD(P))



Universal model QCPU: The serial number (first five digits) is "11043" or later.



Setting data	Internal device		R, ZR	J:G:G		U:G:G	Zn	Constant K, H	Others
	Bit	Word <sup>6</sup>		Bit	Word				
n	—	○				—		○	—
Ⓓ	—	○				—		—	—

### ○ Set Data

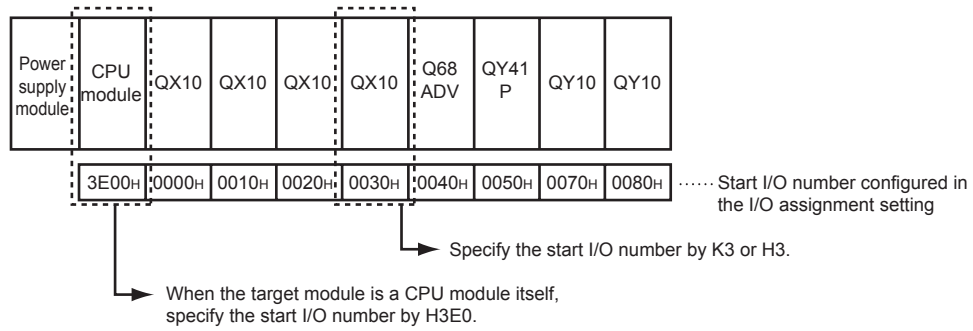
Setting data	Description		Setting range	Set by	Data type
n	Value obtained by dividing the start I/O number of a module whose model name is to be read by 16		0 to FFH, 3E0 to 3E3H	User	BIN 16 bits
Ⓓ	Ⓓ +0	Execution result of the instruction	Within each device range	System	BIN 16 bits
	Ⓓ +1 to Ⓓ +9	Module model name			Character string

### ★ Function

- (1) This instruction reads the module information stored in the area starting from the I/O number specified by "n", and stores it in the area starting from the device specified by Ⓓ. The following 6 modules (Q series only) support the instruction.
- CPU module
  - Input module
  - Output module
  - I/O combined module
  - Intelligent function module
  - GOT (bus connection)

(2) Specify the start I/O number of a module whose model name is to be read by "n" as follows:

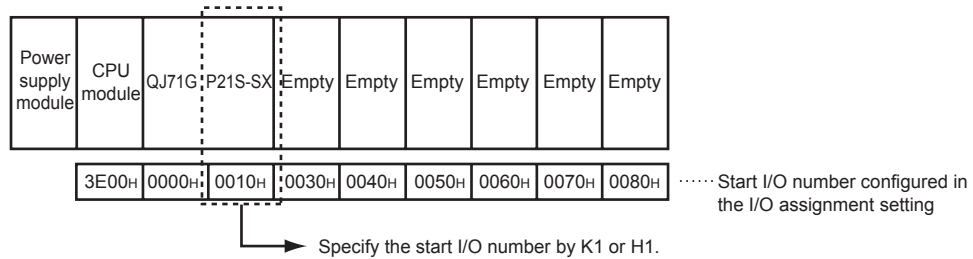
- Specify the value obtained by dividing the start I/O number of the target module by 16.



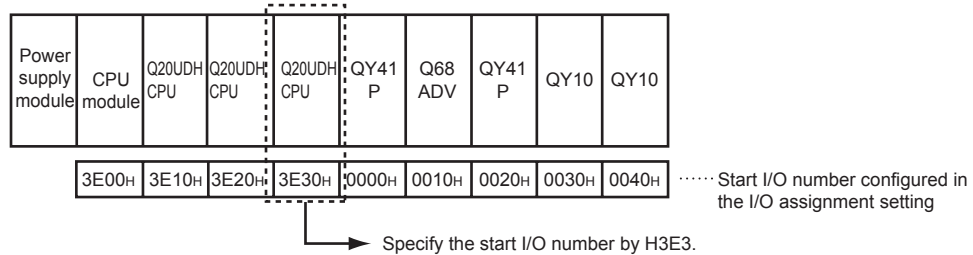
- When the target module occupies two slots  
The start I/O number to be specified may differ from that of the mounted module. For the start I/O number, refer to the manual of each module. Specify the value obtained by dividing the start I/O number of the target module by 16.

Example) QJ71GP21S-SX

Specify a value to which 0010H, start I/O number of the mounted module, is added.



- When the target module is a CPU module in multiple CPU systems  
Specify the value obtained by dividing the start I/O number of the target CPU module by 16.

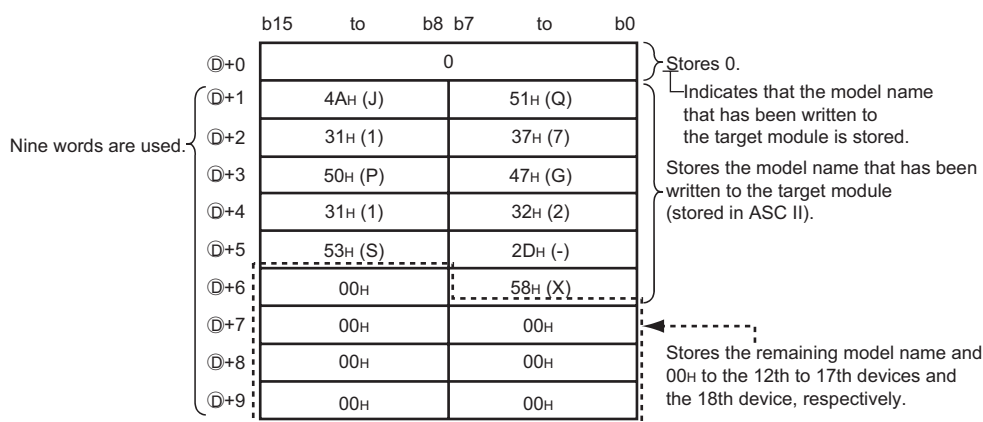


Or, the model name can be read by specifying the start I/O number of a module controlled by another CPU.

- (3)  $\text{D}+0$  and  $\text{D}+1$  to  $\text{D}+9$  store the execution result of the instruction and module model name, respectively.

A value stored in  $\text{D}$  is as follows:

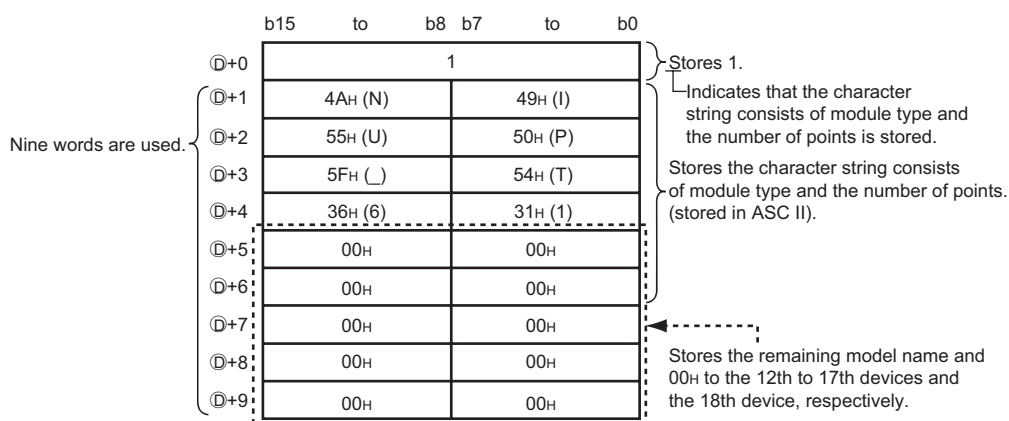
- (a) When the model name has been written to the target module (example: QJ71GP21-SX)



The following table shows the examples of model names stored in  $\text{D}+1$  to  $\text{D}+9$ .

Target module	Stored model name
CPU module	Q06UDEHCPU
Intelligent function module	QJ71GP21-SX
GOT	GOT1000

- (b) When the model name has not been written to the target module (example: QX40)



The following table shows the examples of character strings stored in  $\text{D}+1$  to  $\text{D}+9$ .

Target module	Stored character string
Input module	INPUT_16
Output module	OUTPUT_32
I/O combined module	MIXED_64
Intelligent function module	INTELLIGENT_128

[Character string indicating module type]

- Input module: INPUT
- Output module: OUTPUT
- I/O combined module: MIXED
- Intelligent function module\*1: INTELLIGENT
- 1: Includes the QI60 and GOT.

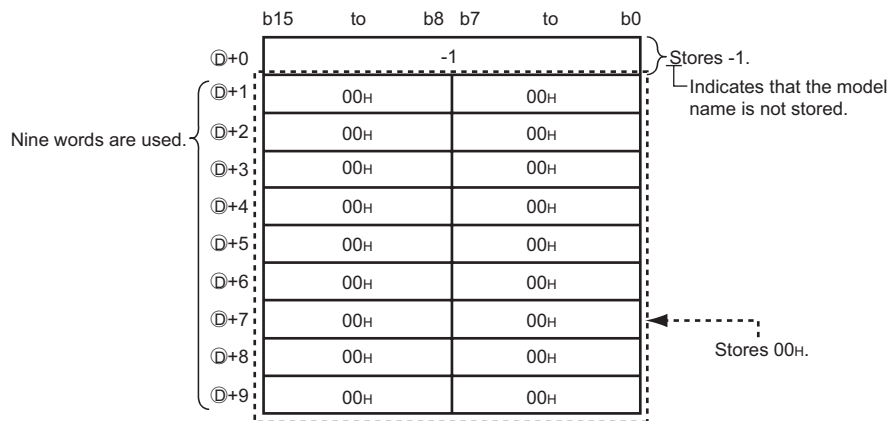


[Character string indicating the number of points]

- 16 points: `_16`
- 32 points: `_32`
- 48 points: `_48`
- 64 points: `_64`
- 128 points: `_128`
- 256 points: `_256`
- 512 points: `_512`
- 1024 points: `_1024`

(c) Others

- The specified slot is empty or the target module is during online module change.
- The specified value (n) is not the start I/O number.
- The specified value (n) is within the allowable setting range, but cannot be set in the I/O assignment setting screen of the PLC parameter dialog box.



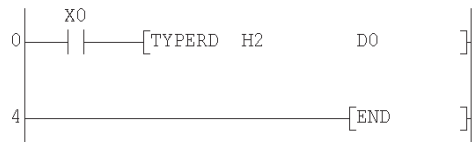
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.
- The target module cannot be communicated due to a failure. (Error code: 2110)
  - Devices by 10 words starting from the device specified by  $\text{D}$  exceed the device range. (Error code: 4101)
  - The specified value (n) is except 0 to FFH and 3E0 to 3E3H. (Error code: 4101)

## Program Example

- (1) The following program stores the model name of a module having the start I/O number 0020H in the area starting from the device specified by  $\text{\textcircled{D}}$  when X0 is turned on.

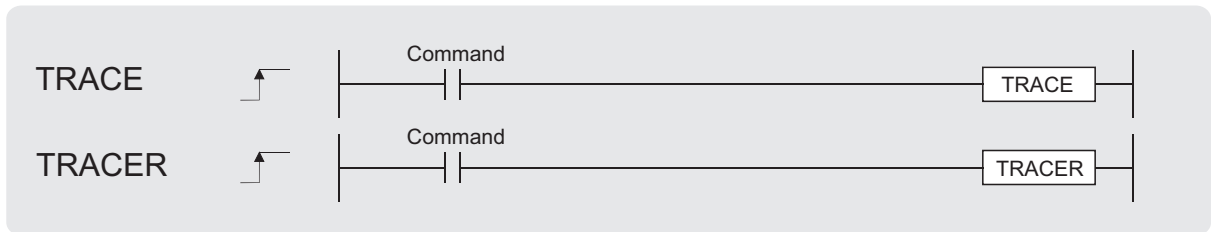
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	TYP ERD	H2
4	END	DO

## 7.18.11 Trace Set/Reset (TRACE, TRACER)

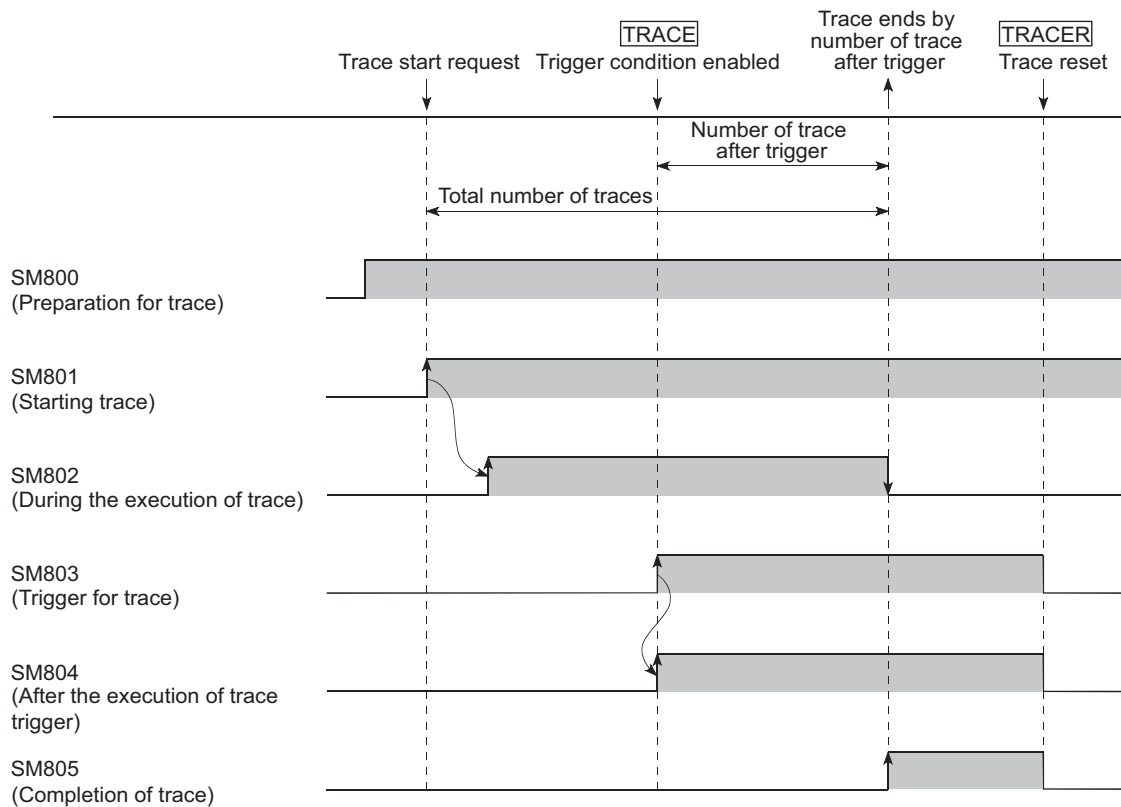


Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

### ★ Function

The sampling trace function collects the specified device data of a CPU module consecutively at the specified timing.

With the sampling trace function, the traced results obtained through the specified number of trace operations will be stored in the trace file of the memory card when SM800, SM801, and SM802 are turned ON.



## TRACE

- (1) The TRACE instruction turns ON SM803, executes sampling by the number of times set for "After trigger number of times" in the Trace condition settings, latches the data and stops sampling trace.
- (2) The sampling is stopped if SM801 is turned OFF during the trace execution.
- (3) After the TRACE instruction is executed and the trace is completed, SM805 is turned ON.
- (4) Once the TRACE instruction is executed, the second and the subsequent TRACE instructions are ignored.  
When the TRACER instruction is executed, the TRACE instruction is enabled again.

## TRACER

- (1) The TRACER instruction resets the TRACE instruction. When the TRACER instruction is executed, the TRACE instruction is enabled again.
- (2) When the TRACER instruction is executed, SM803 to SM805 are turned OFF.

### Remark

1. For details of the trace, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).
2. For trace execution with GX Developer, refer to the GX Developer Operating Manual.



## Operation Error

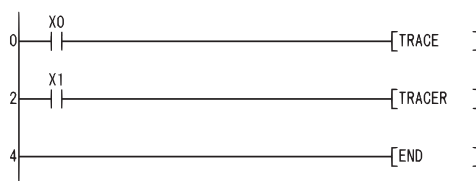
- (1) There are no operation errors associated with the TRACE or TRACER instruction.



## Program Example

- (1) The following program executes the TRACE instruction when X0 is turned ON, and resets the TRACE instruction with the TRACER instruction when X1 is turned ON.

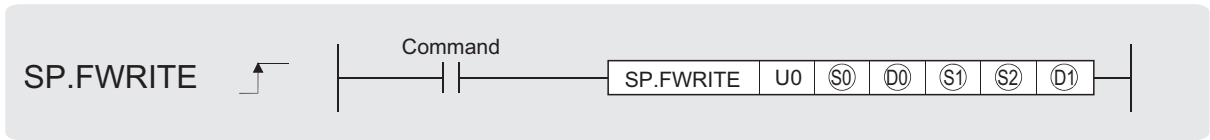
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	TRACE	
2	LD	X1
3	TRACER	
4	END	

# 7.18.12 Writing Data to Designated File (SP.FWRITE)



Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H	\$	
S0	○	○				—		○	—	—
D0	—	○				—		—	—	—
S1	—	○				—		—	—	—
S2	—	○				—		○	—	—
D1	△*1	△*1				—		—	—	—

\*1: Local devices and the devices designated for individual programs cannot be used.

Setting Data	Meaning		Setting Range	Set by	Data Type
U0	Dummy		—	—	
S0	Drive designation		2	User	
D0	Head number of the devices storing the control data. The following control data is required.				
	Device	Item	Contents/Setting Data	Setting Range	Set by
	D0	Execution/completion type	Designate the execution type. 0000H : Write binary data 0100H : Write data after CSV format conversion	0000H 0100H	User
	D0+1	(Not used)	Used by system	—	System
	D0+2	Writing result (No. of written data)	Contains the number of actually written data against the data designated by S2. The unit of the value depends on data type specified at D0+7.	—	System
	D0+3	(Not used)	—	—	—
	D0+4 D0+5	File position	Set the file position when binary data writing is specified by D0. 00000000H : Starting at the beginning of the file 00000001H to FFFFFFFEH : From the specified position The unit of the value depends on data type specification. FFFFFFFFH : Addition starts from the end of the file.  When CSV format write is specified at D0 • For the High Performance model QCPU of which the first 5 digits of the serial number are "01111" or lower, always set the beginning (0H) of the file. • For the High Performance model QCPU/ Process CPU/Redundant CPU/Universal model QCPU of which the first 5 digits of the serial number are "01112" or higher, set the file position. 00000000H to FFFFFFFEH : Starting at the beginning of the file FFFFFFFFH : Addition starts at the end of the file.	00000000H to FFFFFFFFH	User

7.18 Other instructions  
7.18.12 Writing Data to Designated File (SP.FWRITE)

Setting Data	Meaning			Setting Range	Set by	Data Type
Ⓚ	Ⓚ+6	No. of columns designation	When binary write is specified at Ⓚ, always set 0. When CSV format write is specified at Ⓚ, set the number of columns where data will be written. 0 : No columns. Regarded as one row. Other than 0 : Set to the specified number of columns.	0H to FFFFH (0 to 65535)	User	
	Ⓚ+7	Data type specification	0: Word 1: Byte	0,1	User	
Ⓛ	Head number of the devices storing a file name. A file name is expressed as follows:					
	Device	Item	Contents/Setting Data	Setting Range	Set by	BIN 16 bits
	Ⓛ to Ⓛ+□	File name character string	Designate the character string of a file name. • When omitting an extension, also omit the "." (Period). • Limit the file name within 8 characters + period + 3 characters. • When 9 or more characters are used, the extension is ignored regardless of its presence, and "BIN" or "CSV" is automatically assigned as an extension.	Character string	User	
Head number of the devices storing the data. Written data is expressed as follows:						
Ⓜ	Device	Item	Contents/Setting Data	Setting Range	Set by	
	Ⓜ	No. of request write data	Designate the number of data to request writing (word units). This data should be designated in units of words even when byte is designated by Ⓚ+7.	1 to 480 1 to 32767 *2	User	
	Ⓜ+1 to Ⓜ+□	Write data	Data to request writing.	0000H to FFFFH		
Ⓨ	Bit device that turned ON at the completion of the processing. (Ⓨ+1 is also turned ON at error completion.)					
	Device	Item	Contents/Setting Data	Setting Range	Set by	Bit
	Ⓨ	Completion signal	Indicates the completion of the processing. ON: Completed OFF: Not completed	—	System	
Ⓨ+1	Error completion signal	Indicates whether the processing is normally completed or abnormally completed. ON: Error completion OFF: Normal completion	—			

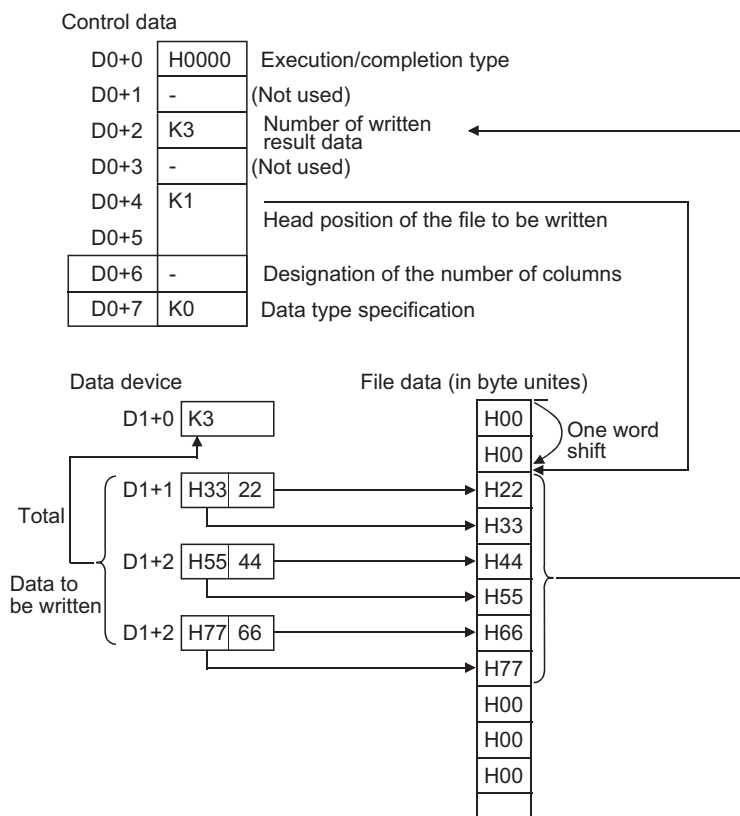
\*2: Indicates the range applicable only for the Universal model QCPU.

 **Caution**

- (1) At Ⓢ (drive designation), only the ATA card drive (2) can be set.  
Note that when the Flash card is loaded, the SP.FWRITE instruction cannot be used to perform writing.  
The SRAM card, standard RAM or standard ROM drive cannot be set.
- (2) For CSV setting, the data written are decimal values.  
**Example** Character "A" (41H) → "65" is written.  
Handling range: -32768 to 32767
- (3) For binary write, the word-specified file position setting range is 00000000H to 7FFFFFFFH and FFFFFFFFH.

# ★ Function

- (1) The designated number of data is written to the designated file.  
 Set the execution/completion type in the control data to designate whether to write binary data without any conversion or to convert binary data into CSV format data before writing it. (The writing target is the ATA card only.)
- (2) The execution completion bit device (D1) is automatically turned ON at the END processing after the completion of the instruction is detected. The bit device is turned OFF at the execution of the END instruction in the next scan.  
 Use this bit device as the execution completion flag for the SP.FWRITE instruction.  
 When this instruction is completed abnormally, the error completion device (D1+1) is turned ON/OFF in synchronization with the processing complete (D1) device. Use this device as the error completion flag for this instruction.  
 SM721 is turned ON during the execution of the instruction.  
 This instruction cannot be executed while SM721 is ON. (If an attempt is made, no processing is performed.)  
 When an error is detected at the execution of the instruction (before SM721 is turned ON), the processing complete device (D1), the error completion device (D1+1), and SM721 are not turned ON.
- (3) Be sure to use in units of words to designate the No. of request write data (S2) and the file position (D0+4 and D0+5).  
 The following shows the method for writing binary data when No. of request write data and file position are specified.





- (4) When writing binary data
- (a) If the extension of the target file is omitted, ".BIN" is used as an extension.
  - (b) When the designated file does not exist, a new file is created and the data is added/saved from the beginning of the file.  
The attributes of this new file are set using the archive attributes.
  - (c) When the size of the data exceeds that of the existing area in the file during the writing, the excess data is added/saved.
  - (d) If the file position specified is greater than the existing file size:
    - The High Performance model QCPU of which the first 5 digits of the serial number are "01111" or lower results in an error.
    - The High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU of which the first 5 digits of the serial number are "01112" or higher performs writing at point 0 and is completed normally.
  - (e) An error occurs when the saving space becomes full while data is added and saved. In such a case, the data that is successfully added/saved remains in the medium. The error completion is indicated after as much data as possible is added/saved.
- (5) When writing data after CSV format conversion
- (a) If the extension is omitted, ".CSV" is used as an extension.
  - (b) When the existing file is specified:
 

[High Performance model QCPU of which the first 5 digits of the serial number are "01111" or lower]  
File contents are all deleted and data are saved, starting at the beginning.

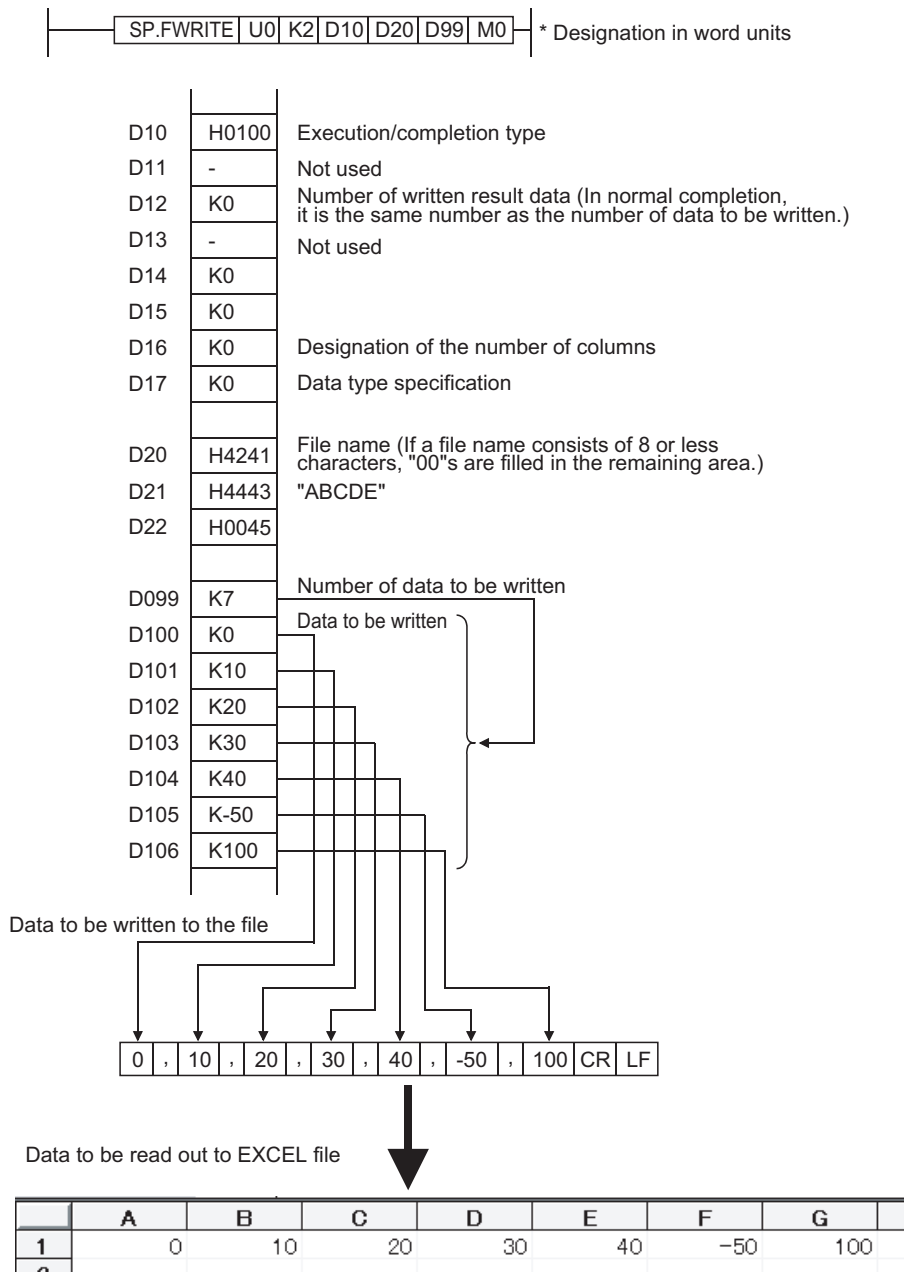
[High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU of which the first 5 digits of the serial number are "01112" or higher]

    - When other than FFFFFFFFH is set at (ⓐ+4, ⓐ+5), file contents are all deleted and data are saved, starting at the beginning.
    - When FFFFFFFFH is set at (ⓐ+4, ⓐ+5), data are saved, starting at the end of the file.
  - (c) When the designated file does not exist, a new file is created and the data is added/saved from the beginning of the file.  
The attributes of this new file are set using the archive attributes.
  - (d) An error occurs when the saving space becomes full while data is added and saved. In such a case, the data that is successfully added/saved remains in the medium. The error completion is indicated after as much data as possible is added/saved.

- (e) When the designated number of columns is "0", the data is stored as single-row data in CSV format file.

**Example**

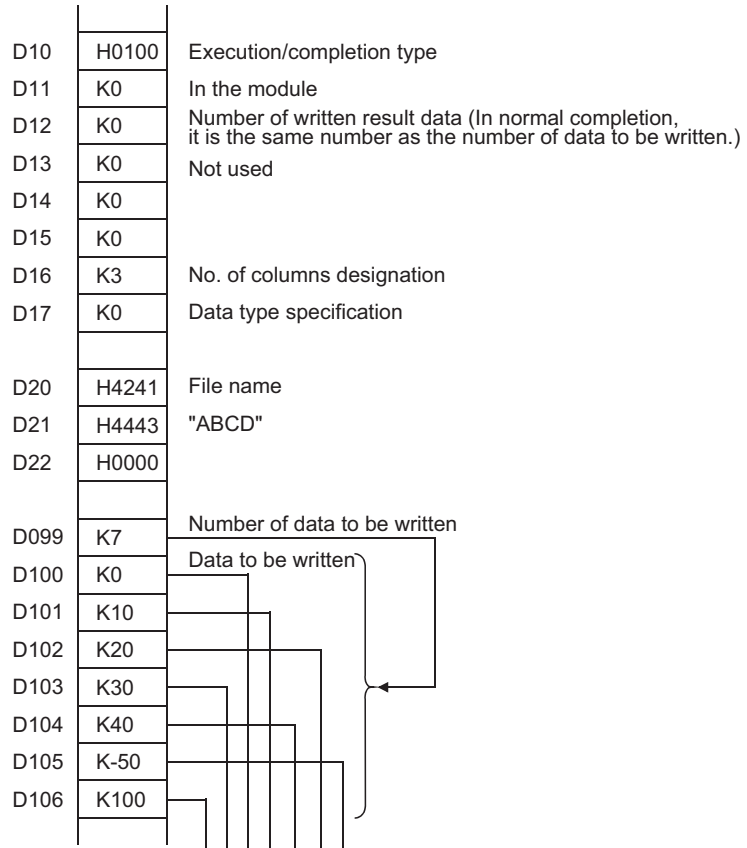
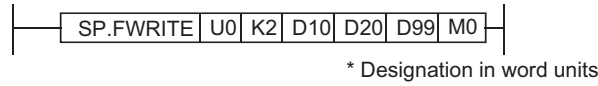
When data is written after CSV format conversion and the designated No. of columns is "0":



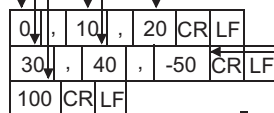
- (f) When data is written after CSV format conversion and the designated number of columns is other than "0", the data is stored as table data with designated number of columns in a CSV format file.

**Example**

When data is written after CSV format conversion and the designated No. of columns is other than "0":



Data to be written to the file



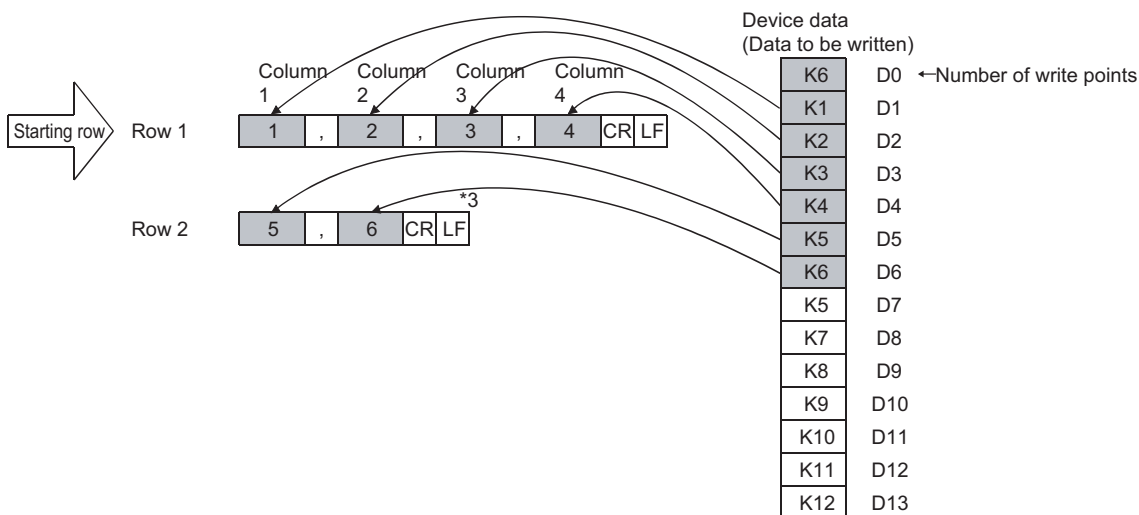
Data to be read to EXCEL file

	A	B	C
1	0	10	20
2	30	40	-50
3	100		

- (g) When data is added by the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU of which the first 5 digits of the serial number are 01112 or higher:

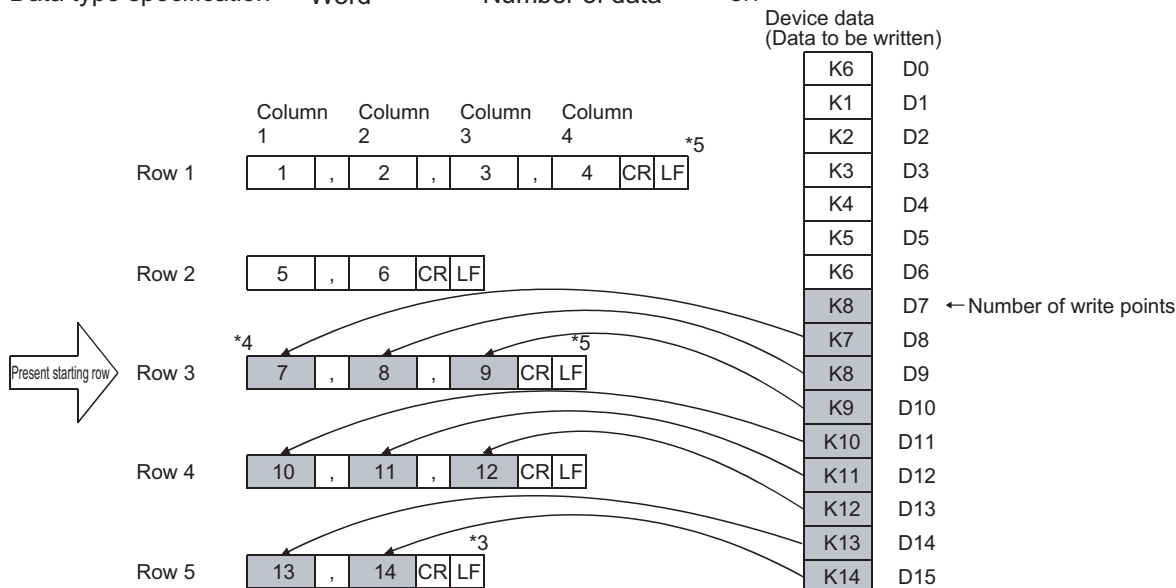
[Specify the file to which data will be written.] (If a file exists, delete it and create a new file again.)

Execution type = CSV format      File position = 0H (New file is created)  
 Column designation = 4H<sup>\*3,5</sup>      Write head device = D0  
 Data type specification = Word      Number of data = 6H<sup>\*3</sup>



[In the addition mode, make addition from the end of the file.]

Execution type = CSV format      File position = FFFFFFFFH (Continuation mode)  
 Column designation = 3H<sup>\*3,5</sup>      Write head device = D7  
 Data type specification = Word      Number of data = 8H<sup>\*3</sup>



\*3: Unless the "number of write points" is set to an integral multiple of "column designation", the column numbers will be random.

\*4: Since the last data is always followed by the line feed code, addition normally starts at the beginning of the new row in the addition mode.

\*5: If, in the addition mode, "column designation" is changed from that in the previous writing, the column numbers are shifted.

- (h) Do not execute the SP.FWRITE instruction in an interrupt program.  
 (If execute it, the operation is not guaranteed.)

## Operation Error

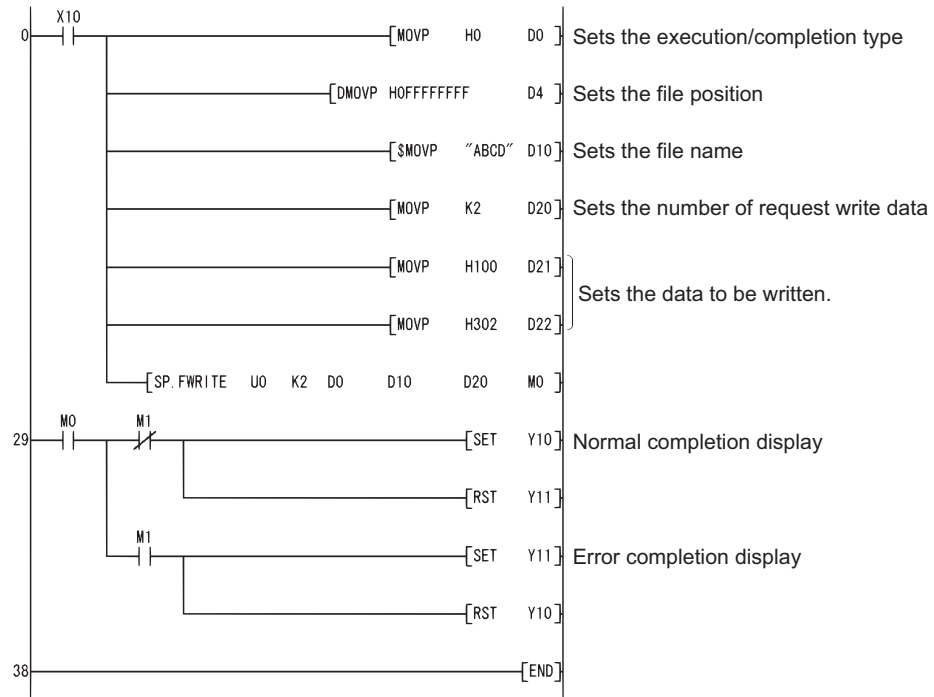
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Drive specified by drive designation device (S0) contains the medium other than the ATA card. (Error code: 4100)
  - Values specified in control data (D0) and the subsequent devices are out of the setting range. (Error code: 4100)
  - Value designated by "No. of request write data" (S2) is out of the setting range, or exceeds the device range designated by (S2 + 1) or the subsequent devices. (Error code: 4101)
  - Empty space in the ATA card is insufficient. (Error code: 4100)
  - No free space is found when an attempt is made to create a new file. (Error code: 4100)
  - Invalid device is designated. (Error code: 4004)
  - Access error occurred in the ATA card. (Error code: 4100)
  - An unusable value is set for a file name (S1). (Error code: 4100)
  - The attribute of a file name (S1) is "read only". (Error code: 4100)
  - The device specified by (D0) or (D1) exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

## Program Example

(1) When X10 is turned ON, the following program adds four bytes of binary data (00H, 01H, 02H, and 03H) to file "ABCD.BIN" in the memory card inserted to drive 2.

- Assume that 8 points from (D0) are reserved for the control data devices.

[Ladder Mode]

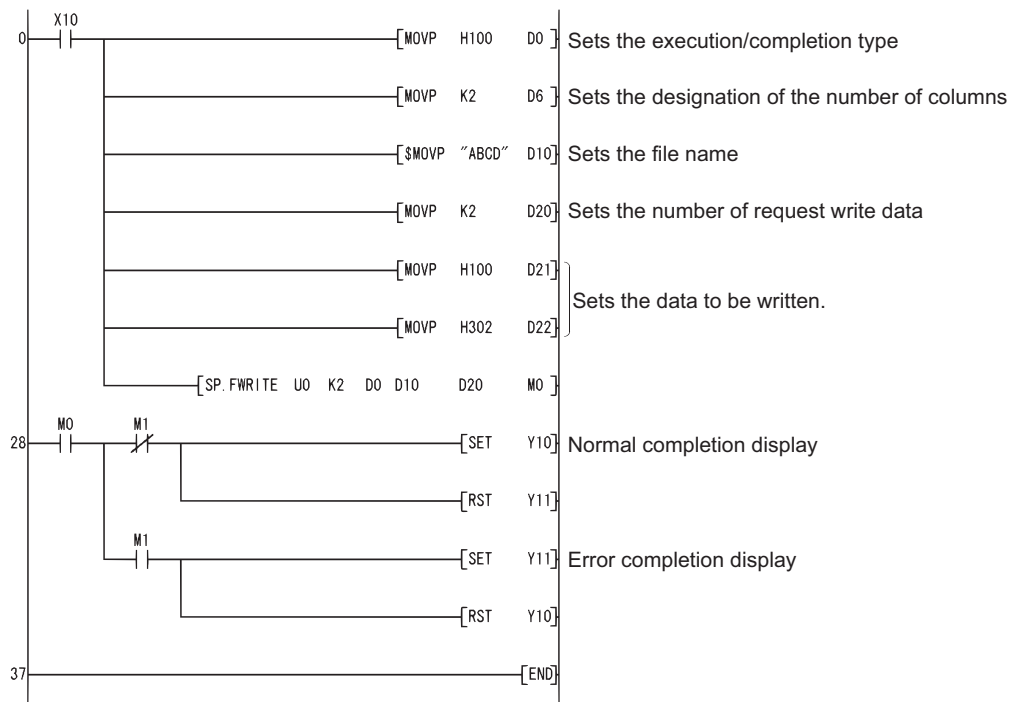


[List Mode]

Step	Instruction	Device
0	LD	X10
1	MOV P	H0 D0
3	DMOV P	H0FFFFFF D4
6	\$MOV P	"ABCD" D10
11	MOV P	K2 D20
13	MOV P	H100 D21
15	MOV P	H302 D22
17	SP.FWRITE	U0 K2 D0 D10 D20 M0
29	LD	M0
30	MPS	
31	ANI	M1
32	SET	Y10
33	RST	Y11
34	MPP	
35	AND	M1
36	SET	Y11
37	RST	Y10
38	END	

- (2) When X10 is turned ON, the following program creates a file named "ABCD.CSV" in the memory card inserted to drive 1, and writes four bytes of data (00H, 01H, 02H, and 03H) as two-column table data in CSV format.
- The written file is displayed as follows:

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	MOV	H100 D0
3	MOV	K2 D6
5	SMOV	"ABCD" D10
10	MOV	K2 D20
12	MOV	H100 D21
14	MOV	H302 D22
16	SP.FWRITE	U0 K2 D0 D10 D20 M0
28	LD	M0
29	MPS	
30	ANI	M1
31	SET	Y10
32	RST	Y11
33	MPP	
34	AND	M1
35	SET	Y11
36	RST	Y10
37	END	

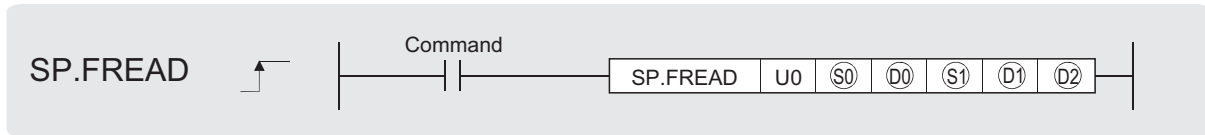
- Assume that 8 points from ⑩ are reserved for the control data devices.

0	,	1	,	CR LF	Contents of the file to be written
2	,	3	,	CR LF	

↓ Data to be read to the EXCEL file

	A	B
1	0	1
2	2	3

# 7.18.13 Reading Data from Designated File (SP.FREAD)



Setting Data	Internal Devices		R, RZ	J:GO		U:GO	Zn	Constants		Other
	Bit	Word		Bit	Word			K, H	\$	
(S0)	○	○				—		○	—	—
(D0)	—	○				—		—	—	—
(S1)	—	○				—		—	—	—
(D1)	—	○				—		—	○	—
(D2)	△*1	△*1				—		—	—	—

\*1: Local devices and the devices designated for individual programs cannot be used.

Setting Data	Meaning			Setting Range	Set by	Data Type
U0	Dummy			—	—	
(S0)	Drive designation			2	User	
(D0)	Head number of the devices storing the control data The following control data is required.					BIN 16 bits
	Device	Item	Contents/Setting Data	Setting Range	Set by	
	(D0)	Execution/ completion type	Designate the execution type. 0000H: Read binary data 0100H: Read data after CSV format conversion	0000H 0100H	User	
	(D0)+1	(Not used)	Used by system	—	System	
	(D0)+2	No. of request read data	Designate the number of data to request reading. (Unit: Word) Even when byte is specified at (D0)+7 by data type specification, specify the value in units of words (16 bits), not in units of bit devices.	1 to 480 1 to 32767*2	User	
(D0)+3	(Not used)	—	—	—		

\*2: Indicates the range applicable only for the Universal model QCPU.



Setting Data	Meaning		Setting Range	Set by	Data Type	
D0	D0+4 D0+5	File position	<p>Designate the file position to start reading when binary data reading is designated by D0.</p> <p>00000000H: Starting at the beginning of the file 00000001H to FFFFFFFEH :From the designated position The unit for the value is determined by word/byte unit designation.</p> <p>FFFFFFFH: Setting disabled</p> <p>When CSV format read is specified at D0</p> <ul style="list-style-type: none"> <li>For the High Performance model QCPU of which the first 5 digits of the serial number are "01111" or lower, always set the beginning (0H) of the file.</li> <li>For the High Performance model QCPU/ Process CPU/Redundant CPU/Universal model QCPU of which the first 5 digits of the serial number are "01112" or higher, set the file position (Row).</li> </ul> <p>00000000H: Read starts at the beginning of the file. 00000001H to FFFFFFFEH :Read starts at the specified row. FFFFFFFH: Read continues, starting at the previous read position.</p>	00000000H to FFFFFFFFH	User	BIN 16 bits
	D0+6	No. of columns designation	<p>When binary read is specified at D0, always set 0.</p> <p>When CSV format read is specified at D0, set the number of columns from where data will be read.</p> <p>0: No columns. Regarded as one row. Other than 0: Regarded as the specified number of columns.</p>	0H to FFFFH (0 to 65535)	User	
	D0+7	Data type specification	<p>0: Word 1: Byte</p>	0,1	User	
S1	Head number of the devices storing a file name. A file name is expressed as follows:					
	Device	Item	Contents/Setting Data	Setting Range	Set by	
S1 to S1+□	File name character string	<p>Designate the character string of a file name.</p> <ul style="list-style-type: none"> <li>When omitting an extension, also omit the "." (Period).</li> <li>Limit the file name within 8 characters + period + 3 characters.</li> <li>When 9 or more characters are used, the extension is ignored regardless of its presence, and "BIN" or "CSV" is automatically assigned as an extension.</li> </ul>	Character string	User		
D1	Head number of the devices for storing the read data.					
	Device	Item	Contents/Setting Data	Setting Range	Set by	
	D1	Reading result (No. of read data)	<p>Contains the number of actually read data against the data designated by D0+2. The unit on the value depends on data type specification.</p>	—	System	
D1+1 to D1+□	Reading data	Read data	—	System		

7.18 Other instructions  
7.18.13 Reading Data from Designated File (SP.FREAD)

Setting Data	Meaning			Setting Range	Set by	Data Type
①②	Bit device that turned ON at the completion of the processing. (①②+1 is also turned ON at error completion.)					Bit
	Device	Item	Contents/Setting Data	Setting Range	Set by	
	①②	Completion signal	Indicates the completion of the processing. ON: Completed OFF: Not completed	—	System	
①②+1	Error completion signal	Indicates whether the processing is normally completed or abnormally completed. ON: Error completion OFF: Normal completion	—			

## Caution

- At ①② (drive designation), only the ATA card drive (2) can be set.  
Note that when the Flash card is loaded, the SP.FREAD instruction cannot be used to perform read.  
The SRAM card, standard RAM or standard ROM drive cannot be set.
- For CSV setting, the data written are decimal values.

### **Example**

Character "A" (41H) → "65" is written.  
Handling range: -32768 to 32767

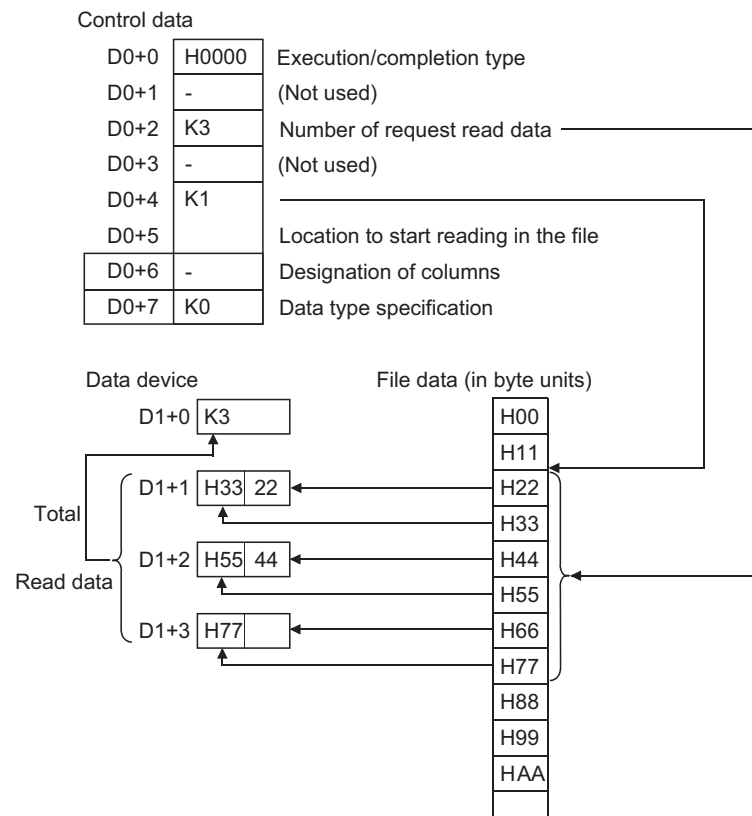
- For binary read, the word-specified file position setting range is 00000000H to 7FFFFFFFH.

## Function

- Data is read from the designated file.  
Set the execution/completion type in the control data to designate whether to read binary data without any conversion or to convert binary data into CSV format data before reading it.  
(The reading target is the ATA card only.)
- The execution completion bit device (①②) is automatically turned ON at the END processing after the completion of the instruction is detected. The bit device is turned OFF at the execution of the END instruction in the next scan.  
Use this bit device as the execution completion flag for the SP.FWRITE instruction.  
When this instruction is completed abnormally, the error completion device (①②+1) is turned ON/OFF in synchronization with the execution completion (①②) device. Use this device as the error completion flag for this instruction.  
SM721 is turned ON during the execution of the instruction.  
This instruction cannot be executed while SM721 is ON. (If an attempt is made, no processing is performed.)  
When an error is detected at the execution of the instruction (before SM721 is turned ON), the processing complete device (①①), the error completion device (①①+1), and SM721 are not turned ON.

- (3) Be sure to use word units to designate the number of request read data ( $\text{D0}+2$ ), file position ( $\text{D0}+4$  and  $\text{D0}+5$ ), and read data device size ( $\text{D1}$ ).

The following shows how the individual device data is read in binary data reading operation.



- (4) When reading binary data
- If the extension of the target file is omitted, ".BIN" is used as an extension.
  - When the designated file does not exist, an error occurs.
  - If the position specified is greater than the existing file size:
    - The High Performance model QCPU of which the first 5 digits of the serial number are "01111" or lower results in an error.
    - The High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU of which the first 5 digits of the serial number are '01112' or higher will perform reading at point 0 and will be completed normally.
- (5) When reading data after CSV format conversion
- The elements in CSV format file (cells for EXCEL) are read by each row. The numerical value and character strings are converted into binary data and stored in the device.
  - If the extension is omitted, ".CSV" is used as an extension.
  - When the designated file does not exist, an error occurs.
  - The elements designated by the number of request read data ( $\text{D0}+2$ ) are read from the beginning of the file.  
When the last data of the file is reached before the specified number of data are read:
    - The High Performance model QCPU of which the first 5 digits of the serial number are "01111" or lower results in an error.
    - The High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU whose the first 5 digits of the serial number are '01112' or higher reads the data up to the point where the reading is possible.

- (e) When the designated number of columns is 0, the data is read by ignoring the rows in CSV format file.

**Example** When data is read after CSV format conversion and the designated No. of columns is 0:

Data created by EXCEL

	A	B	C
1	Main / sub item		Measured value
2	Length	1	3
3	Temperature	-21	

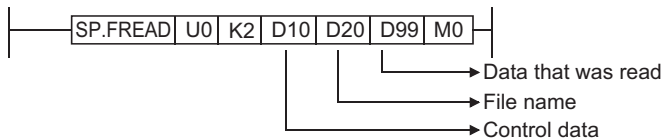


Data saved in the CSV format

Main / sub item	,	,	Measured value	CR	LF	
Length	,	1	,	3	CR	LF
Temperature	,	-21	,		CR	LF



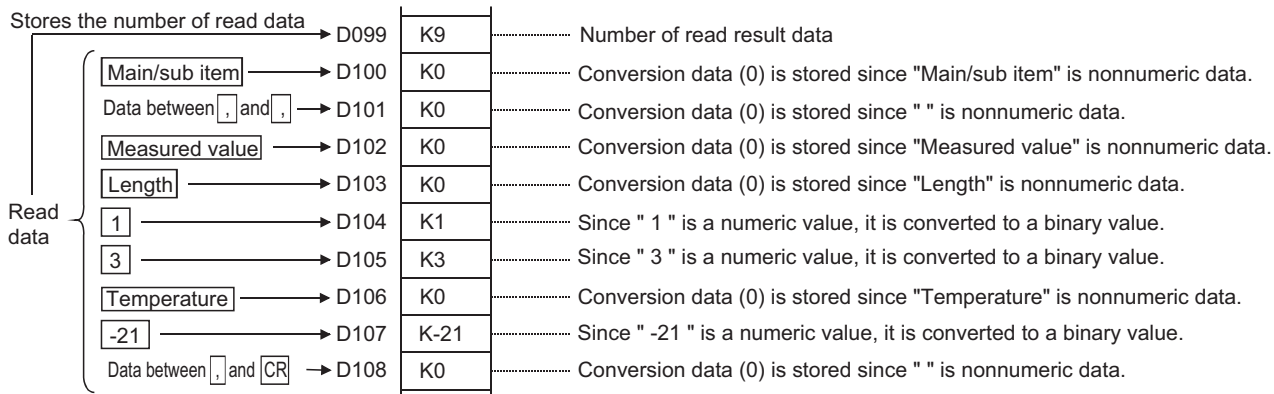
Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Number of unused read data
D12	K9	Request
D13	-	Not used
D14	K0	
D15	K0	
D16	K0	Designation of the number of columns
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCDE"
D22	H0045	

Loaded data



If the number of columns varies in each row, the data is also read by ignoring the rows.

**POINT**

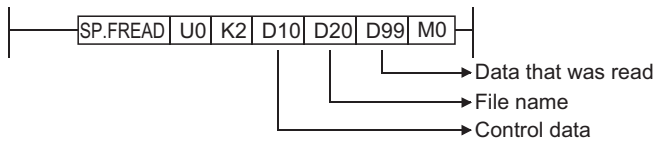
Such file cannot be created using EXCEL. This happens when CSV file is modified by a user.

**Example** If the number of columns varies in each row when the data is read:

Main / sub item	,	,	Measured value	,	Excess	CR	LF
Length	CR	LF					
Temperature	,	-21	,	CR	LF		



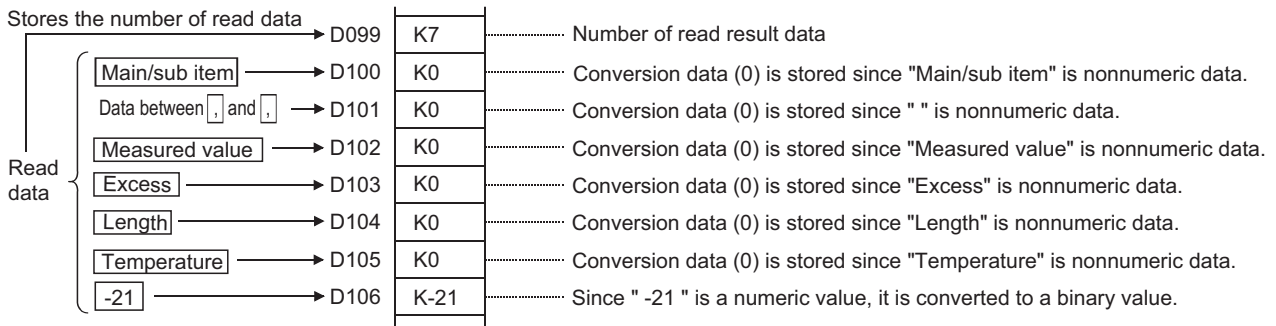
Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Not used
D12	K7	Number of request read data
D13	-	Not used
D14	K0	
D15	K0	
D16	K0	Designation of the number of columns
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCD"
D22	H0000	

Loaded data



- (f) When data is read after CSV format conversion and the designated number of columns is other than 0, the data is read as the table with designated number of columns in CSV format file. The elements outside of the designated columns are ignored.

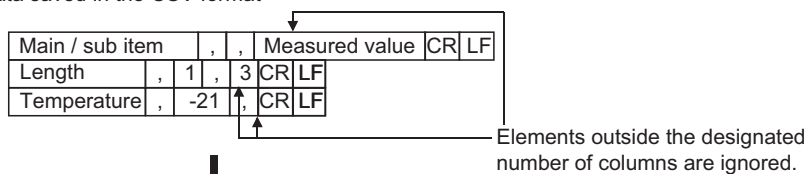
**Example** When data is read after CSV format conversion and the designated No. of columns is other than "0":

Data created by EXCEL

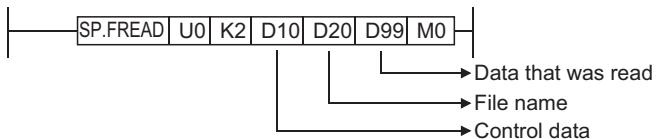
	A	B	C
1	Main / sub item		Measured value
2	Length	1	3
3	Temperature	-21	



Data saved in the CSV format



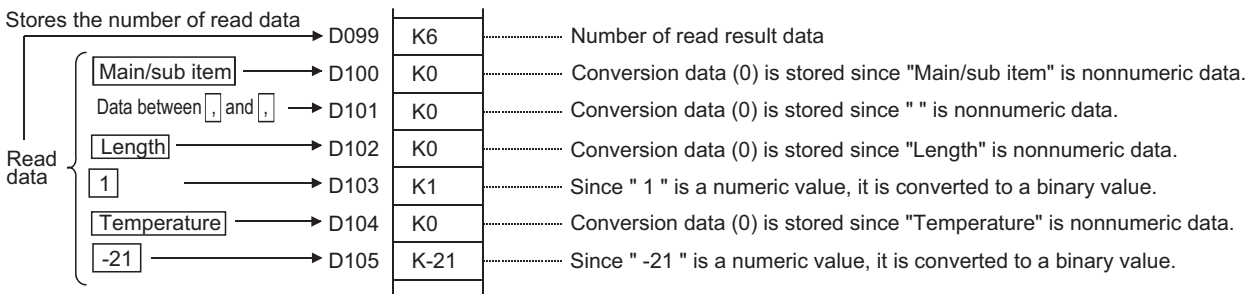
Data to be read into devices



Control data

D10	H0100	Execution/completion type
D11	-	Not used
D12	K6	Number of request read data
D13	-	Not used
D14	K0	
D15	K0	
D16	K2	Designation of the number of columns
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCD"
D22	H0000	

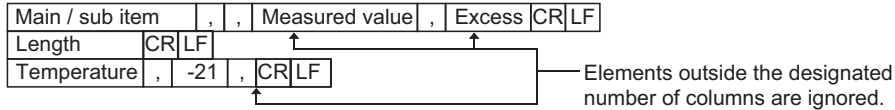
Loaded data



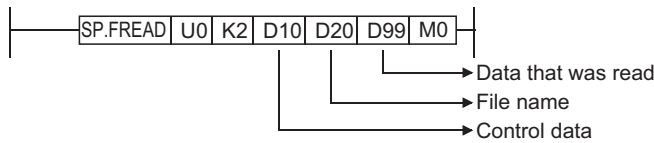
If the number of columns varies in each row, the elements outside of the designated columns are ignored and "0" is added to the places where elements do not exist.

**Example**

If the number of columns varies in each row when the data is read:

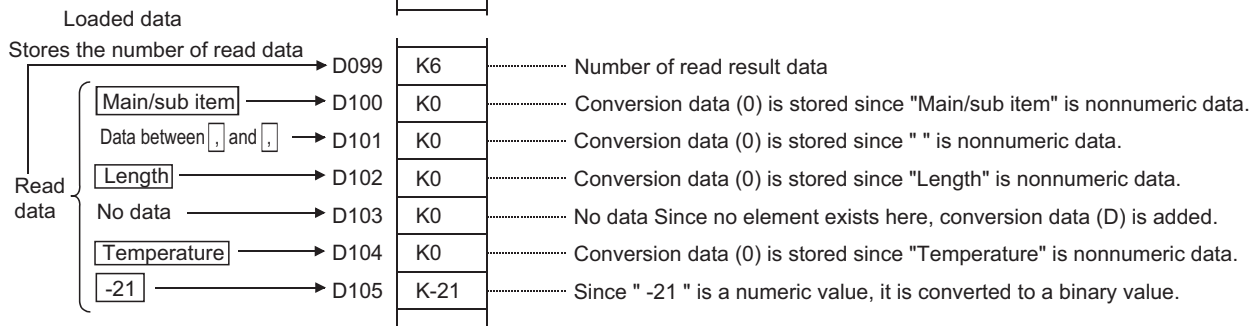


Data to be read into devices



Control data

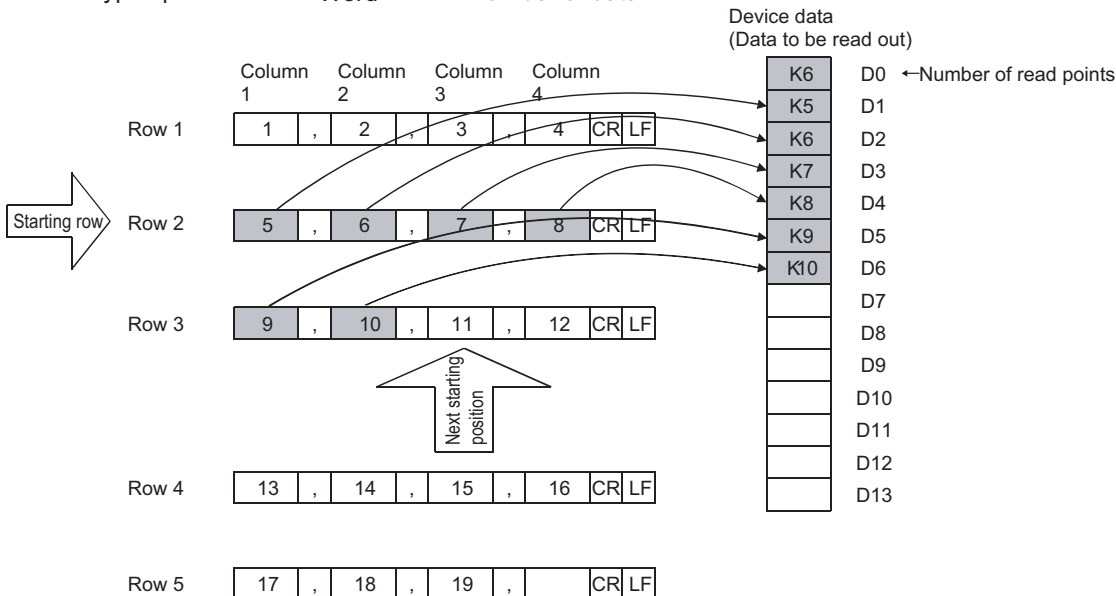
D10	H0100	Execution/completion type
D11	-	Not used
D12	K6	Number of request read data
D13	-	Not used
D14	K0	
D15	K0	
D16	K2	Designation of the number of columns
D17	K0	Data type specification
D20	H4241	File name
D21	H4443	"ABCD"
D22	H0000	



(g) With the High Performance model QCPU/Process CPU/Redundant CPU/Universal model QCPU whose first 5 digits of the serial number are "01112" or later, it is possible to divide read operation into multiple times.

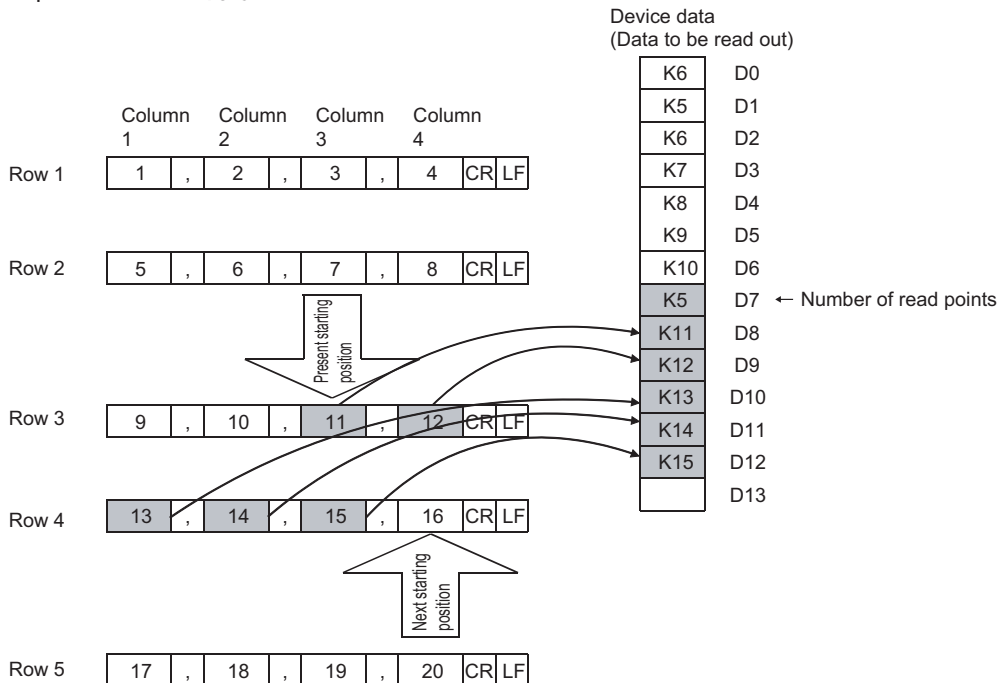
[Specify the row desired to start read.]

Execution type = CSV format Starting row number = 2H  
 Column designation = 4H Read head device = D0  
 Data type specification = Word Number of data = 6H



[In the continuation mode, read continues from the end of the previous read position.]

Execution type = CSV format Starting row number = FFFFFFFH (Continuation mode)  
 Column designation = 4H Read head device = D7  
 Data type specification = Word Number of data = 5H



- When read is performed in the continuation mode, the previous addition cannot be made normally if the "execution type", "column designation" and "Data type specification" settings differ from those at the previous time.
- The previous addition cannot be made normally if the SP.FREAD instruction or SP.FWRITE instruction with another setting is executed while data is being read continuously in the continuation mode.



- (h) When data is read after CSV format conversion, the numerical values that are out of range or the elements other than numerical values in the object CSV format file are converted into 0H.
- (i) When data is read after CSV format conversion, numerical values are read and converted as follows:

Numerical Values in CSV Format		-32768 to -1	0 to 32767	32768 to 65535
Word device	Without a sign	32768 to 65535	0 to 32767	32768 to 65535
	With a sign	-32768 to -1	0 to 32767	-32768 to -1

- (j) Do not execute this instruction in an interrupt program.  
(Otherwise, a malfunction may result.)

## Operation Error

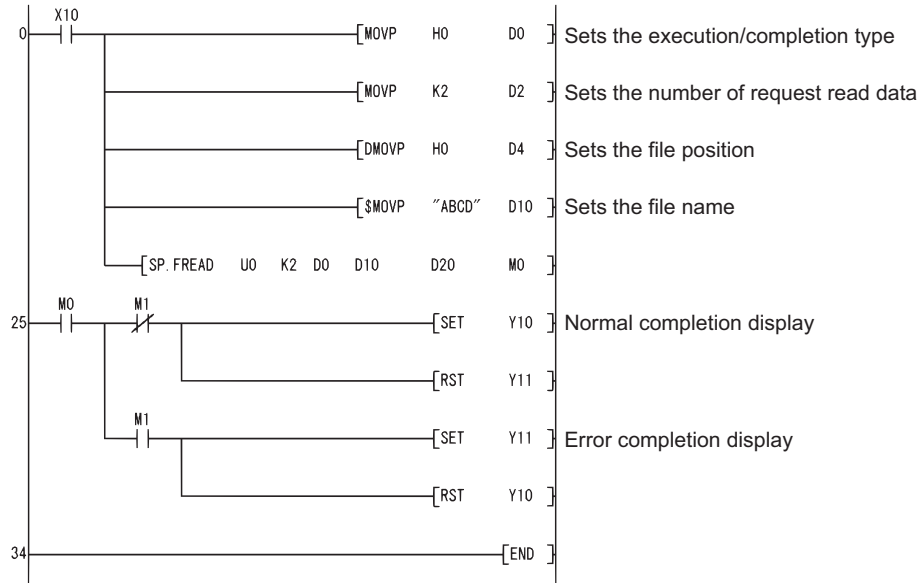
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- Drive specified by drive designation device ( $\textcircled{S0}$ ) contains the medium other than the ATA card. (Error code: 4100)
  - Values designated in control data ( $\textcircled{D0}$ ) and the subsequent devices are out of the setting range. (Excluding  $\textcircled{D0}+2$ ) (Error code: 4100)
  - Value designated by number of data blocks to be read ( $\textcircled{D0}+2$ ) is out of the setting range. (Error code: 4101)
  - Invalid device is designated. (Error code: 4004)
  - File name designated by file name character string ( $\textcircled{S1}$ ) or the subsequent devices does not exist in the designated drive. (Error code: 2410)
  - Size of read data exceeds the size of reading device. (Error code: 4101)
  - When binary data is read, the number of data in the file is less than the size designated by the number of request read data ( $\textcircled{D0}+2$ ).  
(High Performance model QCPU of which the first 5 digits of the serial number are '01111' or lower) (Error code: 4100)
  - Access error occurred in the ATA card. (Error code: 4100)
  - The device specified by  $\textcircled{D0}$  or  $\textcircled{D2}$  exceeds the range of the corresponding device.  
(For the Universal model QCPU only.) (Error code: 4101)

## Program Example

(1) The following program reads 4 bytes of binary data from the beginning of file "ABCD.BIN" in the memory card inserted to drive 2 when X10 is turned ON.

- Assume that 8 points from (D0) are reserved for the control data devices.
- Assume that 100 bytes from D20 are reserved for the reading devices.

[Ladder Mode]



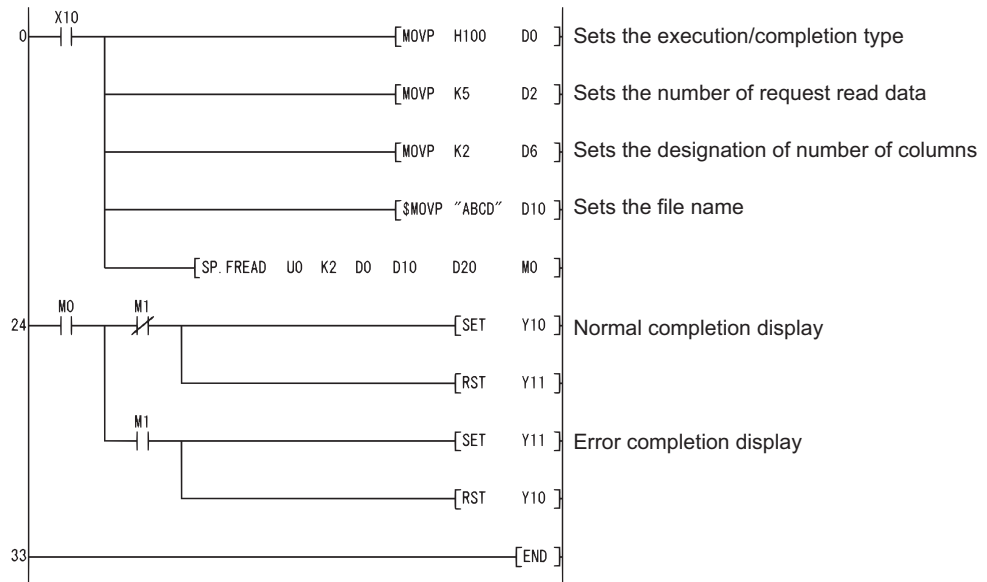
[List Mode]

Step	Instruction	Device
0	LD	X10
1	MOV P	H0 D0
3	MOV P	K2 D2
5	DMOV P	H0 D4
8	SMOV P	"ABCD" D10
13	SP.FREAD	U0 K2 D0 D10 D20 M0
25	LD	M0
26	MPS	
27	ANI	M1
28	SET	Y10
29	RST	Y11
30	MPP	
31	AND	M1
32	SET	Y11
33	RST	Y10
34	END	

(2) The following program reads file "ABCD.CSV" in the PC card inserted to slot 0 as two-column table data in CSV format when X10 is turned ON.

- Assume that 8 points from (D0) are reserved for the control data devices.
- Assume that 100 bytes from D20 are reserved for the reading devices.
- Assume that the target CSV format file contains numerical values only.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X10
1	MOV P	H100 D0
3	MOV P	K5 D2
5	MOV P	K2 D6
7	\$MOV P	"ABCD" D10
12	SP.FREAD	U0 K2 D0 D10 D20 M0
24	LD	M0
25	MPS	
26	ANI	M1
27	SET	Y10
28	RST	Y11
29	MPP	
30	AND	M1
31	SET	Y11
32	RST	Y10
33	END	

# 7.18.14 Writing Data to Standard ROM (SP.DEVST)



n1: Write offset of the device data storage file (specified in units of 16-bit words) (BIN 32-bit)

S: Head device number written to the standard ROM (device name)

n2: The number of write points (BIN 16-bit)

D: D +0 : FCompletion device (bit)

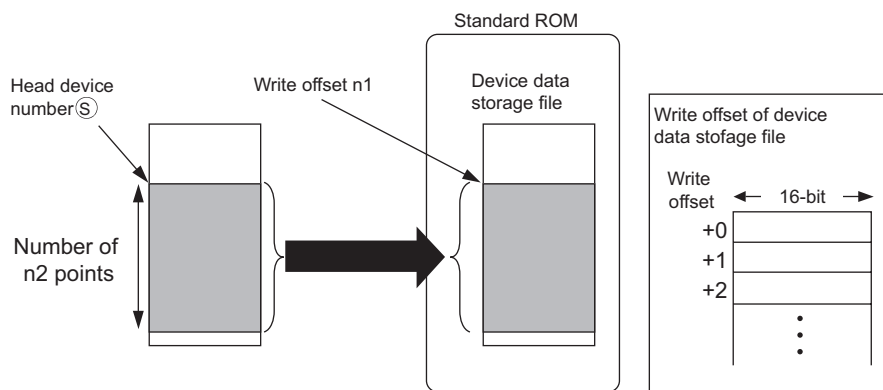
D +1 : FError completion device (bit)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K,H	Other
	Bit	Word		Bit	Word				
n1	—	○	○			—		○	—
S	—	○	○			—		—	—
n2	—	○	○			—		○	—
D	△*1	—	△*1			—		—	—

\*1: Devices assigned as local devices can not be used.

## ★ Function

- Writes device data for the number of points specified at n2 of the device S to the write offset, which is specified for n1, of the device data storage file in the standard ROM. n1 is the offset from the head of device data storage file and specified by word offset (in units of 16-bit words).



- Since the device data write position completion device (D +0) in the standard ROM automatically turns ON at execution of the END instruction, which detects the completion of this instruction, and turns OFF with the END instruction of next scan, it is used as an execution completion flag of this instruction.
- When this instruction is completed in error, the error completion device (D +1) turns ON/OFF at the same timing with the completion device (D +0). This device is used as an error completion flag of this instruction.

- (4) SM721 turns ON during execution of this instruction.  
When SM721 has already turned ON, this instruction can not be executed. (If executed, no processing is performed.)
- (5) When an error is detected at execution of this instruction, the completion device ( $\text{D} + 0$ ), error completion device ( $\text{D} + 1$ ) and SM721 do not turn ON.

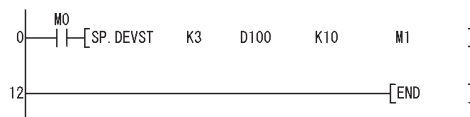
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - The write offset specified at n1 is out of the device data storage file range. (Error code: 4100)
  - The number of n2 points from the write offset specified at n1 is out of the device data storage file range. (Error code: 4100)
  - The range for the number of n2 points from the device  $\text{S}$  exceeds the corresponding device. (Error code: 4141)
  - The device data storage file is not set at "PLC file" of PLC parameter on GX Developer. (Error code: 2410)
  - The device specified by  $\text{D}$  exceeds the range of the corresponding device. (Error code: 4101)

## Program Example

- (1) The program which writes the ten points of data from D100 to the device data storage file in the standard ROM when M0 turns ON.

[Ladder Mode]



[List Mode]

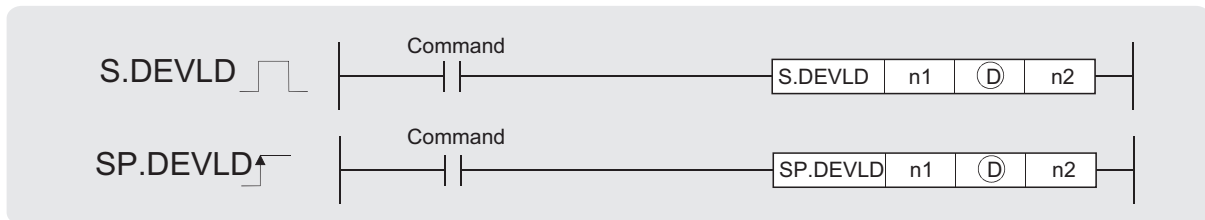
Step	Instruction	Device
0	LD	M0
1	SP.DEVST	K3 D100 K10 M1
12	END	

## Caution

- (1) The value written to the standard ROM is the value at execution of this instruction.
- (2) The standard ROM write count index (SD687 and SD688) is increased by the execution of the SP.DEVST instruction. If the standard ROM write count index exceeds hundred thousand times, FLASH ROM ERROR (error code: 1610) occurs.
- (3) To prevent the number of ROM writes from increasing due to executing instruction carelessly, set the specification of writing to standard ROM instruction count (SD695) to restrict the number of writes a day.

Exceeding the number of writes (the default values are 36 times.) set causes OPERATION ERROR (error code: 4113).

# 7.18.15 Read Data from Standard ROM (S(P).DEVLD)



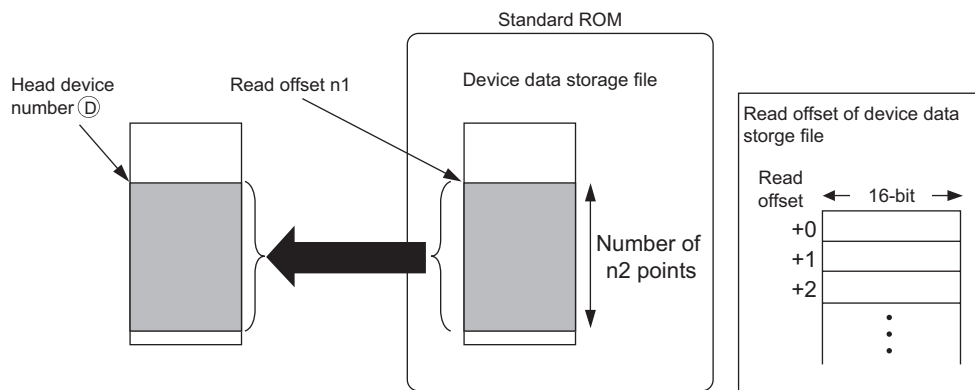
n1 : Read offset of the device data storage file (specified in units of 16-bit words) (BIN 32-bit)  
 Ⓣ : Head device number read from the standard ROM (device name)  
 n2 : The number of reading points (BIN 16-bit)

Setting Data	Internal Devices		R, ZR	J20		U20	Zn	Constants E	Other
	Bit	Word		Bit	Word				
n1	—	○				—		○	—
Ⓣ	—	○				—		—	—
n2	—	○				—		○	—

## ★ Function

- Reads device data for the number of points specified at n2 from the read offset, which is specified for n1, of the device data storage file in the standard ROM, and stores the data to the device specified for Ⓣ.

n1 is the offset from the head of device data storage file and specified by word offset (in units of 16-bit words).



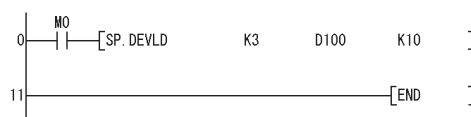
## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The address specified at n1 is out of the standard ROM range. (Error code: 4100)
  - The number of n2 points from the address specified at n1 is out of the standard ROM range. (Error code: 4100)
  - The range for the number of n2 points from the device  $\text{\textcircled{D}}$  exceeds the corresponding device. (Error code: 4101)
  - The device data storage file is not set at "PLC file" of PLC parameter on GX Developer. (Error code: 2410)

## Program Example

- (1) The program which reads the ten points of data from D100 to the device data storage file in the standard ROM when M0 turns ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	SP.DEVLD	K3
11	END	D100 K10

# 7.18.16 Load Program from Memory Card (PLOADP)



Ⓢ : Drive No. storing the program to be loaded, character string data of the file name, or head number of the devices storing the character string data (BIN 16 bits) \*1

Ⓣ : Device that turns ON for 1 scan by the instruction completion (bits)

Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○				—		○	—
Ⓣ	△*2	—				—		—	—

\*1: Designated as "<Drive No.>:<File Name>". Example) 1:MAIN

\*2: Local devices cannot be used.

## ★ Function

- (1) The program stored in the memory card or standard ROM is transferred to the program memory (drive 0).  
 If the transferred program is not registered to the program setting of the PLC parameter dialog box, its program setting in the CPU module is set to the standby type.  
 At this time, the program setting of the PLC parameter dialog box does not change.  
 (To transfer a program with the PLOADP instruction, a continuous free space is required in the program memory.)
- (2) The program added using the PLOADP instruction is assigned the lowest number among the unused program Nos.  
 (To assign a program number manually, store the program number to be assigned in SD720.)  
 The following example assumes that "MAIN6" is added by the PLOADP instruction.
  - (a) When the program Nos. have been set consecutively, the new program is added at the end of the preset program Nos.  
 When programs No. 1 to 5 have been set, the new program is added as program No. 6.

Program No.	Program name
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5

Adds "MAIN6" by the PLOADP instruction.

Program No.	Program name
1	MAIN1
2	MAIN2
3	MAIN3
4	MAIN4
5	MAIN5
6	MAIN6

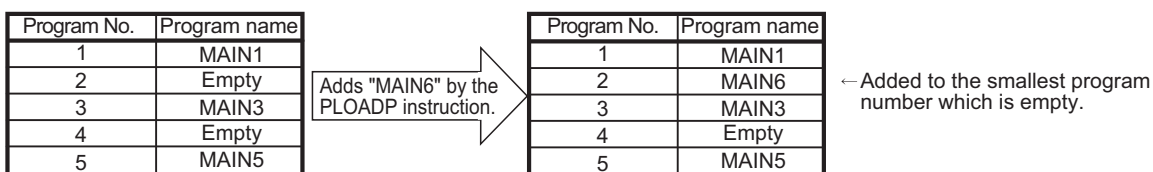
← Added at the end.



- (b) When there are multiple open program Nos., the program designated by the PLOADP instruction is added to the lowest number among them to be added.

(The open program Nos. are made when programs are deleted by the PUNLOADP instruction.)

When programs No. 2 and 4 are open, the new program is added as program No. 2.



- (3) Drive Nos. 1, 2, and 4 can be specified. (Drive 3 cannot be specified.)
- Drive 1: Memory card (RAM)
  - Drive 2: Memory card (ROM)
  - Drive 4: Standard ROM
- (4) An extension (.QPG) need not be specified for the file name.
- (5) The bit device specified by  $\textcircled{D}$  is turned ON during the END processing of the scan where this instruction is completed. The bit device is turned OFF at the next END processing.
- (6) The PLOADP, PUNLOADP and PSWAPP instructions cannot be executed simultaneously. If two or more of the above instructions are executed simultaneously, the instruction executed later will not be executed.  
When using the above instructions, provide interlocks manually to avoid simultaneous execution.
- (7) Do not execute this instruction in an interrupt program.  
(Otherwise, a malfunction may result.)
- (8) To execute the program that was transferred to the program memory with the PLOADP instruction, execute the scan execution type with the PSCAN instruction (See Section 7.17.3).
- (9) The PLC file settings of the loaded program are set as follows:
- (a) File usage for each program  
All the usage of file register, device initial value, comment, and local device of the program transferred by this instruction are set as "Use PLC file setting". However, an error will be returned if both of the conditions below are met when the program is transferred using this instruction.
- Setting is made so that local devices are used in the PLC file setting.
  - The number of programs in the program memory exceeds the number of programs set at the parameters.
- To use local devices in the program transferred by this instruction, register a dummy program file in the parameter, delete the dummy file with the PUNLOADP instruction, and then load the program with the PLOADP instruction.
- (b) I/O refresh setting  
Nothing is set for both input and output for the I/O refresh setting of the program transferred by this instruction.

- (10) The "PLOADP instruction" and "Write during RUN" processing cannot be executed simultaneously.
- (a) When a write during RUN request is given during processing of the PLOADP instruction, write during RUN is delayed.  
Write during RUN is started after the processing of the PLOADP instruction is completed.
  - (b) When the PLOADP instruction is executed during write during RUN, the processing of the PLOADP instruction is delayed.  
The processing of the PLOADP instruction is started after completion of write during RUN.

## Operation Error

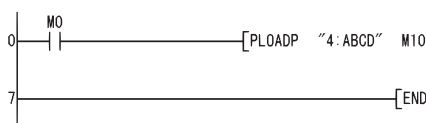
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- File name does not exist at the drive number designated by Ⓢ. (Error code: 2410)
  - The drive No. designated by Ⓢ is invalid. (Error code: 4100)
  - There is not enough memory to load the specified program in drive 0. (Error code: 2413)
  - The number of files registered in the program memory is as much number as the one indicated in the table below. (Error code: 4101)
  - The program No. stored in SD720 is already used, or larger than the largest program number shown in the table below. (Error code: 4101)
  - The program file which has the same name as the program file to be loaded already exists. (Error code: 2410)
  - The file size of the local devices cannot be reserved. (Error code: 2401)

CPU Model Name	Program Memory (No. of Files)	Largest Program No.
Q02 (H) CPU	28	28
Q06HCPU	60	60
Q12HCPU	124	124
Q25HCPU	124	124
Q12PHCPU	124	124
Q25PHCPU	124	124

## Program Example

- (1) The following program transfers "ABCD.QPG" stored in drive 4 to drive 0 and places the program in standby status when M0 is turned ON.

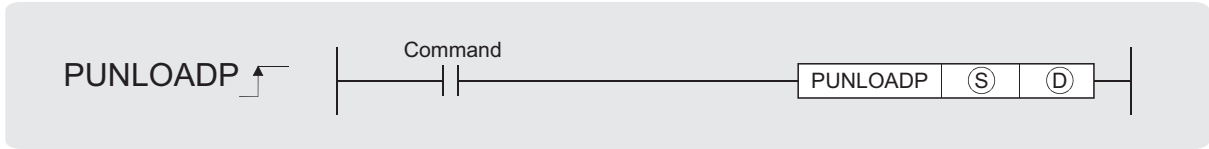
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	M0
1	PLOADP	"4:ABCD" M10
7	END	

# 7.18.17 Unload Program from Program Memory (PUNLOADP)



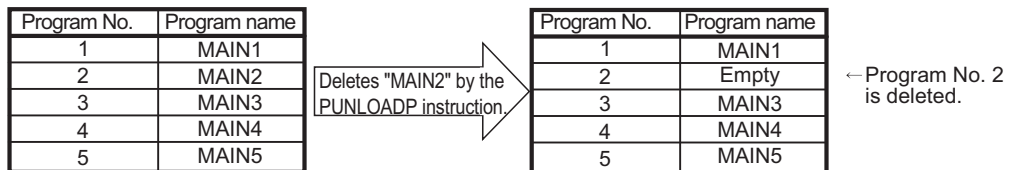
- (S) : Character string data of the program file name to be unloaded, or head number of the devices storing the character string data (BIN 16 bits)
- (D) : Device turned ON for 1 scan on completion of the instruction (bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
(S)	—	○						○	—
(D)	△*1	—						—	—

\*1: Local devices cannot be used.

## ★ Function

- The standby program stored in the program memory (drive 0) is deleted from the program memory.  
(The program set as the "scan execution type" with the PSCAN instruction or the program set as the "low speed execution type" with the PLOW instruction cannot be deleted.)
- The program No. deleted by the PUNLOADP instruction is made "Empty".  
When programs No. 1 to 5 have been set in the program setting of the PLC parameter dialog box, deleting program No. 2 with this instruction makes program No. 2 open.



- An extension (.QPG) need not be specified for the file name.
- The bit device specified by (D) is turned ON during the END processing of the scan where this instruction is completed. The bit device is turned OFF at the next END processing.
- The PLOADP, PUNLOADP and PSWAPP instructions cannot be executed simultaneously. If two or more of the above instructions are executed simultaneously, the instruction executed later will not be executed.  
When using the above instructions, provide interlocks manually to avoid simultaneous execution.

7.18 Other instructions  
7.18.17 Unload Program from Program Memory (PUNLOADP)

- (6) When the Programmable Controller is powered OFF, then ON or the CPU module is reset after execution of the PUNLOADP instruction, the following operation is performed.
- (a) When boot setting has been made in the PLC parameter dialog box, the program where the boot setting has been made is transferred to the program memory.  
When the program deleted by the PUNLOADP instruction is not to be executed, delete the corresponding program name from the boot setting and program setting of the PLC parameter dialog box.
  - (b) When boot setting has not been made in the PLC parameter dialog box, "FILE SET ERROR (error code: 2400)" occurs.
    - 1) When the program deleted by the PUNLOADP instruction is not to be executed, delete the corresponding program name from the program setting of the PLC parameter dialog box.
    - 2) When the program deleted by the PUNLOADP instruction is to be executed again, write the corresponding program to the CPU module.
- (7) Do not execute this instruction in an interrupt program.  
(Otherwise, a malfunction may result.)
- (8) The program to be deleted from the program memory by this instruction should be set to the "standby execution type" with the PSTOP instruction beforehand. (See Section 7.17.1)
- (9) The "PUNLOADP instruction" and "write during RUN" processing cannot be executed simultaneously.
- (a) When a write during RUN request is given during processing of the PUNLOADP instruction, write during RUN is delayed.  
Write during RUN is started after the processing of the PUNLOADP instruction is completed.
  - (b) When the PUNLOADP instruction is executed during write during RUN, the processing of the PUNLOADP instruction is delayed.  
The processing of the PUNLOADP instruction is started after completion of write during RUN.



## Operation Error

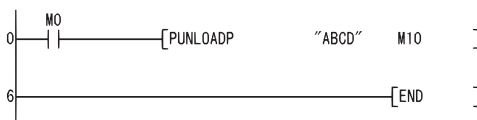
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The file name designated by ⑤ does not exist. (Error code: 2410)
  - The program designated by ⑤ is not in standby status or is being executed. (Error code: 4101)



## Program Example

- (1) The following program deletes "ABCD.QPG" stored in drive 0 from the memory when M0 turns from OFF to ON.

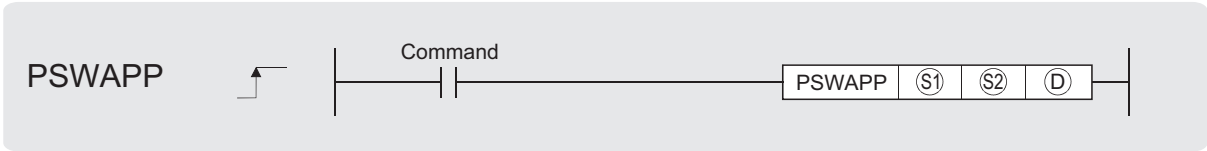
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	
1	PUNLOADP	M0
6	END	"ABCD" M10

# 7.18.18 Load + Unload (PSWAPP)



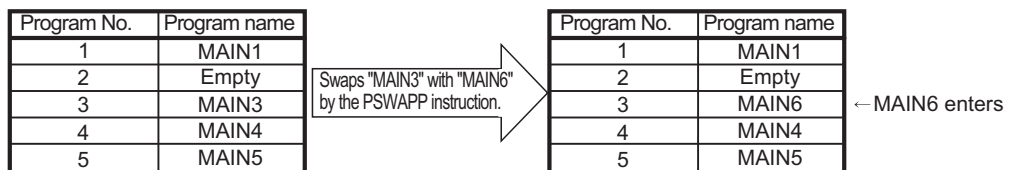
- Ⓢ1 : Character string data of the file name of the program to be unloaded, or head number of the devices storing the character string data (BIN 16 bits)
- Ⓢ2 : Drive No. storing the program to be loaded, character string data of the file name, or head number of the devices storing the character string data (BIN 16 bits) \*1
- ⓓ : Device turned ON for 1 scan on completion of the instruction (bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants \$	Other
	Bit	Word		Bit	Word				
Ⓢ1	—	○				—		○	—
Ⓢ2	—	○				—		○	—
ⓓ	△*2	—				—		—	—

\*1: Designated as "<Drive No.>:<File Name>". Example) 1:MAIN  
 \*2: Local devices cannot be used.

## ★ Function

- (1) The standby type program stored in the program memory (drive 0) designated by Ⓢ1 is deleted from the program memory, and at the same time, the program stored in the memory card or standard ROM designated by Ⓢ2 is transferred to the program memory and placed in standby status.  
 (When the program is transferred to the program memory, the program must have a continuous free space.)  
 The program set as the "scan execution type" with the PSCAN instruction or the program set as the "low speed execution type" with the PLOW instruction cannot be deleted.
- (2) The program to be transferred to the program memory by the PSWAPP instruction will have the program No. of the program to be deleted from the program memory.  
 (If there is an open program No. before the program to be deleted from the program memory, the program to be transferred to the program memory will not have the open program No.)  
 When program No. 2 is "Empty", the program transferred to the program memory is registered as program No. 3 by the program swapping of program No. 3 with this instruction.



7.18 Other instructions  
7.18.18 Load + Unload (PSWAPP)

- (3) Drive Nos. 1, 2, and 4 can be specified. (Drive 3 cannot be specified.)
  - Drive 1: Memory card (RAM)
  - Drive 2: Memory card (ROM)
  - Drive 4: Standard ROM
- (4) An extension (.QPG) need not be specified for the file name.
- (5) The bit device specified by Ⓓ is turned ON during the END processing of the scan where this instruction is completed. The bit device is turned OFF at the next END processing.
- (6) The PLOADP, PUNLOADP and PSWAPP instructions cannot be executed simultaneously. If two or more of the above instructions are executed simultaneously, the instruction executed later will not be executed.  
When using the above instructions, provide interlocks manually to avoid simultaneous execution.
- (7) When the Programmable Controller is powered OFF, then ON or the CPU module is reset after execution of the PSWAPP instruction, the following operation is performed.
  - (a) When boot setting has been made in the PLC parameter dialog box, the program where the boot setting has been made is transferred to the program memory.  
When the program replaced by the PSWAPP instruction is to be executed, change the boot setting and program setting of the PLC parameter dialog box for the corresponding program name.
  - (b) When boot setting has not been made in the PLC parameter dialog box, "FILE SET ERROR (error code: 2400)" occurs.
    - 1) When the program replaced by the PSWAPP instruction is to be executed, change the program setting of the PLC parameter dialog box for the corresponding program name.
    - 2) To execute the program set in the program setting of the PLC parameter dialog box, write the corresponding program to the CPU module again.
- (8) Do not execute this instruction in an interrupt program.  
(Execution of this instruction in an interrupt program can cause a malfunction.)
- (9) The PLC file settings of the program on which the PSWAPP instruction has been conducted are set as follows:
  - (a) File usage for each program  
All the usage of file register, device initial value, comment, and local device of the program after the execution of the PSWAPP instruction are set as "Use PLC file setting".
  - (b) I/O refresh setting  
Nothing is set for both input and output for the I/O refresh setting of the program after the PSWAPP instruction has been executed.
- (10) The "PSWAPP instruction" and "write during RUN" processing cannot be executed simultaneously.
  - (a) When a write during RUN request is given during processing of the PSWAPP instruction, write during RUN is delayed.  
Write during RUN is started after the processing of the PSWAPP instruction is completed.
  - (b) When the PSWAPP instruction is executed during write during RUN, the processing of the PSWAPP instruction is delayed.  
The processing of the PSWAPP instruction is started after completion of write during RUN.

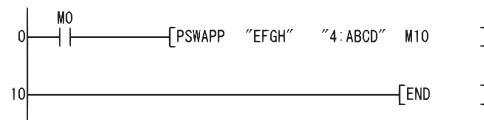
## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The drive No. or the file name designated by ① or ② does not exist. (Error code: 2410)
  - The drive No. designated by ① is invalid. (Error code: 4100)
  - There is not enough memory to load the specified program in drive 0. (Error code: 2413)
  - The program designated by ③ is not in standby status or is being executed. (Error code: 4101)

## Program Example

- (1) The following program deletes "EFGH.QPG" stored in drive 0 from the memory, transfers "ABCD.QPG" stored in drive 4 to drive 0, and places the program in standby status when M0 is turned from OFF to ON.

[Ladder Mode]

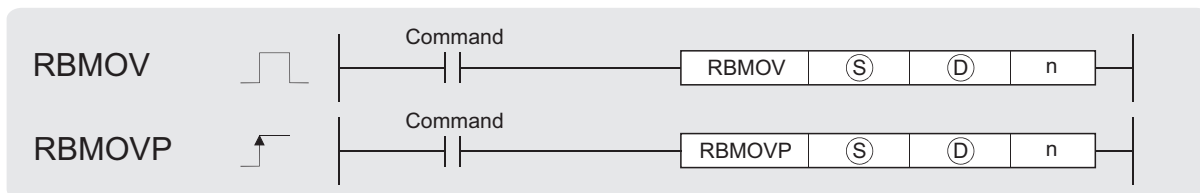


[List Mode]

Step	Instruction	Device
0	LD	M0
1	PSWAPP	"EFGH" "4:ABCD" M10
10	END	

# 7.18.19 High-speed Block Transfer of File Register (RBMOV(P))

Basic
High performance
Process
Redundant
Universal

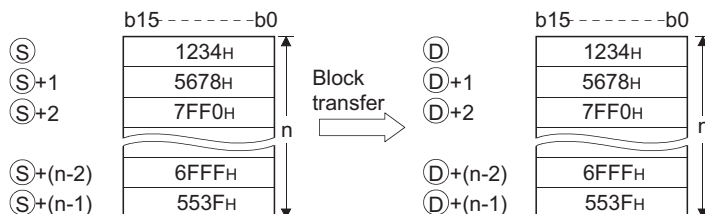


Ⓢ : Head number of the devices where the data to be transferred is stored (BIN 16 bits)  
 Ⓣ : Head number of the devices of transfer destination (BIN 16 bits)  
 n : Number of data to be transferred (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ				○			—		—
Ⓣ				○			—		—
n				○			○		—

## ★ Function

- (1) Transfers in batch 16-bit data of n points from the device designated by Ⓢ to location n points from the device designated by Ⓣ.



- (2) The transfer is available even if there is an overlap between the source and destination devices.

For the transmission to the smaller number of device, the data is transferred from Ⓢ. For the transmission to the larger number of device, the data is transferred from Ⓢ+(n-1).

However, as shown in the example below, when transferring data from R to ZR, or from ZR to R, the range to be transferred (source) and the range of destination must not overlap.

- ZR transfer range ((specified head No. of ZR) to (specified head No. of ZR + the number of transfers -1))

R transfer range ((specified head No. of R + file register block No. × 32768) to (specified head No. of R + file register block No. × 32768 + the number of transfers -1))

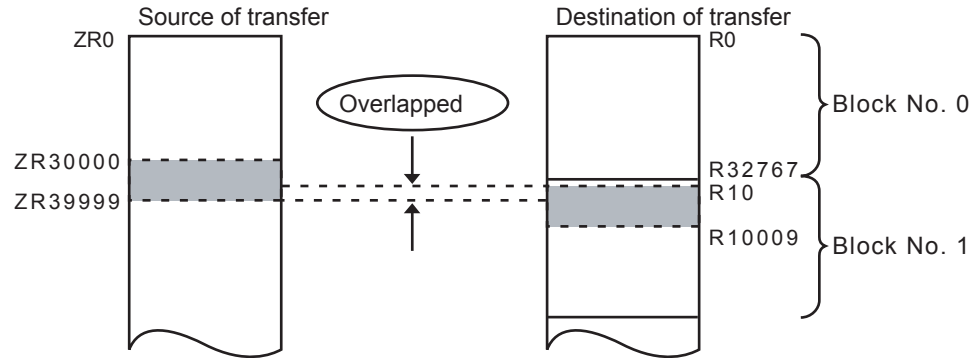


**Example**

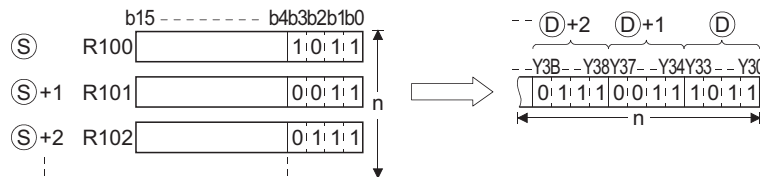
Transfer ranges of ZR and R overlap when transferring 10000 points of data from ZR30000 (source) to R10 (block No.1 of the destination).

- ZR transfer range → (30000) to (30000+10000-1) →(30000) to (39999)
- R transfer range → (10+(1×32768)) to (10+(1×32768)+10000-1)  
→ (32778) to (42777)

Therefore, the range 32778 to 39999 overlaps.



- (3) When  $\textcircled{S}$  is a word device and  $\textcircled{D}$  is a bit device, the number of bits designated by the bit device digit specification will be transferred. If K1Y30 has been designated by  $\textcircled{D}$ , the lower four bits of the word device designated by  $\textcircled{S}$  will be transferred.



**! Operation Error**

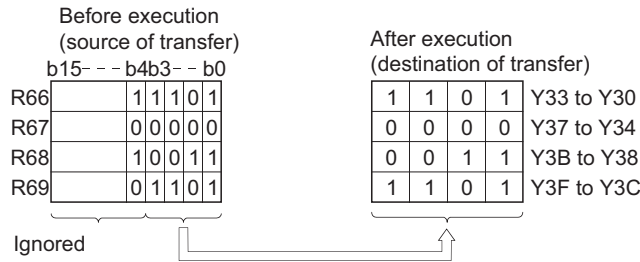
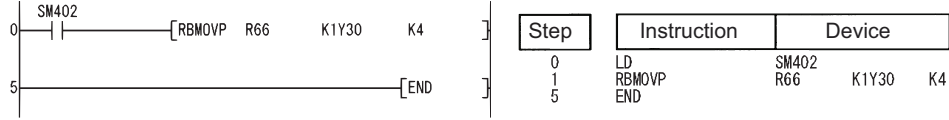
- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The device range of n points from  $\textcircled{S}$  or  $\textcircled{D}$  exceeds the corresponding device range. (Error code: 4101)
  - The file register is not specified for either  $\textcircled{S}$  or  $\textcircled{D}$ . (Error code: 4101)

# Program Example

- (1) The following program outputs the lower four bits of data in R66 to R69 to Y30 through Y3F in units of 4 points.

[Ladder Mode]

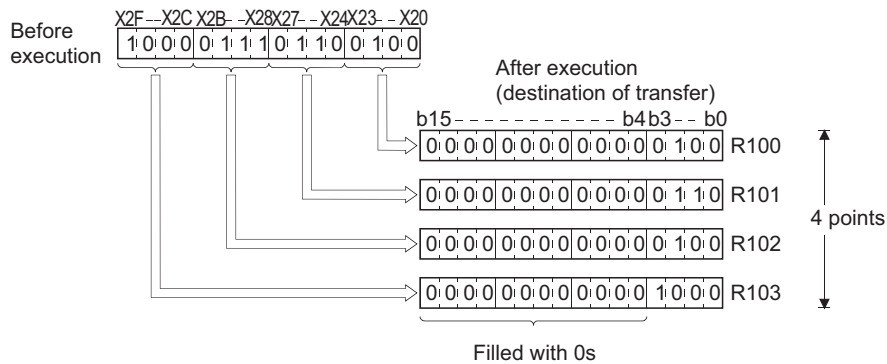
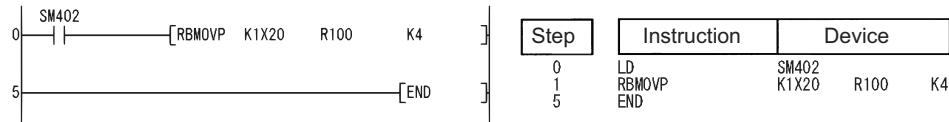
[List Mode]



- (2) The following program outputs the data in X20 to X2F to R100 to R103 in units of 4 points.

[Ladder Mode]

[List Mode]



## POINT

The RBMOV (P) instruction is useful to batch transfer a large quantity of file register data with the QnHCPU/QnPHCPU/QnPRHCPU.

For the QnUCPU, the processing speed of the RBMOV instruction is equivalent to that of the BMOV instruction.

The comparison of processing speed between the RBMOV and BMOV instructions is as follows:

(1) Transfer from file registers to internal devices/internal devices to file registers

CPU	Instruction	Target memory where file register is stored	1 word		1000 words		10000 words	
			Min.	Max.	Min.	Max.	Min.	Max.
QnHCPU QnPHCPU QnPRHCPU	RBMOV	Standard RAM	20.0 $\mu$ s		91.0 $\mu$ s		775.0 $\mu$ s	
		SRAM card	22.0 $\mu$ s		305.0 $\mu$ s		2900.0 $\mu$ s	
		Flash card *1	22.5 $\mu$ s		405.0 $\mu$ s		3950.0 $\mu$ s	
	BMOV	Standard RAM	7.5 $\mu$ s		76.2 $\mu$ s		720.0 $\mu$ s	
		SRAM card	8.0 $\mu$ s		384.0 $\mu$ s		3900.0 $\mu$ s	
		Flash card *1	8.0 $\mu$ s		418.0 $\mu$ s		4250.0 $\mu$ s	
QnCPU	RBMOV	Standard RAM	45.5 $\mu$ s		215.0 $\mu$ s		1850.0 $\mu$ s	
		SRAM card	49.5 $\mu$ s		540.0 $\mu$ s		5150.0 $\mu$ s	
		Flash card *1	49.5 $\mu$ s		540.0 $\mu$ s		5150.0 $\mu$ s	
	BMOV	Standard RAM	17.5 $\mu$ s		177.0 $\mu$ s		1700.0 $\mu$ s	
		SRAM card	18.0 $\mu$ s		500.0 $\mu$ s		5050.0 $\mu$ s	
		Flash card *1	18.0 $\mu$ s		572.0 $\mu$ s		5800.0 $\mu$ s	
Q00UCPU Q01UCPU	RBMOV	Standard RAM	12.2 $\mu$ s	34.9 $\mu$ s	121.5 $\mu$ s	145.1 $\mu$ s	1111.5 $\mu$ s	1135.1 $\mu$ s
		SRAM card*2	-	-	-	-	-	-
		Flash card *2	-	-	-	-	-	-
	BMOV	Standard RAM	7.3 $\mu$ s	13.8 $\mu$ s	116.5 $\mu$ s	124.2 $\mu$ s	1106.5 $\mu$ s	1114.2 $\mu$ s
		SRAM card*2	-	-	-	-	-	-
		Flash card *2	-	-	-	-	-	-
Q02UCPU	RBMOV	Standard RAM	9.4 $\mu$ s	31.3 $\mu$ s	118.5 $\mu$ s	141.3 $\mu$ s	1108.5 $\mu$ s	1131.3 $\mu$ s
		SRAM card	9.4 $\mu$ s	31.4 $\mu$ s	178.5 $\mu$ s	201.3 $\mu$ s	1708.5 $\mu$ s	1731.3 $\mu$ s
		Flash card *1	9.4 $\mu$ s	32.1 $\mu$ s	278.5 $\mu$ s	301.3 $\mu$ s	2708.5 $\mu$ s	2731.3 $\mu$ s
	BMOV	Standard RAM	5.0 $\mu$ s	11.6 $\mu$ s	114.5 $\mu$ s	122.3 $\mu$ s	1104.5 $\mu$ s	1112.3 $\mu$ s
		SRAM card	5.1 $\mu$ s	11.7 $\mu$ s	174.5 $\mu$ s	182.3 $\mu$ s	1704.5 $\mu$ s	1712.3 $\mu$ s
		Flash card *1	5.0 $\mu$ s	11.6 $\mu$ s	274.5 $\mu$ s	282.3 $\mu$ s	2704.5 $\mu$ s	2712.3 $\mu$ s
Q03UD(E)CPU	RBMOV	Standard RAM	11.3 $\mu$ s	16.8 $\mu$ s	120.7 $\mu$ s	127.1 $\mu$ s	1110.7 $\mu$ s	1117.1 $\mu$ s
		SRAM card	11.2 $\mu$ s	16.7 $\mu$ s	180.7 $\mu$ s	187.1 $\mu$ s	1710.7 $\mu$ s	1717.1 $\mu$ s
		Flash card *1	11.3 $\mu$ s	16.8 $\mu$ s	280.7 $\mu$ s	287.1 $\mu$ s	2710.7 $\mu$ s	2717.1 $\mu$ s
	BMOV	Standard RAM	4.8 $\mu$ s	6.6 $\mu$ s	114.7 $\mu$ s	117.1 $\mu$ s	1104.7 $\mu$ s	1107.1 $\mu$ s
		SRAM card	4.8 $\mu$ s	6.6 $\mu$ s	147.7 $\mu$ s	177.1 $\mu$ s	1704.7 $\mu$ s	1707.1 $\mu$ s
		Flash card *1	4.8 $\mu$ s	6.5 $\mu$ s	274.7 $\mu$ s	277.1 $\mu$ s	2704.7 $\mu$ s	2707.1 $\mu$ s
Q04UD(E)HCPU Q06UDE(H)CPU Q10UDE(H)CPU Q13UDE(H)CPU Q20UDE(H)CPU Q26UDE(H)CPU	RBMOV	Standard RAM	9.2 $\mu$ s	15.1 $\mu$ s	61.0 $\mu$ s	68.6 $\mu$ s	531.0 $\mu$ s	538.6 $\mu$ s
		SRAM card	9.4 $\mu$ s	15.6 $\mu$ s	165.0 $\mu$ s	172.6 $\mu$ s	1576.0 $\mu$ s	1583.6 $\mu$ s
		Flash card *1	9.4 $\mu$ s	15.7 $\mu$ s	260.0 $\mu$ s	267.6 $\mu$ s	2526.0 $\mu$ s	2533.6 $\mu$ s
	BMOV	Standard RAM	4.1 $\mu$ s	5.6 $\mu$ s	56.0 $\mu$ s	58.6 $\mu$ s	526.0 $\mu$ s	528.6 $\mu$ s
		SRAM card	4.5 $\mu$ s	6.1 $\mu$ s	160.0 $\mu$ s	162.6 $\mu$ s	1571.0 $\mu$ s	1573.6 $\mu$ s
		Flash card *1	4.3 $\mu$ s	6.2 $\mu$ s	255.0 $\mu$ s	257.6 $\mu$ s	2521.0 $\mu$ s	2523.6 $\mu$ s

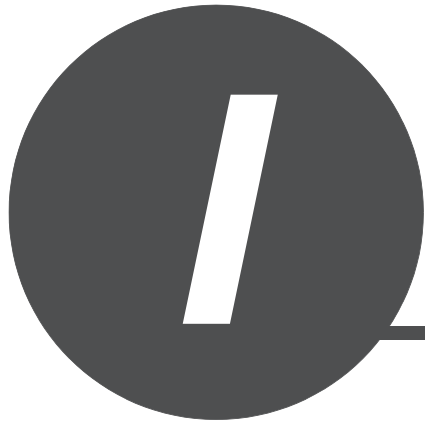
\*1 : When file registers are stored in the Flash card, no processing is performed for transfer from internal devices to file registers.

\*2 : Unusable for the Q00UCPU and Q01UCPU.

## (2) Transfer from file registers to file registers

CPU	Instruction	Target memory where file register is stored	1 word		1000 words		10000 words	
			Min.	Max.	Min.	Max.	Min.	Max.
QnHCPU QnPHCPU QnPRHCPU	RBMOV	Standard RAM	20.0 $\mu$ s		91.0 $\mu$ s		775.0 $\mu$ s	
		SRAM card	22.5 $\mu$ s		545.0 $\mu$ s		5300.0 $\mu$ s	
	BMOV	Standard RAM	7.5 $\mu$ s		77.0 $\mu$ s		720.0 $\mu$ s	
		SRAM card	8.5 $\mu$ s		692.0 $\mu$ s		7050.0 $\mu$ s	
QnCPU	RBMOV	Standard RAM	45.5 $\mu$ s		215.0 $\mu$ s		1850.0 $\mu$ s	
		SRAM card	50.0 $\mu$ s		870.0 $\mu$ s		8350.0 $\mu$ s	
	BMOV	Standard RAM	17.5 $\mu$ s		179.0 $\mu$ s		1700.0 $\mu$ s	
		SRAM card	18.5 $\mu$ s		839.0 $\mu$ s		8600.0 $\mu$ s	
Q00UCPU Q01UCPU	RBMOV	Standard RAM	12.6 $\mu$ s	35.3 $\mu$ s	232.5 $\mu$ s	256.1 $\mu$ s	2211.5 $\mu$ s	2235.1 $\mu$ s
		SRAM card*2	-	-	-	-	-	-
	BMOV	Standard RAM	7.7 $\mu$ s	14.2 $\mu$ s	227.5 $\mu$ s	234.2 $\mu$ s	2206.5 $\mu$ s	2214.2 $\mu$ s
		SRAM card*2	-	-	-	-	-	-
Q02UCPU	RBMOV	Standard RAM	9.6 $\mu$ s	31.5 $\mu$ s	228.5 $\mu$ s	252.3 $\mu$ s	2208.5 $\mu$ s	2231.3 $\mu$ s
		SRAM card	9.6 $\mu$ s	31.5 $\mu$ s	378.5 $\mu$ s	401.3 $\mu$ s	3708.5 $\mu$ s	3731.3 $\mu$ s
	BMOV	Standard RAM	5.2 $\mu$ s	11.8 $\mu$ s	224.5 $\mu$ s	232.3 $\mu$ s	2204.5 $\mu$ s	2212.3 $\mu$ s
		SRAM card	5.2 $\mu$ s	11.8 $\mu$ s	374.5 $\mu$ s	382.3 $\mu$ s	3704.5 $\mu$ s	3712.3 $\mu$ s
Q03UD(E)CPU	RBMOV	Standard RAM	11.2 $\mu$ s	16.7 $\mu$ s	230.7 $\mu$ s	237.1 $\mu$ s	2210.7 $\mu$ s	2217.1 $\mu$ s
		SRAM card	11.6 $\mu$ s	16.7 $\mu$ s	380.7 $\mu$ s	387.1 $\mu$ s	3710.7 $\mu$ s	3717.1 $\mu$ s
	BMOV	Standard RAM	4.9 $\mu$ s	6.7 $\mu$ s	224.7 $\mu$ s	227.1 $\mu$ s	2204.7 $\mu$ s	2207.1 $\mu$ s
		SRAM card	5.2 $\mu$ s	6.7 $\mu$ s	374.7 $\mu$ s	377.1 $\mu$ s	3704.7 $\mu$ s	3707.1 $\mu$ s
Q04UD(E)HCPU Q06UDE(H)CPU Q10UDE(H)CPU Q13UDE(H)CPU Q20UDE(H)CPU Q26UDE(H)CPU	RBMOV	Standard RAM	9.3 $\mu$ s	15.5 $\mu$ s	118.0 $\mu$ s	124.6 $\mu$ s	1102.0 $\mu$ s	1107.6 $\mu$ s
		SRAM card	9.7 $\mu$ s	15.5 $\mu$ s	365.0 $\mu$ s	371.6 $\mu$ s	3571.0 $\mu$ s	3578.6 $\mu$ s
	BMOV	Standard RAM	4.3 $\mu$ s	6.2 $\mu$ s	113.0 $\mu$ s	115.6 $\mu$ s	1096.0 $\mu$ s	1098.6 $\mu$ s
		SRAM card	4.5 $\mu$ s	6.1 $\mu$ s	360.0 $\mu$ s	362.6 $\mu$ s	3566.0 $\mu$ s	3568.6 $\mu$ s

\*1 : Unusable for the Q00UCPU and Q01UCPU.



# INDEX

---



[Symbols]

- (BIN 16-bit subtraction operations)..... 6-22  
 \$+ (Linking character strings) ..... 6-65,6-67  
 \$=, \$<>, \$>, \$<=, \$<, \$>= (Character string data comparisons)..... 6-11  
 \$MOV (Character string transfers) ..... 6-112  
 \* (BIN 16-bit multiplication operations) ..... 6-30  
 + (BIN 16-bit addition operations)..... 6-22  
 / (BIN 16-bit division operations) ..... 6-30  
 <(BIN 16-bit data comparisons)..... 6-2  
 <=(BIN 16-bit data comparisons)..... 6-2  
 <>(BIN 16-bit data comparisons)..... 6-2  
 =(BIN 16-bit data comparisons)..... 6-2  
 >(BIN 16-bit data comparisons)..... 6-2  
 >=(BIN 16-bit data comparisons)..... 6-2

[Numerics]

16-bit data block transfers (FMOV) ..... 6-120,6-122  
 16-bit data checks (SUM) ..... 7-69  
 16-bit data exchange (XCH) ..... 6-124  
 16-bit data exclusive NOR operation (WXNR)..... 7-27  
 16-bit data searches (SER) ..... 7-66  
 16-bit dead band controls (BAND)..... 7-324  
 16-bit exclusive OR operations (WXOR) ..... 7-19  
 16-bit negation transfers (CML)..... 6-114  
 16-bit transfers (MOV) ..... 6-106  
 1-bit shift to left of n-bit data (BSFL)..... 7-49,7-51  
 1-bit shift to right of n-bit data (BSFR) ..... 7-49,7-51  
 1-word shift to left of n-word data (DSFL).... 7-54,7-56  
 1-word shift to right of n-word data (DSFR) ..... 7-54,7-56  
 32-bit data checks (DSUM) ..... 7-69  
 32-bit data exchanges (DXCH)..... 6-124  
 32-bit data exclusive NOR operation (DXNR) .... 7-27  
 32-bit data searches (DSER)..... 7-66  
 32-bit dead band controls (DBAND) ..... 7-324  
 32-bit exclusive OR operations (DXOR)..... 7-19  
 32-bit negation transfers (DCML) ..... 6-114  
 32-bit transfers (DMOV) ..... 6-106  
 4-bit dissociation of 16-bit data (DIS) ..... 7-77  
 4-bit linking of 16-bit data (UNI) ..... 7-79  
 7-segment decode (SEG)..... 7-75

[A]

A contact operation start (LD)..... 5-2  
 A contact parallel connection (OR)..... 5-2  
 A contact series connection (AND)..... 5-2  
 ACOS (COS<sup>-1</sup> operation on floating-point data (Single precision))..... 7-267  
 ACOSD (COS<sup>-1</sup> operation on floating-point data (Double precision)) ..... 7-269  
 Addition  
 Addition of floating decimal point (Double precision) (ED+)..... 6-50,6-52  
 Addition of floating decimal point (Single precision) (E+) ..... 6-46,6-48  
 BCD 4-digit addition (B+) ..... 6-34  
 BCD 8-digit addition (DB+)..... 6-38

BIN 16-bit addition operations (+) ..... 6-22  
 BIN 32-bit addition operations (D+)..... 6-26  
 Block addition (BK+)..... 6-59,6-62  
 Addition and subtraction of floating decimal point data (Double precision) (ED+, ED-) ..... 6-50,6-52  
 Addition and subtraction of floating decimal point data (Single precision) (E+, E-)..... 6-46,6-48  
 ADRSET (Indirect address read) ..... 7-395  
 ANB (Ladder block series connections)..... 5-10  
 AND (=, <>, >, <=, <, >=) (BIN 16-bit data comparisons) ..... 6-2  
 AND (A contact series connection)..... 5-2  
 AND (D=, D<>, D>, D<=, D<, D>=) (BIN 32-bit data comparisons) ..... 6-4  
 AND (E=, E<>, E>, E<=, E<, E>=) (Floating decimal point data comparisons(Single precision))..... 6-6  
 AND (ED=, ED<>, ED>, ED<=, ED<, ED>=) (Floating decimal point data comparisons(Double precision)) ..... 6-8  
 And inverse (ANI) ..... 5-3  
 AND(\$=, \$<>, \$>, \$<=, \$<, \$>=) (Character string data comparisons) ..... 6-11  
 ANDF (Pulse series connections / trailing edge leading edge)..... 5-5,5-7  
 ANDP (Pulse series connections / leading edge leading edge)..... 5-5,5-7  
 ANDPI, ANDFI ..... 5-8  
 ANI (B contact series connection) ..... 5-2  
 Annunciator output (OUT F) ..... 5-28  
 Application instructions ..... 2-29  
 Arithmetic operation instructions..... 2-16  
 ASC (Conversion from hexadecimal BIN to ASCII) ..... 7-228  
 ASIN (SIN<sup>-1</sup> operation on floating-point data (Single precision)) ..... 7-262  
 ASIND (SIN<sup>-1</sup> operation on floating-point data (Double precision)) ..... 7-265  
 ATAN (TAN<sup>-1</sup> operation on floating-point data (Single precision)) ..... 7-271  
 ATAND (TAN<sup>-1</sup> operation on floating-point data ..... 7-273

[B]

B- (BCD 4-digit subtraction)..... 6-34  
 B contact operation start (LDI)..... 5-2  
 B\* (BCD 4-digit multiplication) ..... 6-42  
 B+ (BCD 4-digit addition) ..... 6-34  
 B/ (BCD 4-digit division)..... 6-42  
 BACOS (BCD type COS<sup>-1</sup> operation) ..... 7-317  
 BAND (16-bit dead band controls)..... 7-324  
 Basic instructions ..... 2-10  
 BASIN (BCD type SIN<sup>-1</sup> operation)..... 7-315  
 BATAN (BCD type TAN<sup>-1</sup> operation) ..... 7-313  
 Batch recovery of index register (ZPOP) ..... 7-400  
 Batch reset of bit devices (BKRST) ..... 7-64  
 Batch save of index register (ZPUSH) ..... 7-400  
 BCD (BIN data to 4-digit) ..... 6-73  
 BCD 4-digit addition and subtraction operations (B+, B-) ..... 6-34

BCD 4-digit multiplication and division operations (B*, B/)	6-42
BCD 4-digit square roots (BSQR)	7-306
BCD 8-digit addition and subtraction operations (DB+, DB-)	6-38
BCD 8-digit multiplication and division operations (DB*, DB/)	6-44
BCD 8-digit square roots (BDSQR)	7-306
BCD conversion	
Conversion from BIN data to 4-digit BCD (BCD)	6-73
Conversion from BIN data to 8-digit BCD (DBCD)	6-73
BCD type COS operations (BCOS)	7-311
BCD type COS <sup>-1</sup> operations (BACOS)	7-317
BCD type SIN operation (BSIN)	7-309
BCD type SIN <sup>-1</sup> operation (BASIN)	7-315
BCD type TAN operation (BTAN)	7-313
BCD type TAN <sup>-1</sup> operations (BATAN)	7-319
BCDDA (Conversion from BCD 4-digit to decimal ASCII)	7-189
BCOS (BCD type COS operations)	7-311
BDSQR (BCD 8-digit square roots)	7-306
BIN (BCD 4-digit data to BIN data)	6-75
BIN 16-bit addition and subtraction operations (+, -)	6-22
BIN 16-bit data comparisons (=, <>, >, <=, <, >=)	6-2
BIN 16-bit data sort operations (SORT)	7-95
BIN 16-bit multiplication and division operations (*, /)	6-30
BIN 16-bit to BIN 32-bit (DBL)	6-88
BIN 16-bit to Gray code (GRY)	6-90
BIN 32-bit addition and subtraction operations (D+, D-)	6-26
BIN 32-bit block data comparisons (DBKCOMP □, DBKCOMP □ P)	6-18
BIN 32-bit data block addition and subtraction operations (DBK+(P),DBK-(P))	6-62
BIN 32-bit data comparisons (D=, D<>, D>, D<=, D<, D>=)	6-4
BIN 32-bit data sort operations (DSORT)	7-95
BIN 32-bit data to BIN 16-bit data (WORD)	6-89
BIN 32-bit data to Gray code (DGRY)	6-90
BIN 32-bit multiplication and division operations (D*, D/)	6-32
BIN block data comparisons (BKCOMP □) ...	6-15,6-18
BINDA (Conversion from BIN 16-bit data to decimal ASCII)	7-183
BINHA (Conversion from BIN 16-bit data to hexadecimal ASCII)	7-186
Bit data	3-3
Bit device output reverse (FF)	5-40
Bit device shifts (SET)	5-44
Bit processing instructions	2-34
Bit reset for word devices (BRST)	7-59
Bit set for word devices (BSET)	7-59
Bit tests (TEST/DTEST)	7-61
BK- (Block subtraction)	6-59,6-62
BK+ (Block addition)	6-59,6-62
BKAND (Block logical products)	7-9
BKBCD (Conversion from block BIN 16-bit data to BCD 4-digit data)	6-98
BKBIN (Conversion from block BCD 4-digit data to block BIN 16-bit data)	6-100
BKCOMP □ (BIN block data comparisons)	6-15,6-18
BKOR (Block logical sum operations)	7-17
BKRST (Batch reset of bit devices)	7-64
BKXNR (Block exclusive NOR operations)	7-33
BKXOR (Block exclusive OR operations)	7-25
Block 16-bit exchanges (BXCH)	6-126
Block 16-bit transfers (BMOV)	6-117
Block addition (BK+)	6-59,6-62
Block exclusive NOR operations (BKXNR)	7-33
Block exclusive OR operations (BKXOR)	7-25
Block logical products (BKAND)	7-9
Block logical sum operations (BKOR)	7-17
Block subtraction (BK-)	6-59,6-62
BMOV (Block 16-bit data transfers)	6-117
BREAK (Forced end of FOR to NEXT instruction loop)	7-108
BRST (Bit reset for word devices)	7-59
BSET (Bit set for word devices)	7-59
BSFL (1-bit shift to left of n-bit data)	7-49,7-51
BSFR (1-bit shift to right of n-bit data)	7-49,7-51
BSIN (BCD type SIN operation)	7-309
BSQR (BCD 4-digit square roots)	7-306
BTAN (BCD type TAN operation)	7-313
BTOW (Data linking in byte units)	7-85
Buffer memory access instructions	2-41
BXCH (Block 16-bit data exchanges)	6-126
[C]	
Calculation of averages for 16-bit or 32-bit data (MEAN(P),DMEAN(P))	7-103
Calculation of totals for 16-bit data (WSUM)	7-99
Calculation of totals for 32-bit data (DWSUM)	7-101,7-103
CALL (Subroutine program calls)	7-110
Cautions on programming	3-27
Changing check format of CHK instruction (CHKCIR, CHKEND)	7-179
Character string data	3-11
Character string data comparisons	6-11
Character string length detection (LEN)	7-204
Character string processing instructions	2-43
Character string search (INSTR)	7-239,7-241,7-243
Character string transfers (\$MOV)	6-112
CHKCIR (Changing check format of CHK instruction)	7-179
CHKEND (Changing check format of CHK instruction)	7-179
CHKST, CHK (Special format failure checks)	7-175
CJ (Pointer branch instruction)	6-129
Clock comparison (TM=,TM>,TM<,TM=)	7-361
Clock data addition operation (DATE+)	7-348
Clock data subtraction operation (DATE-)	7-350
Clock instructions	2-52

CML (16-bit negation transfers).....	6-114	Conversion from ASCII to hexadecimal BIN (HEX)	7-230
COM (Refresh instruction).....	7-134,7-137,7-141	Conversion from BCD 4-digit to decimal ASCII (BCDDA).....	7-189
Common logarithm operation on floating-point data (Double precision) (LOG10D(P)).....	7-302	Conversion from BCD 8-digit to decimal ASCII (DBCDDA).....	7-189
Common logarithm operation on floating-point data (Single precision) (LOG10(P)).....	7-300	Conversion from BIN 16-bit to character string (STR).....	7-206
Comparison operation instruction table.....	2-10	Conversion from BIN 16-bit to decimal ASCII (BINDA).....	7-183
Comparison operation instructions.....	6-2	Conversion from BIN 16-bit to floating decimal point (Double precision) (FLTD).....	6-81
Comparisons (BIN 16-bit data).....	6-2	Conversion from BIN 16-bit to floating decimal point (Single precision) (FLT).....	6-78
Comparisons (BIN 32-bit data).....	6-4	Conversion from BIN 16-bit to hexadecimal ASCII (BINHA).....	7-186
Comparisons (Character string data).....	6-11	Conversion from BIN 32-bit to character string (DSTR).....	7-206
Complement of 2 of BIN 16-bit data (NEG).....	6-94	Conversion from BIN 32-bit to decimal ASCII (DBINDA).....	7-183
Complement of 2 of BIN 32-bit data (DNEG).....	6-94	Conversion from BIN 32-bit to floating decimal point (Double precision) (DFLTD).....	6-81
COMRD (Reading device comment data).....	7-201	Conversion from BIN 32-bit to floating decimal point (Single precision) (DFLT).....	6-78
Conditions for execution of instructions.....	3-33	Conversion from BIN 32-bit to hexadecimal ASCII (DBINHA).....	7-186
Connection instructions		Conversion from block BCD 4-digit data to block BIN 16-bit data (BKBIN).....	6-100
Association instruction table.....	2-7	Conversion from block BIN 16-bit data to BCD 4-digit data (BKBCD).....	6-98
Ladder block parallel connection (ORB).....	5-10	Conversion from character string to BIN 16-bit (VAL).....	7-212
Ladder block series connection (ANB).....	5-10	Conversion from character string to BIN 32-bit (DVAL).....	7-212
Linking character strings (\$+).....	6-65	Conversion from character string to floating decimal point (EVAL).....	7-224
Contact instruction.....	2-6	Conversion from decimal ASCII to BCD 4-digit (DABCD).....	7-198
Contact instructions		Conversion from decimal ASCII to BCD 8-digit (DDABCD).....	7-198
Operation start (LD, LDI).....	5-2	Conversion from decimal ASCII to BIN 16-bit (DABIN).....	7-192
Parallel connection (OR, ORI).....	5-2	Conversion from decimal ASCII to BIN 32-bit (DDABIN).....	7-192
Pulse operation start (LDF, LDP).....	5-5,5-7	Conversion from floating decimal point to character string (ESTR).....	7-217
Pulse parallel connection (ORF, ORP).....	5-5,5-7	Conversion from floating-point angle to radian (Double precision) (RADD).....	7-277
Pulse serial connection (ANF, ANP).....	5-5,5-7	Conversion from floating-point angle to radian (Single precision) (RAD).....	7-275
Series connection (AND, ANI).....	5-2	Conversion from floating-point radian to angle (Double precision) (DEGD).....	7-281,7-283,7-285
Conversion		Conversion from floating-point radian to angle (Single precision) (DEG).....	7-279
BCD 4-digit to BIN (BIN).....	6-75	Conversion from hexadecimal ASCII to BIN 16-bit (HABIN).....	7-195
BCD 8-digit to BIN (DBIN).....	6-75	Conversion from hexadecimal ASCII to BIN 32-bit (DHABIN).....	7-195
BIN 16-bit to BIN 32-bit (DBL).....	6-88		
BIN 16-bit to floating decimal point (Double precision) (FLTD).....	6-81		
BIN 16-bit to floating decimal point (Single precision) (FLT).....	6-78		
BIN 16-bit to Gray code (GRY).....	6-90		
BIN 32-bit to BIN 16-bit (WORD).....	6-89		
BIN 32-bit to floating decimal point (Double precision) (DFLTD).....	6-81		
BIN 32-bit to floating decimal point (Single precision) (DFLT).....	6-78		
BIN 32-bit to Gray code (DGRY).....	6-90		
BIN to BCD 4-digit (BCD).....	6-73		
BIN to BCD 8-digit (DBCD).....	6-73		
Double precision to Single precision (EDCON).....	6-104		
Floating decimal point data to BIN 16-bit (Double precision) (INTD).....	6-86		
Floating decimal point data to BIN 16-bit (Single precision) (INT).....	6-83		
Floating decimal point data to BIN 32-bit (Single precision) (DINT).....	6-83		
Floating decimal point data to BIN32-bit (Double precision) (DINTD).....	6-86		
Gray code to BIN 16-bit (GBIN).....	6-92		
Gray code to BIN 32-bit (DGBIN).....	6-92		
Single precision to Double precision (ECON).....	6-102		



Conversion from hexadecimal BIN to ASCII (ASC)	7-228	Data table operation instructions	2-40
Conversion of Gray code to BIN 16-bit (GBIN)	6-92	DATE- (Clock data subtraction operation)	7-350
Conversion of Gray code to BIN 32-bit (DGBIN)	6-92	Date comparison (DT=,DT>,DT=)	7-356
Conversion to BIN		DATE+ (Clock data addition operation)	7-348
BCD 4-digit to BIN 16-bit (BIN)	6-75	DATERD (Reading clock data)	7-344
BCD 8-digit to BIN 32-bit (DBIN)	6-75	DATEWR (Writing clock data)	7-346
Floating decimal point data to BIN 16-bit (Double precision) (INTD)	6-86	DB- (BCD 8-digit subtraction)	6-38
Floating decimal point data to BIN 16-bit (Single precision) (INT)	6-83	DB* (BCD 8-digit multiplication)	6-44
Floating decimal point data to BIN 32-bit (Double precision) (DINTD)	6-86	DB+ (BCD 8-digit addition)	6-38
Floating decimal point data to BIN 32-bit (Single precision) (DINT)	6-83	DB/ (BCD 8-digit division)	6-44
Conversion to floating decimal point (Double precision) (FLTD, DFLTD)	6-81	DBAND (32-bit dead band controls)	7-324
Conversion to floating decimal point (Single precision) (FLT, DFLT)	6-78	DBCD (Conversion from BIN to BCD 8-digit)	6-73
COS (COS operation on floating-point data (Single precision))	7-254	DBCDDA (Conversion from BCD 8-digit to decimal ASCII)	7-189
COS operation on floating-point data (Double precision) (COSD)	7-256	DBIN (BCD 8-digit to BIN 16-bit conversion)	6-75
COS operation on floating-point data (Single precision) (COS)	7-254	DBINDA (Conversion from BIN 32-bit to decimal ASCII)	7-183
COS <sup>-1</sup> operation on floating-point data (Double precision) (ACOSD)	7-269	DBINHA (Conversion from BIN 32-bit to hexadecimal ASCII)	7-186
COS <sup>-1</sup> operation on floating-point data (Single precision) (ACOS)	7-267	DBK-	6-63
COSD (COS operation on floating-point data (Double precision))	7-256	DBK+	6-62
Count 1-phase input or down (UDCNT1)	6-143	DBL (BIN 16-bit to BIN 32-bit)	6-88
Count 2-phase input or down (UDCNT2)	6-146	DCML (32-bit negation transfers)	6-114
Counters (OUT C)	5-26	DDABCD (Conversion from decimal ASCII to BCD 8-digit)	7-198
[D]		DDABIN (Conversion from decimal ASCII to BIN 32-bit)	7-192
D- (BIN 32-bit subtraction operations)	6-26	DDEC (Decrementing 32-bit BIN)	6-71
D(P).DDR(Reading Devices to Another CPU)	10-17	Debugging and failure diagnosis instructions	2-42
D(P).DDWR(Writing Devices to Another CPU)	10-13	DEC (Decrementing 16-bit BIN)	6-69
D* (BIN 32-bit multiplication operations)	6-32	DECO (Decoding from 8 to 256 bits)	7-71
D+ (BIN 32-bit addition operations)	6-26	Decoding from 8 to 256 bits (DECO)	7-71
D/ (BIN 32-bit division operations)	6-32	Decrement	
D=, D<>, D>, D<=, D<, D>= (BIN 32-bit data comparisons)	6-4	BIN 16-bit (DEC)	6-69
DABCD (Conversion from decimal ASCII to BCD 4-digit)	7-198	BIN 32-bit (DDEC)	6-71
DABIN (Conversion from decimal ASCII to BIN 16-bit)	7-192	Decrementing 16-bit BIN (DEC)	6-69
DAND (Logical products with 32-bit data)	7-3	Decrementing 32-bit BIN (DDEC)	6-71
Data control instructions	2-49	DEG (Conversion from floating-point radian to angle (Single precision))	7-279
Data conversion instruction table	2-22	DEGD (Conversion from floating-point radian to angle (Double precision))	7-281,7-283,7-285
Data conversion instructions	6-73	Deleting data from data tables (FDEL)	7-157
Data dissociation in byte units (WTOB)	7-85	Deletion of character string (STRDEL(P))	7-243
Data link instructions	2-59	DELTA (Pulse conversion of direct output)	5-42
Data linking in byte units (BTOW)	7-85	Designating data	3-3
Data processing instructions	2-35	Designation of modification values in index modification (IXDEV, IXSET)	7-148
		Device range check	3-27
		DFLT (Conversion from BIN 32-bit to floating decimal point (Single precision))	6-78
		DFLTD (Conversion from BIN 32-bit to floating decimal point (Double precision))	6-81
		DFRO (Reading 2-word data from intelligent function modules)	7-160
		DGBIN (Conversion of Gray code to BIN 16-bit)	6-92
		DGRY (BIN 32-bit to Gray code)	6-90
		DHABIN (Conversion from hexadecimal ASCII to BIN 32-bit)	7-195

DI (Interrupt disable).....	6-133
Digit designation .....	3-4
Digit designation of bit devices .....	3-4
DINC (Incrementing 32-bit BIN).....	6-71
DINT (Floating decimal point data to BIN 32-bit (Single precision)).....	6-83
DINTD (Floating decimal point data to BIN 32-bit (Double precision)) .....	6-86
Direct 1-byte read from file register (ZRRDB)....	7-391
DIS (4-bit grouping of 16-bit data) .....	7-77
Display instructions.....	2-41
Dissociation of random data (NDIS) .....	7-81
Division	
BCD 4-digit (B/).....	6-42
BCD 8-digit division (DB/) .....	6-44
BIN 16-bit (/).....	6-30
Division of floating decimal point(Double precision) (ED/).....	6-56
Division of floating decimal point(Single precision) (E/) .....	6-54
DLIMIT (Upper and lower limit controls for BIN 32-bit) .....	7-321
DMAX (Maximum value search for 32-bit data)....	7-89
DMEAN(P).....	7-103
DMIN (Minimum value search for 32-bit data)....	7-92
DMOV (32-bit transfers) .....	6-106
DNEG (Complement of 2 of BIN 32-bit data) .....	6-94
DOR (Logical sums of 32-bit data) .....	7-11
Double precision to Single precision conversion (EDCON) .....	6-104
Double word data .....	3-6
DRCL (Left rotation of 32-bit data) .....	7-44
DRCR (Right rotation of 32-bit data) .....	7-41
DROL (Left rotation of 32-bit data) .....	7-44
DROR (Right rotation of 32-bit data) .....	7-41
DSCL(P) .....	7-331
DSCL2(P) .....	7-335
DSER (32-bit data searches).....	7-66
DSFL (1-word shift to left of n-word data)....	7-54,7-56
DSFR (1-word shift to right of n-word data) .....	7-54,7-56
DSORT (BIN 32-bit data sort).....	7-95
DSTR (Conversion from BIN 32-bit to character string) .....	7-206
DSUM (32-bit data checks) .....	7-69
DTEST (Bit tests).....	7-61
DTO (Writing 2-word data to intelligent function modules).....	7-163
DUTY (Timing pulse generation) .....	7-388
DVAL (Conversion from character string to BIN 32-bit) .....	7-212
DWSUM (Calculation of totals for 32-bit data) .....	7-101,7-103
DXCH (16-bit data exchanges).....	6-124
DXNR (32-bit data exclusive NOR operation) .....	7-27
DXOR (32-bit exclusive OR operations).....	7-19
DZONE (Zone control for BIN 32-bit data) .....	7-327,7-330,7-334

[E]	
E- (Subtraction of floating decimal point data (Single precision)) .....	6-46,6-48
E* (Multiplication of floating decimal point data (Single precision)) .....	6-54
E+ (Addition of floating decimal point data (Single precision)) .....	6-46,6-48
E/ (Division of floating decimal point data (Single precision)) .....	6-54
E=, E<>, E>, E<=, E<, E>= (Floating decimal point data comparisons(Single precision)) .....	6-6
ECALL (Sub-routine calls between program files) .....	7-120
ECON (Single precision to Double precision conversion) .....	6-102
ED- (Subtraction of floating decimal point data (Double precision)) .....	6-50,6-52
ED* (Multiplication of floating decimal point data (Double precision)) .....	6-56
ED+ (Addition of floating decimal point data (Double precision)) .....	6-50,6-52
ED/ (Division of floating decimal point data (Double precision)) .....	6-56
ED=,ED<>,ED>,ED<=,ED<,ED>= (Floating decimal point data comparisons (Double precision)) .....	6-8
EDCON (Double precision to Single precision conversion) .....	6-104
EDMOV (Floating-point data transfer (Double precision)) .....	6-110
EDNEG (Floating-point sign inversion (Double precision)) .....	6-97
EFCALL (Output OFF calls between program files) .....	7-125
EGF (Pulse operation results / leading edge).....	5-18
EGP (Pulse operation results / trailing edge).....	5-18
EI (Interrupt enable).....	6-133
EMOD (Floating decimal point to BCD) .....	7-245
EMOV (Floating-point data transfer (Single precision)) .....	6-108
ENCO (Encoding from 256 to 8 bits) .....	7-73
Encoding from 256 to 8 bits (ENCO) .....	7-73
END (End sequence program) .....	5-53
End main routine program (FEND) .....	5-51
End sequence program (END) .....	5-53
ENEG (Floating-point sign inversion(Single precision)) .....	6-96
EREXP (From BCD format data to floating decimal point).....	7-248
Error display and annunciator reset instruction (LEDR) .....	7-172
ESTR (Conversion from floating decimal point to character string).....	7-217
EVAL (Conversion from character string to floating decimal point) .....	7-217
EXP (Exponent operation on floating-point data (Single precision)).....	7-291
Expansion clock data addition operation (S.DATE+) .....	7-366



[J]		[M]	
	JMP (Pointer branch)..... 6-129		Master control instructions ..... 5-47
	Jump to END (GOEND)..... 6-132		Matrix input (MTR) ..... 6-166
[K]			MAX (Maximum value search for 16-bit data) .... 7-89
	KEY (Numerical key input from keyboard)..... 7-396		Maximum value search for 16-bit data (MAX) .... 7-89
[L]			Maximum value search for 32-bit data (DMAX)... 7-89
	Ladder block parallel connections (ORB) ..... 5-10		MC (Setting the master control)..... 5-47
	Ladder block series connections (ANB) ..... 5-10		MCR (Resetting the master control) ..... 5-47
	LD (\$=, \$<>, \$>, \$<=, \$<, \$>=) (Character string data comparisons) ..... 6-11		MEAN(P)..... 7-103
	LD (=, <>, >, <=, <, >=) (BIN 16-bit data comparisons) ..... 6-2		MEF (Pulse operation results / trailing edge)..... 5-17
	LD (A contact operation start)..... 5-2		MEP (Pulse operation results / leading edge) .... 5-17
	LD (D=, D<>, D>, D<=, D<, D>=) (BIN 32-bit data comparisons) ..... 6-4		MIDR (Random selection from character strings) ..... 7-235
	LD (E=, E<>, E>, E<=, E<, E>=) (Floating decimal point data comparisons(Single precision)) ..... 6-6		MIDW (Random replacement in character strings) ..... 7-235
	LD (ED=, ED, ED>, ED<=, ED<, ED>=) (Floating decimal point data comparisons(Double precision)) ..... 6-8		MIN (Minimum value search for 16-bit data)..... 7-92
	LDF (Pulse operation start / trailing edge)..... 5-5,5-7		Minimum value search for 16-bit data (MIN)..... 7-92
	LDI (B contact operation start)..... 5-2		Minimum value search for 32-bit data (DMIN) .... 7-92
	LDP (Pulse operation start / leading edge)..... 5-5,5-7		MOV (16-bit transfers) ..... 6-106
	LDPI, LDFI ..... 5-7		MPP (Operation results pop) ..... 5-12
	Leading edge output (PLS)..... 5-37		MPS (Operation results push) ..... 5-12
	LEDR (Error display and annunciator reset instruction) ..... 7-172		MRD (Operation results read)..... 5-12
	LEFT (Extracting character string data from the left) ..... 7-232		MTR (Matrix input) ..... 6-166
	Left rotation of 16-bit data (ROL, RCL)..... 7-38		Multiplication
	Left rotation of 32-bit data (DROL, DRCL) ..... 7-44		BCD 4-digit (B*)..... 6-42
	LEN (Character string length detection) ..... 7-204		BCD 8-digit (DB*) ..... 6-44
	LIMIT (Upper and lower limit controls for BIN 16-bit) ..... 7-321		BIN 16-bit (*) ..... 6-30
	Link refresh instructions..... 2-59		BIN 32-bit (D*) ..... 6-32
	Linking character strings (\$) ..... 6-65,6-67		Multiplication of floating decimal point (Double precision) (ED*)..... 6-56
	Linking of random data (NUNI)..... 7-81		Multiplication of floating decimal point (Single precision) (E*) ..... 6-54
	Load (LD)..... 5-2		Multiplication and division of floating decimal point (Double precision)(ED*, ED/)..... 6-56
	Load + unload (PSWAPP) ..... 7-445		Multiplication and division of floating decimal point (Single precision)(E*, E/) ..... 6-54
	Load inverse (LDI) ..... 5-2		
	Load program from Memory Card (PLOADP) ... 7-440		[N]
	LOG (Natural logarithm operation on floating-point data (Single precision))..... 7-296,7-302		Natural logarithm operation on floating-point data (Double precision) (LOGD) ..... 7-298
	LOGD (Natural logarithm operation on floating-point data (Double precision)) ..... 7-298		Natural logarithm operation on floating-point data (Single precision) (LOG) ..... 7-296,7-302
	Logical operation instructions ..... 2-29		n-bit shift to left of 16-bit data (SFL) ..... 7-46
	Logical product ..... 7-2		n-bit shift to right of 16-bit data (SFR)..... 7-46
	Logical products with 16-bit data (WAND)..... 7-3		n-bit shift to right or left of n-bit data (SFTBR(P), SFTBL(P))..... 7-51
	Logical products with 32-bit data (DAND) ..... 7-3		n-bit shift to right or left of n-word data (SFTWR(P), SFTWL(P))..... 7-56
	Logical sum ..... 7-2		NEG (complement of 2 of BIN 16-bit data) ..... 6-94
	Logical sums of 16-bit data (WOR) ..... 7-11		Network refresh instruction (ZCOM) ..... 8-2
	Logical sums of 32-bit data (DOR) ..... 7-11		NEXT (FOR to NEXT)..... 7-105
	Low speed retentive timer (OUTH ST) ..... 5-22		No operation (NOP, NOPLF, PAGE) ..... 5-57
	Low speed timer (OUT T) ..... 5-22		NOP ..... 5-57
			NOP (No operation) ..... 5-57
			NOPLF (No operation page change) ..... 5-57
			Number of steps ..... 3-34
			Numerical key input (KEY)..... 7-396
			Numerical key input from keyboard (KEY)..... 7-396
			NUNI (Linking of random data) ..... 7-81

[O]	
Operation errors	3-27
Operation results inversion (INV)	5-15
Operation results pop (MPP)	5-12
Operation results push (MPS)	5-12
Operation results read (MRD)	5-12
Operation start (LD, LDI)	5-2
OR (\$=, \$<>, \$>, \$<=, \$<, \$>=) (Character string data comparisons)	6-11
OR (=, <>, >, <=, <, >=) (BIN 16-bit data comparisons)	6-2
OR (A contact parallel connection)	5-2
OR (D=, D<>, D>, D<=, D<, D>=) (BIN 32-bit data comparisons)	6-4
OR (E=, E<>, E>, E<=, E<, E>=) (Floating decimal point data comparisons (Single precision))	6-6
OR (ED=, ED<>, ED>, ED<=, ED<, ED>=) (Floating decimal point data comparisons (Double precision))	6-8
Or inverse (ORI)	5-2
ORB (Ladder block parallel connections)	5-10
ORF (Pulse parallel connection / trailing edge)	5-5,5-7
ORI (B contact parallel connection)	5-2
ORP (Pulse parallel connection / leading edge)	5-5,5-7
ORPI, ORFI	5-8
Other convenient instructions	2-6
Other instructions	5-55
Application instructions	2-29
Sequence instructions	2-6
OUT	
Annunciator output (OUT F)	5-28
Counters (OUT C)	5-26
High speed retentive timer (OUTH ST)	5-22
High speed timer (OUTH T)	5-22
Low speed retentive timer (OUT ST)	5-22
Low speed timer (OUT T)	5-22
Output (OUT)	5-20
Out instructions (OUT)	5-20
Output instruction table	2-8
Output instructions (OUT)	5-20
Output of sub-routine program OFF calls (FCALL)	7-116
Output OFF calls between program files (EFCALL)	7-125
Output reverse (FF)	5-40

[P]	
PAGE (No operation page change)	5-57
Page change (NOPLF)	5-57
Page change (PAGE n)	5-57
Parallel connection (OR, ORI)	5-2
Parallel connections (ORB)	5-10
PCHK (Program low speed execution registration instruction)	7-384
PLF (Trailing edge output)	5-37
PLOADP (Load program from Memory Card)	7-440

PLOW (Program low speed execution registration)	7-382
PLS (Leading edge output)	5-37
PLSY (Fixed cycle pulse output)	6-162
POFF (Program output OFF standby instruction)	7-378
Pointer branching instruction (CJ, SCJ, JMP)	6-129
Pop (MPP)	5-12
PR (Print ASCII code instruction)	7-166
PRC (Print comment instruction)	7-169
Print ASCII code instruction (PR)	7-166
Print comment instruction (PRC)	7-169
Program branch instruction table	2-27
Program control instructions	2-56
Program execution control instruction table	2-27
Program low speed execution registration instruction (PCHK)	7-384
Program low speed execution registration (PLOW)	7-382
Program output OFF standby instruction (POFF)	7-378
Program scan execution registration instruction (PSCAN)	7-380
Program standby instruction (PSTOP)	7-377
PSCAN (Program scan execution registration instruction)	7-380
PSTOP (Program standby instruction)	7-377
PSWAPP (Load + unload)	7-445
Pulse conversion	
(DELTA)	5-42
(EGF, EGP)	5-18
(MEF, MEP)	5-17
Pulse conversion of direct output (DELTA)	5-42
Pulse density measurement (SPD)	6-160
Pulse NOT operation start, pulse NOT series connection, pulse NOT parallel connection LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI)	5-7
Pulse operation results	
Operation result conversions (MEF, MEP)	5-17
Pulse conversions of edge relay operation results (EGF, EGP)	5-18
Pulse operation start (LDF, LDP)	5-5,5-7
Pulse parallel connection (ORF, ORP)	5-5,5-7
Pulse series connection (ANDF, ANDP)	5-5
Pulse width modulation (PWM)	6-164
PUNLOADP (Unload program from program memory)	7-443
Push (MPS)	5-12
PWM (Pulse width modulation)	6-164

[Q]	
QCDSET (File setting for comments)	7-342
QCPU dedicated instructions	2-60
QDRSET (Setting files for file register use)	7-339

[R]	
RAD (Conversion from floating-point angle to radian (Single precision))	7-275

RADD (Conversion from floating-point angle to radian (Double precision))	7-277	ROTC (Rotary table shortest direction control)	6-154
RAMP (Ramp signal)	6-157	RSET (Switching file register numbers)	7-337
Ramp signal (RAMP)	6-157	RST	
Random number generation (RND/SRND)	7-304	Resetting devices (RST)	5-32
Random selection from and replacement in character strings (MIDR)	7-235	Resetting the annunciators (RST F)	5-35
Random selection replacement in character strings (MIDW)	7-235	RTREAD (Reading routing information)	8-6
RBMOV (High-speed block transfer of file register)	7-448	RTWRITE (Writing routing information)	8-8
RCL (Left rotation of 16-bit data)	7-38		
RCR (Right rotation of 16-bit data)	7-35	[S]	
Read (MRD)	5-12	S.DATE- (Expansion clock data subtraction operation)	7-366
Read data from standard ROM (S.DEVLD)	7-438	S.DATE+ (Expansion clock data addition operation)	7-366
Reading 1-word data from intelligent function modules (FROM)	7-160	S.DATERD (Reading expansion clock data)	7-366
Reading 2-word data from intelligent function modules (DFRO)	7-160	S.DEVLD (Read data from standard ROM)	7-438
Reading clock data (DATERD)	7-344	S.TO (Write to host CPU shared memory)	9-4
Reading data from designated file (SP.FREAD)	7-424	Scaling (Point-by-point coordinate data) (SCL(P), DSCL(P))	7-330
Reading device comment data (COMRD)	7-201	Scaling (Point-by-point coordinate data) (SCL2(P), DSCL2(P))	7-334
Reading expansion clock data (S.DATERD)	7-366	SCJ (Pointer branching instruction)	6-129
Reading from other CPU shared memory (FROM)	9-12	SCL(P)	7-330
Reading module information (UNIRD)	7-402	SCL2	7-334
Reading newest data from data tables (FPOP)	7-155	SECOND (Time data conversion)	7-352
Reading oldest data from data tables (FIFR)	7-153	SEG (7-segment decode)	7-75
Reading routing information (RTREAD)	8-6	Sequence instructions	2-6
Real number data	3-8	Sequence program stop (STOP)	5-55
Recovery from interrupt programs (IRET)	6-139	SER (16-bit data searches)	7-66
Refresh instruction (COM)	7-134	Series connection (AND, ANI)	5-2
Related programming manuals	1-2	Series connections (ANB)	5-10
Resetting devices (RST)	5-32,5-35	SET	
Resetting the annunciators (RST F)	5-35	Setting devices (SET)	5-30
Resetting the master control (MCR)	5-47	Setting the annunciators (SET F)	5-35
Resetting watchdog timer (WDT)	7-386	Setting devices (SET)	5-30,5-35
RET (Return from sub-routine programs)	7-115	Setting files for file register use (QDRSET)	7-339
Return from sub-routine programs (RET)	7-115	Setting the annunciators (SET F)	5-35
Revercing		Setting the master control (MC)	5-47
Bit device output reverse (FF)	5-40	SFL (n-bit shift to left of 16-bit data)	7-46
Floating-point sign inversion (Double precision) (EDNEG)	6-97	SFR (n-bit shift to right of 16-bit data)	7-46
Floating-point sign inversion (Single precision) (ENEG)	6-96	SFT (Bit device shifts)	5-44
Operation results inversion (INV)	5-15	SFTBL(P)	7-52
RFS (I/O refresh)	6-141	SFTBR(P)	7-51
RIGHT (Extracting character string data from the right)	7-232	SFTWL(P)	7-57
Right rotation of 16-bit data (ROR, RCR)	7-35	SFTWR(P)	7-56
Right rotation of 32-bit data (DROR, DRCL)	7-41	Shift instruction	5-44,7-46
RND (Random number generation and series update)	7-304	(Application instructions)	2-29
ROL (Left rotation of 16-bit data)	7-38	Shift instruction table	
ROR (Right rotation of 16-bit data)	7-35	(Sequence instructions)	2-6
Rotary table shortest direction control (ROTC)	6-154	SIN (SIN operation on floating-point data (Single precision))	7-250
Rotation instructions	2-32	SIN operation on floating-point data (Double precision) (SIND)	7-252
		SIN operation on floating-point data (Single precision) (SIN)	7-250
		SIN <sup>-1</sup> operation on floating-point data (Double precision) (ASIND)	7-265
		SIN <sup>-1</sup> operation on floating-point data (Single precision) (ASIN)	7-262

SIND (SIN operation on floating-point data (Double precision)).....	7-252
Single precision to Double precision conversion (ECON).....	6-102
SORT (BIN 16-bit data sort).....	7-95
SP.CONTSW (System switching instruction).....	11-2
SP.DEVST (Writing data to standard ROM).....	7-436
SP.FREAD (Reading data from designated file).....	7-424
SP.FWRITE (Writing data to designated file)....	7-413
SPD (Pulse density measurement).....	6-160
Special format failure checks (CHKST, CHK) ...	7-175
Special function instructions.....	2-46
Special timer (STMR).....	6-151
SQR (Square root operation for floating-point data (Single precision)).....	7-287
SQRD (Square root operation for floating-point data (Double precision)).....	7-289
Square root operation for floating-point data (Double precision) (SQRD).....	7-289
Square root operation for floating-point data (Single precision) (SQR).....	7-287
SRND (Random number generation and series updates).....	7-304
STMR (Special function timer).....	6-151
STOP (Sequence program stop).....	5-55
STR (Conversion from BIN 16-bit to character string).....	7-206
Structure creation instructions.....	2-38
Subroutine program calls (CALL).....	7-110
Subroutine calls (XCALL).....	7-129
Subroutine calls between program files (ECALL).....	7-120
Subroutine program output OFF calls (FCALL).....	7-116
Subset processing.....	3-25
Subtraction	
BCD 4-digit subtraction (B-).....	6-34
BCD 8-digit subtraction (DB-).....	6-38
BIN 16-bit subtraction operations (-).....	6-22
BIN 32-bit subtraction operations (D-).....	6-26
Block subtraction (BK-).....	6-59,6-62
Subtraction of floating decimal point data (Double precision) (ED-).....	6-50,6-52
Subtraction of floating decimal point data (Single precision) (E-).....	6-46,6-48
SUM (16-bit data checks).....	7-69
SWAP (Upper and lower byte exchanges).....	6-128
Switching file register numbers (RSET).....	7-337
Switching instructions.....	2-51
System Switching (SP.CONTSW).....	11-2

[T]

TAN (TAN operation on floating-point data (Single precision)).....	7-258
TAN operation on floating-point data (Double precision)(TAND).....	7-260
TAN operation on floating-point data (Single precision)(TAN).....	7-258

TAN <sup>-1</sup> operation on floating-point data (Double precision)(ATAND).....	7-273
TAN <sup>-1</sup> operation on floating-point data (Single precision)(ATAN).....	7-271
TAND (TAN operation on floating-point data (Double precision)).....	7-260
Teaching timer (TTMR).....	6-149
Termination instruction table.....	2-9
TEST (Bit tests).....	7-61
TIMCHK (Time check instruction).....	7-390
Time check instruction (TIMCHK).....	7-390
Time data conversion (HOUR).....	7-354,7-356,7-361
Time data conversion (SECOND).....	7-352
Timer (OUT T).....	5-22
Timing pulse generation (DUTY).....	7-388
TO (Writing 1-word data to intelligent function modules).....	7-163
TRACE (Trace set).....	7-411
TRACER (Trace reset).....	7-411
TTMR (Teaching timer).....	6-149
Types of Instructions.....	2-2

[U]

UDCNT1 (Counter 1-phase input up or down) ..	6-143
UDCNT2 (Counter 2-phase input up or down) ..	6-146
UNI (4-bit linking of 16-bit data).....	7-79
UNIRD (Reading module information).....	7-402
Unload program from program memory (PUNLOADP).....	7-443
Up / Down counter	
Count 1-phase input or dawn (UDCNT1).....	6-143
Count 2-phase input or down (UDCNT2).....	6-146
Upper and lower byte exchanges (SWAP).....	6-128
Upper and lower limit controls for BIN 32-bit (DLIMIT).....	7-321

[V]

VAL (Conversion from character string to BIN 16-bit).....	7-212
-----------------------------------------------------------	-------

[W]

WAND (Logical products with 16-bit data).....	7-3
WDT (Resetting watchdog timer).....	7-386
WOR (Logical sums of 16-bit data).....	7-11
WORD (Conversion from BIN 32-bit to BIN 16-bit).....	6-89
Word data.....	3-4
Word device bit designation.....	3-3
Writing 1-word data to intelligent function modules (TO).....	7-163
Writing 2-word data to intelligent function modules (DTO).....	7-163
Writing clock data (DATEWR).....	7-346
Writing data to designated file (SP.FWRITE)....	7-413
Writing data to standard ROM (SP.DEVST).....	7-436
Writing data to the data tables (FIFW).....	7-151
Writing routing information (RTWRITE).....	8-8
Writing to the CPU shared memory of host CPU...	9-2

S.TO.....	9-4
TO.....	9-7
WSUM (Calculation of totals for 16-bit data) .....	7-99
WTOB (Data dissociation in byte units).....	7-85
WXNR (16-bit data exclusive NOR operation).....	7-27
WXNR (16-bit data non-exclusive logical sum operations).....	7-30
WXOR (16-bit exclusive OR operations) .....	7-19,7-22

[X]

XCALL (Subroutine program call).....	7-129
XCH (32-bit data exchange) .....	6-124

[Z]

ZCOM (Network refresh instruction).....	8-2
ZONE (Zone control for BIN 16-bit) .....	7-327,7-330,7-334
Zone control for BIN 16-bit (ZONE) .....	7-327,7-330,7-334
Zone control for BIN 32-bit data (DZONE) .....	7-327,7-330,7-334
ZPOP (Batch recovery of index register).....	7-400
ZPUSH (Batch save of index register).....	7-400
ZRRDB (Direct 1-byte read from file register)....	7-391
ZRWRB (File register direct 1-byte write).....	7-393



# **Warranty**

Please confirm the following product warranty details before using this product.

## **1. Gratis Warranty Term and Gratis Warranty Range**

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place.

Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

- (1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.
- (2) Even within the gratis warranty term, repairs shall be charged for in the following cases.
  1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
  2. Failure caused by unapproved modifications, etc., to the product by the user.
  3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
  4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
  5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
  6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
  7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

## **2. Onerous repair term after discontinuation of production**

- (1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued.

Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.
- (2) Product supply (including repair parts) is not available after production is discontinued.

## **3. Overseas service**

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

## **4. Exclusion of loss in opportunity and secondary loss from warranty liability**

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation of damages caused by any cause found not to be the responsibility of Mitsubishi, loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products, special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products, replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

## **5. Changes in product specifications**

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

## **6. Product application**

- (1) In using the Mitsubishi MELSEC programmable controller, the usage conditions shall be that the application will not lead to a major accident even if any problem or fault should occur in the programmable controller device, and that backup and fail-safe functions are systematically provided outside of the device for any problem or fault.
- (2) The Mitsubishi programmable controller has been designed and manufactured for applications in general industries, etc. Thus, applications in which the public could be affected such as in nuclear power plants and other power plants operated by respective power companies, and applications in which a special quality assurance system is required, such as for Railway companies or Public service purposes shall be excluded from the programmable controller applications.

In addition, applications in which human life or property that could be greatly affected, such as in aircraft, medical applications, incineration and fuel devices, manned transportation, equipment for recreation and amusement, and safety devices, shall also be excluded from the programmable controller range of applications.

However, in certain cases, some applications may be possible, providing the user consults their local Mitsubishi representative outlining the special requirements of the project, and providing that all parties concerned agree to the special circumstances, solely at the users discretion.

Microsoft, Windows, Windows NT are registered trademarks of Microsoft Corporation in the United States and other countries.

Pentium and Celeron are trademarks of Intel Corporation in the United States and other countries.

Ethernet is a trademark of Xerox Co., Ltd. in the United States.

CompactFlash is a trademark of SanDisk Corporation.

VxWorks, Tornado, WindPower, WindSh and WindView are registered trademarks of Wind River Systems, Inc.

Other company names and product names used in this document are trademarks or registered trademarks of respective owners.



# QCPU Programming Manual

## Common Instruction 1/2

MODEL	QCPU-P-KY-E
MODEL CODE	13JW10
SH(NA)-080809ENG(1/2)-C(0907)KWIX	

 **MITSUBISHI ELECTRIC CORPORATION**

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN  
NAGOYA WORKS : 1-14, YADA-MINAMI 5-CHOME, HIGASHI-KU, NAGOYA, JAPAN

When exported from Japan, this manual does not require application to the Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.

# MITSUBISHI

Mitsubishi Programmable Controller

MELSEC **Q** series

---

## QCPU Programming Manual

Common Instruction 2/2

# QSERIES



# ● SAFETY PRECAUTIONS ●

(Always read these cautions before using the product)

Before using this product, please read this manual and the related manuals introduced in this manual, and pay full attention to safety to handle the product correctly.

Please store this manual in a safe place and make it accessible when required. Always forward a copy of the manual to the end user.

# REVISIONS

\*The manual number is given on the bottom left of the back cover.

Print Date	*Manual Number	Revision
Dec., 2008	SH (NA)-080809ENG-A	First edition
Mar., 2009	SH (NA)-080809ENG-B	<p><b>Partial corrections</b></p> <p>Section 3.3, 3.8, 5.1.3, 6.1.7, 6.2.14, 7.3.3, 7.11.18, 7.11.19, 7.12.1.5, 12.7, 7.12.11, 7.12.25, 7.12.26, 7.13.4, 7.13.5, 7.15.7, 7.15.8</p>
Jul., 2009	SH (NA)-080809ENG-C	<p>Revision because of function support by the Universal model QCPU having a serial number "11043" or later</p> <p><b>Partial corrections</b></p> <p>Section 2.1, 2.5.6, 2.5.18, 2.5.20, 7.6.9, 7.12.7, 7.12.11, 12.1.3, 12.1.4, APPENDIX 1.2, 1.3, 1.4.2, 3, 5.1</p> <p><b>Additions</b></p> <p>Section 2.5.16, 7.16, 7.18.10</p> <p><b>Modification</b></p> <p>Section 2.5.21 → 2.5.22, Section 2.5.22 → 2.5.21, Section 9.13 → 7.6.10,            Section 9.14 → 7.6.1, Section 9.15 → 7.16, Section 9.15.1 → 7.16.1, Section 9.15.2 → 7.16.2,            Section 9.15.3 → 7.16.3, Section 9.1 → 7.18.9, Section 9.2 → 7.18.11, Section 9.3 → 7.18.12,            Section 9.4 → 7.18.13, Section 9.5 → 7.18.14, Section 9.6 → 7.18.15, Section 9.7 → 7.18.16,            Section 9.8 → 7.18.17, Section 9.9 → 7.18.18, Section 9.10 → 7.18.19, Section 9.11 → 9.1,            Section 9.11.1 → 9.1.1, Section 9.11.2 → 9.1.2, Section 9.12 → 9.2, Section 9.12.1 → 9.2.1,            Chapter 10 → 11, Chapter 11 → 10</p>

Japanese Manual Version SH-080804-B

This manual confers no industrial property rights or any rights of any other kind, nor does it confer any patent licenses. Mitsubishi Electric Corporation cannot be held responsible for any problems involving industrial property rights which may occur as a result of using the contents noted in this manual.

© 2008 MITSUBISHI ELECTRIC CORPORATION



# INTRODUCTION

This manual explains the common instructions required for programming of the QCPU.

- The common instructions refer to all instructions except those dedicated to special function modules (such as AJ71QC24 and AJ71PT32-S3) and to AD57 models, as well as PID control instructions, SFC instructions and ST instructions.

Before using this product, please read this manual and the relevant manuals carefully and develop familiarity with the functions and performance of the Q series programmable controller to handle the product correctly.

## ■ Relevant CPU module

CPU module	Model
Basic model QCPU	Q00JCPU, Q00CPU, Q01CPU
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU
Redundant CPU	Q12PRHCPU, Q25PRHCPU
Universal model QCPU	Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU

# CONTENTS

SAFETY PRECAUTIONS .....	A - 1
REVISIONS .....	A - 2
INTRODUCTION .....	A - 3
CONTENTS .....	A - 4
MANUALS.....	A - 14

## Common Instructions 1/2

<b>1. GENERAL DESCRIPTION</b>	<b>1 - 1 to 1 - 8</b>
1.1 Related Programming Manuals	1 - 2
1.2 Abbreviations and Generic Names	1 - 5
<b>2. INSTRUCTION TABLES</b>	<b>2 - 1 to 2 - 62</b>
2.1 Types of Instructions	2 - 2
2.2 How to Read Instruction Tables	2 - 4
2.3 Sequence Instructions	2 - 6
2.3.1 Contact instructions .....	2 - 6
2.3.2 Association instructions .....	2 - 7
2.3.3 Output instructions.....	2 - 8
2.3.4 Shift instructions .....	2 - 8
2.3.5 Master control instructions.....	2 - 9
2.3.6 Termination instructions .....	2 - 9
2.3.7 Other instructions .....	2 - 9
2.4 Basic instructions	2 - 10
2.4.1 Comparison operation instructions .....	2 - 10
2.4.2 Arithmetic operation instructions .....	2 - 16
2.4.3 Data conversion instructions .....	2 - 22
2.4.4 Data transfer instructions.....	2 - 24
2.4.5 Program branch instructions.....	2 - 27
2.4.6 Program execution control instructions .....	2 - 27
2.4.7 I/O refresh instructions .....	2 - 27
2.4.8 Other convenient instructions .....	2 - 28
2.5 Application Instructions	2 - 29
2.5.1 Logical operation instructions .....	2 - 29
2.5.2 Rotation instructions .....	2 - 32
2.5.3 Shift instructions .....	2 - 33
2.5.4 Bit processing instructions.....	2 - 34
2.5.5 Data processing instructions .....	2 - 35
2.5.6 Structure creation instructions .....	2 - 38
2.5.7 Data table operation instructions.....	2 - 40
2.5.8 Buffer memory access instructions.....	2 - 41
2.5.9 Display instructions.....	2 - 41
2.5.10 Debugging and failure diagnosis instructions .....	2 - 42

2.5.11	Character string processing instructions .....	2 - 43
2.5.12	Special function instructions .....	2 - 46
2.5.13	Data control instructions .....	2 - 49
2.5.14	Switching instructions .....	2 - 51
2.5.15	Clock instructions .....	2 - 52
2.5.16	Expansion clock instruction .....	2 - 55
2.5.17	Program control instructions .....	2 - 56
2.5.18	Other instructions .....	2 - 57
2.5.19	Instructions for Data Link.....	2 - 59
2.5.20	Multiple CPU dedicated instruction.....	2 - 60
2.5.21	Multiple CPU high-speed transmission dedicated instruction.....	2 - 60
2.5.22	Redundant system instructions (For Redundant CPU) .....	2 - 61

### 3. CONFIGURATION OF INSTRUCTIONS 3 - 1 to 3 - 48

3.1	Configuration of Instructions .....	3 - 2
3.2	Designating Data .....	3 - 3
3.2.1	Using bit data.....	3 - 3
3.2.2	Using word (16 bits) data.....	3 - 4
3.2.3	Using double word data (32 bits).....	3 - 6
3.2.4	Using real number data .....	3 - 8
3.2.5	Using character string data.....	3 - 11
3.3	Indexing .....	3 - 12
3.4	Indirect Specification .....	3 - 23
3.5	Reducing Instruction Processing Time .....	3 - 25
3.5.1	Subset Processing.....	3 - 25
3.5.2	Operation processing with standard device registers (Z) (only Universal model QCPU) .....	3 - 26
3.6	Cautions on Programming (Operation Errors) .....	3 - 27
3.7	Conditions for Execution of Instructions .....	3 - 33
3.8	Counting Step Number .....	3 - 34
3.9	Operation when the OUT, SET/RST, or PLS/PLF Instructions Use the Same Device .....	3 - 39
3.10	Precautions for Use of File Registers .....	3 - 44

### 4. HOW TO READ INSTRUCTIONS 4 - 1 to 4 - 4

### 5. SEQUENCE INSTRUCTIONS 5 - 1 to 5 - 60

5.1	Contact Instructions .....	5 - 2
5.1.1	Operation start, series connection, parallel connection (LD,LDI,AND,ANI,OR,ORI)....	5 - 2
5.1.2	Pulse operation start, pulse series connection, pulse parallel connection (LDP,LDF,ANDP,ANDF,ORP,ORF) .....	5 - 5
5.1.3	Pulse NOT operation start, pulse NOT series connection, pulse NOT parallel connection (LDPI,LDFI,ANDPI,ANDFI,ORPI,ORFI) .....	5 - 7
5.2	Association Instructions .....	5 - 10
5.2.1	Ladder block series connection and parallel connection (ANB,ORB) .....	5 - 10
5.2.2	Operation results push,read,pop (MPS,MRD,MPP) .....	5 - 12

5.2.3	Operation results inversion (INV) .....	5 - 15
5.2.4	Operation result conversions (MEP,MEF) .....	5 - 17
5.2.5	Pulse conversions of edge relay operation results (EGP,EGF).....	5 - 18
5.3	<b>Output Instructions</b> .....	5 - 20
5.3.1	Out instruction (excluding timers, counters, and annunciators) (OUT).....	5 - 20
5.3.2	Timers (OUT T,OUTH T) .....	5 - 22
5.3.3	Counter (OUT C) .....	5 - 26
5.3.4	Annunciator output (OUT F) .....	5 - 28
5.3.5	Setting devices (except for annunciators) (SET) .....	5 - 30
5.3.6	Resetting devices (except for annunciators) (RST).....	5 - 32
5.3.7	Setting and resetting the annunciators (SET F,RST F) .....	5 - 35
5.3.8	Leading edge and trailing edge outputs (PLS,PLF).....	5 - 37
5.3.9	Bit device output reverse (FF) .....	5 - 40
5.3.10	Pulse conversions of direct outputs (DELTA(P)) .....	5 - 42
5.4	<b>Shift Instructions</b> .....	5 - 44
5.4.1	Bit device shifts (SFT(P)).....	5 - 44
5.5	<b>Master Control Instructions</b> .....	5 - 47
5.5.1	Setting and resetting the master control (MC,MCR).....	5 - 47
5.6	<b>Termination Instructions</b> .....	5 - 51
5.6.1	End main routine program (FEND).....	5 - 51
5.6.2	End sequence program (END) .....	5 - 53
5.7	<b>Other instructions</b> .....	5 - 55
5.7.1	Sequence program stop (STOP) .....	5 - 55
5.7.2	No operations (NOP,NOPLF,PAGE n) .....	5 - 57

## 6. BASIC INSTRUCTIONS 6 - 1 to 6 - 168

6.1	<b>Comparison Operation Instructions</b> .....	6 - 2
6.1.1	BIN 16-bit data comparisons (=,<>,>,<=,<,>=) .....	6 - 2
6.1.2	BIN 32-bit data comparisons (D=D<>,D>,D<=,D<,D>=) .....	6 - 4
6.1.3	Floating decimal point data comparisons (Single precision) (E=E,<>,E>,E<=,E<,E>=).....	6 - 6
6.1.4	Floating decimal point data comparisons (Double precision) (ED=,ED<>,ED>,ED<=,ED<,ED>=) .....	6 - 8
6.1.5	Character string data comparisons (\$=,\$<>,\$>,\$<=,\$<,\$>=) .....	6 - 11
6.1.6	BIN block data comparisons (BKCMP <input type="checkbox"/> ,BKCMP <input type="checkbox"/> P) .....	6 - 15
6.1.7	BIN 32-bit block data comparisons (DBKCMP <input type="checkbox"/> ,DBKCMP <input type="checkbox"/> P).....	6 - 18
6.2	<b>Arithmetic Operation Instructions</b> .....	6 - 22
6.2.1	BIN 16-bit addition and subtraction operations (+(P),-(P)) .....	6 - 22
6.2.2	BIN 32-bit addition and subtraction operations (D+(P),D-(P)) .....	6 - 26
6.2.3	BIN 16-bit multiplication and division operations (*(P),/(P)).....	6 - 30
6.2.4	BIN 32-bit multiplication and division operations (D*(P),D/(P)) .....	6 - 32
6.2.5	BCD 4-digit addition and subtraction operations (B+(P),B-(P)) .....	6 - 34
6.2.6	BCD 8-digit addition and subtraction operations (DB+(P),DB-(P)).....	6 - 38
6.2.7	BCD 4-digit multiplication and division operations (B*(P),B/(P)) .....	6 - 42
6.2.8	BCD 8-digit multiplication and division operations (DB*(P),DB/(P)) .....	6 - 44
6.2.9	Addition and subtraction of floating decimal point data (Single precision) (E+(P),E-(P)) .....	6 - 46

6.2.10	Addition and subtraction of floating decimal point data (Double precision) (ED+(P),ED-(P)) .....	6 - 50
6.2.11	Multiplication and division of floating decimal point data (Single precision) (E*(P),E/(P)) .....	6 - 54
6.2.12	Multiplication and division of floating decimal point data (Double precision) (ED*(P),ED/(P)) .....	6 - 56
6.2.13	Block addition and subtraction (BK+(P),BK-(P)).....	6 - 59
6.2.14	BIN 32-bit data block addition and subtraction operations (DBK+(P),DBK-(P)) .....	6 - 62
6.2.15	Linking character strings (\$+(P)) .....	6 - 65
6.2.16	Incrementing and decrementing 16-bit BIN data (INC(P),DEC(P)) .....	6 - 69
6.2.17	Incrementing and decrementing 32-bit BIN data (DINC(P),DDEC(P)) .....	6 - 71
<b>6.3</b>	<b>Data conversion instructions</b> .....	<b>6 - 73</b>
6.3.1	Conversion from BIN data to 4-digit and 8-digit BCD (BCD(P),DBCD(P)) .....	6 - 73
6.3.2	Conversion from BCD 4-digit and 8-digit data to BIN data (BIN(P),DBIN(P)) .....	6 - 75
6.3.3	Conversion from BIN 16 and 32-bit data to floating decimal point (Single precision) (FLT(P),DFLT(P)) .....	6 - 78
6.3.4	Conversion from BIN 16 and 32-bit data to floating decimal point (Double precision) (FLTD(P),DFLTD(P)).....	6 - 81
6.3.5	Conversion from floating decimal point data to BIN16- and 32-bit data (Single precision) (INT(P),DINT(P)) .....	6 - 83
6.3.6	Conversion from floating decimal point data to BIN16- and 32-bit data (Double precision) (INTD(P),DINTD(P)).....	6 - 86
6.3.7	Conversion from BIN 16-bit to BIN 32-bit data (DBL(P)).....	6 - 88
6.3.8	Conversion from BIN 32-bit to BIN 16-bit data (WORD(P)).....	6 - 89
6.3.9	Conversion from BIN 16 and 32-bit data to Gray code (GRY(P),DGRY(P)) .....	6 - 90
6.3.10	Conversion of Gray code to BIN 16 and 32-bit data (GBIN(P),DGBIN(P)).....	6 - 92
6.3.11	Complement of 2 of BIN 16- and 32-bit data (sign reversal) (NEG(P),DNEG(P)) .....	6 - 94
6.3.12	Floating-point sign inversion (Single precision) (ENEG(P)) .....	6 - 96
6.3.13	Floating-point sign inversion (Double precision) (EDNEG(P)) .....	6 - 97
6.3.14	Conversion from block BIN 16-bit data to BCD 4-digit data (BKBCD(P)).....	6 - 98
6.3.15	Conversion from block BCD 4-digit data to block BIN 16-bit data (BKBIN(P)).....	6 - 100
6.3.16	Single precision to Double precision conversion (ECON(P)) .....	6 - 102
6.3.17	Double precision to Single precision conversion (EDCON(P)).....	6 - 104
<b>6.4</b>	<b>Data Transfer Instructions</b> .....	<b>6 - 106</b>
6.4.1	16-bit and 32-bit data transfers (MOV(P),DMOV(P)).....	6 - 106
6.4.2	Floating-point data transfer (Single precision) (EMOV(P)) .....	6 - 108
6.4.3	Floating-point data transfer (Double precision) (EDMOV(P)) .....	6 - 110
6.4.4	Character string transfers (\$MOV(P)).....	6 - 112
6.4.5	16-bit and 32-bit negation transfers (CML(P),DCML(P)).....	6 - 114
6.4.6	Block 16-bit data transfers (BMOV(P)) .....	6 - 117
6.4.7	Identical 16-bit data block transfers (FMOV(P)) .....	6 - 120
6.4.8	Identical 32-bit data block transfers (DFMOV(P)).....	6 - 122
6.4.9	16-bit and 32-bit data exchanges (XCH(P),DXCH(P)) .....	6 - 124
6.4.10	Block 16-bit data exchanges (BXCH(P)) .....	6 - 126
6.4.11	Upper and lower byte exchanges (SWAP(P)) .....	6 - 128
<b>6.5</b>	<b>Program Branch Instructions</b> .....	<b>6 - 129</b>
6.5.1	Pointer branch instructions (CJ,SCJ,JMP) .....	6 - 129
6.5.2	Jump to END (GOEND).....	6 - 132

6.6	Program Execution Control Instructions	6 - 133
6.6.1	Interrupt disable/enable instructions, interrupt program mask (DI,EI,IMASK) .....	6 - 133
6.6.2	Recovery from interrupt programs (IRET) .....	6 - 139
6.7	I/O Refresh Instructions	6 - 141
6.7.1	I/O refresh (RFS(P)) .....	6 - 141
6.8	Other Convenient Instructions	6 - 143
6.8.1	Counter 1-phase input up or down (UDCNT1) .....	6 - 143
6.8.2	Counter 2-phase input up or down (UDCNT2) .....	6 - 146
6.8.3	Teaching timer (TTMR) .....	6 - 149
6.8.4	Special function timer (STMR).....	6 - 151
6.8.5	Rotary table shortest direction control (ROTC) .....	6 - 154
6.8.6	Ramp signal (RAMP).....	6 - 157
6.8.7	Pulse density measurement (SPD) .....	6 - 160
6.8.8	Fixed cycle pulse output (PLSY) .....	6 - 162
6.8.9	Pulse width modulation (PWM) .....	6 - 164
6.8.10	Matrix input (MTR).....	6 - 166

<b>7. APPLICATION INSTRUCTIONS</b>	<b>7 - 1 to 7 - 452</b>
------------------------------------	-------------------------

7.1	Logical operation instructions	7 - 2
7.1.1	Logical products with 16-bit and 32-bit data (WAND(P),DAND(P)).....	7 - 3
7.1.2	Block logical products (BKAND(P)) .....	7 - 9
7.1.3	Logical sums of 16-bit and 32-bit data (WOR(P),DOR(P)).....	7 - 11
7.1.4	Block logical sum operations (BKOR(P)).....	7 - 17
7.1.5	16-bit and 32-bit exclusive OR operations (WXOR(P),DXOR(P)).....	7 - 19
7.1.6	Block exclusive OR operations (BKXOR(P)).....	7 - 25
7.1.7	16-bit and 32-bit data exclusive NOR operations (WXNR(P),DXNR(P)).....	7 - 27
7.1.8	Block exclusive NOR operations (BKXNR(P)).....	7 - 33
7.2	Rotation instruction	7 - 35
7.2.1	Right rotation of 16-bit data (ROR(P),RCR(P)) .....	7 - 35
7.2.2	Left rotation of 16-bit data (ROL(P),RCL(P)) .....	7 - 38
7.2.3	Right rotation of 32-bit data (DROR(P),DRCR(P)) .....	7 - 41
7.2.4	Left rotation of 32-bit data (DROL(P),DRCL(P)).....	7 - 44
7.3	Shift instruction	7 - 46
7.3.1	n-bit shift to right or left of 16-bit data (SFR(P),SFL(P)) .....	7 - 46
7.3.2	1-bit shift to right or left of n-bit data (BSFR(P),BSFL(P)) .....	7 - 49
7.3.3	n-bit shift to right or left of n-bit data (SFTBR(P),SFTBL(P)) .....	7 - 51
7.3.4	1-word shift to right or left of n-word data (DSFR(P),DSFL(P)).....	7 - 54
7.3.5	n-bit shift to right or left of n-word data (SFTWR(P),SFTWL(P)) .....	7 - 56
7.4	Bit processing instructions	7 - 59
7.4.1	Bit set and reset for word devices (BSET(P),BRST(P)) .....	7 - 59
7.4.2	Bit tests (TEST(P),DTEST(P)).....	7 - 61
7.4.3	Batch reset of bit devices (BKRST(P)) .....	7 - 64
7.5	Data processing instructions	7 - 66
7.5.1	16-bit and 32-bit data searches (SER(P),DSER(P)).....	7 - 66
7.5.2	16-bit and 32-bit data checks (SUM(P),DSUM(P)).....	7 - 69
7.5.3	Decoding from 8 to 256 bits (DECO(P)) .....	7 - 71
7.5.4	Encoding from 256 to 8 bits (ENCO(P)) .....	7 - 73

7.5.5	7-segment decode (SEG(P)).....	7 - 75
7.5.6	4-bit dissociation of 16-bit data (DIS(P)).....	7 - 77
7.5.7	4-bit linking of 16-bit data (UNI(P)).....	7 - 79
7.5.8	Dissociation or linking of random data (NDIS(P),NUNI(P)).....	7 - 81
7.5.9	Data dissociation and linking in byte units (WTOB(P),BTOW(P)).....	7 - 85
7.5.10	Maximum value search for 16- and 32-bit data (MAX(P),DMAX(P)).....	7 - 89
7.5.11	Minimum value search for 16- and 32-bit data (MIN(P),DMIN(P)).....	7 - 92
7.5.12	BIN 16 and 32 bits data sort operations (SORT,DSORT).....	7 - 95
7.5.13	Calculation of totals for 16-bit data (WSUM(P)).....	7 - 99
7.5.14	Calculation of totals for 32-bit data (DWSUM(P)).....	7 - 101
7.5.15	Calculation of averages for 16-bit or 32-bit data (MEAN(P),DMEAN(P)).....	7 - 103
<b>7.6</b>	<b>Structure creation instructions</b>	<b>7 - 105</b>
7.6.1	FOR to NEXT instruction loop (FOR,NEXT).....	7 - 105
7.6.2	Forced end of FOR to NEXT instruction loop (BREAK(P)).....	7 - 108
7.6.3	Subroutine program calls (CALL(P)).....	7 - 110
7.6.4	Return from subroutine programs (RET).....	7 - 115
7.6.5	Subroutine program output OFF calls (FCALL(P)).....	7 - 116
7.6.6	Subroutine calls between program files (ECALL(P)).....	7 - 120
7.6.7	Subroutine output OFF calls between program files (EFCALL(P)).....	7 - 125
7.6.8	Subroutine program call (XCALL).....	7 - 129
7.6.9	Refresh instruction (COM).....	7 - 134
7.6.10	Select Refresh Instruction (COM).....	7 - 137
7.6.11	Select Refresh Instruction (CCOM).....	7 - 141
7.6.12	Index modification of entire ladder (IX,IXEND).....	7 - 144
7.6.13	Designation of modification values in index modification of entire ladders (IXDEV,IXSET).....	7 - 148
<b>7.7</b>	<b>Data Table Operation Instructions</b>	<b>7 - 151</b>
7.7.1	Writing data to the data table (FIFW(P)).....	7 - 151
7.7.2	Reading oldest data from tables (FIFR(P)).....	7 - 153
7.7.3	Reading newest data from data tables (FPOP(P)).....	7 - 155
7.7.4	Deleting and inserting data from and in data tables (FDEL(P),FINS(P)).....	7 - 157
<b>7.8</b>	<b>Buffer memory access instruction</b>	<b>7 - 160</b>
7.8.1	Reading 1/2-word data from the intelligent function module (FROM(P),DFRO(P)).....	7 - 160
7.8.2	Writing 1/2-word data to intelligent function module (TO(P),DTO(P)).....	7 - 163
<b>7.9</b>	<b>Display instructions</b>	<b>7 - 166</b>
7.9.1	Print ASCII code instruction (PR).....	7 - 166
7.9.2	Print comment instruction (PRC).....	7 - 169
7.9.3	Error display and annunciator reset instruction (LEDR).....	7 - 172
<b>7.10</b>	<b>Debugging and failure diagnosis instructions</b>	<b>7 - 175</b>
7.10.1	Special format failure checks (CHKST,CHK).....	7 - 175
7.10.2	Changing check format of CHK instruction (CHKCIR,CHKEND).....	7 - 179
<b>7.11</b>	<b>Character string processing instructions</b>	<b>7 - 183</b>
7.11.1	Conversion from BIN 16-bit or 32-bit to decimal ASCII (BINDA(P),DBINDA(P)).....	7 - 183
7.11.2	Conversion from BIN 16-bit or 32-bit data to hexadecimal ASCII (BINHA(P),DBINHA(P)).....	7 - 186

7.11.3	Conversion from BCD 4-digit and 8-digit to decimal ASCII data (BCDDA(P),DBCDDA(P)).....	7 - 189
7.11.4	Conversion from decimal ASCII to BIN 16-bit and 32-bit data (DABIN(P),DDABIN(P)).....	7 - 192
7.11.5	Conversion from hexadecimal ASCII to BIN 16-bit and 32-bit data (HABIN(P),DHABIN(P)).....	7 - 195
7.11.6	Conversion from decimal ASCII to BCD 4-digit or 8-digit data (DABCD(P),DDABCD(P)).....	7 - 198
7.11.7	Reading device comment data (COMRD(P)) .....	7 - 201
7.11.8	Character string length detection (LEN(P)) .....	7 - 204
7.11.9	Conversion from BIN 16-bit or 32-bit to character string (STR(P),DSTR(P)) .....	7 - 206
7.11.10	Conversion from character string to BIN 16-bit or 32-bit data (VAL(P),DVAL(P)) ....	7 - 212
7.11.11	Conversion from floating decimal point to character string data (ESTR(P)).....	7 - 217
7.11.12	Conversion from character string to floating decimal point data (EVAL(P)) .....	7 - 224
7.11.13	Conversion from hexadecimal BIN to ASCII (ASC(P)) .....	7 - 228
7.11.14	Conversion from ASCII to hexadecimal BIN (HEX(P)).....	7 - 230
7.11.15	Extracting character string data from the right or left (RIGHT(P),LEFT(P)).....	7 - 232
7.11.16	Random selection from and replacement in character strings (MIDR(P),MIDW(P)) .....	7 - 235
7.11.17	Character string search (INSTR(P)) .....	7 - 239
7.11.18	Insertion of character string (STRINS(P)).....	7 - 241
7.11.19	Deletion of character string (STRDEL(P)) .....	7 - 243
7.11.20	Floating decimal point to BCD (EMOD(P)).....	7 - 245
7.11.21	From BCD format data to floating decimal point (EREXP(P)) .....	7 - 248
7.12	Special function instructions	7 - 250
7.12.1	SIN operation on floating-point data (Single precision) (SIN(P)).....	7 - 250
7.12.2	SIN operation on floating-point data (Double precision) (SIND(P)).....	7 - 252
7.12.3	COS operation on floating-point data (Single precision) (COS(P)) .....	7 - 254
7.12.4	COS operation on floating-point data (Double precision) (COSD(P)) .....	7 - 256
7.12.5	TAN operation on floating-point data (Single precision) (TAN(P)).....	7 - 258
7.12.6	TAN operation on floating-point data (Double precision) (TAND(P)).....	7 - 260
7.12.7	$\text{SIN}^{-1}$ operation on floating point data (Single precision) (ASIN(P)) .....	7 - 262
7.12.8	$\text{SIN}^{-1}$ operation on floating-point data (Double precision) (ASIND(P)) .....	7 - 265
7.12.9	$\text{COS}^{-1}$ operation on floating-point data (Single precision) (ACOS(P)) .....	7 - 267
7.12.10	$\text{COS}^{-1}$ operation on floating-point data (Double precision) (ACOSD(P)) .....	7 - 269
7.12.11	$\text{TAN}^{-1}$ operation on floating-point data (Single precision) (ATAN(P)).....	7 - 271
7.12.12	$\text{TAN}^{-1}$ operation on floating-point data (Double precision) (ATAND(P)).....	7 - 273
7.12.13	Conversion from floating-point angle to radian (Single precision) (RAD(P)) .....	7 - 275
7.12.14	Conversion from floating-point angle to radian (Double precision) (RADD(P)) .....	7 - 277
7.12.15	Conversion from floating-point radian to angle (Single precision) (DEG(P)).....	7 - 279
7.12.16	Conversion from floating-point radian to angle (Double precision) (DEGD(P)) .....	7 - 281
7.12.17	Exponentiation operation on floating-point data (Single precision) (POW(P)).....	7 - 283
7.12.18	Exponentiation operation on floating-point data (Single precision) (POWD(P)).....	7 - 285
7.12.19	Square root operation for floating-point data (Single precision) (SQR(P)) .....	7 - 287
7.12.20	Square root operation for floating-point data (Double precision) (SQRD(P)) .....	7 - 289
7.12.21	Exponent operation on floating-point data (Single precision) (EXP(P)).....	7 - 291
7.12.22	Exponent operation on floating-point data (Double precision) (EXPD(P)).....	7 - 294
7.12.23	Natural logarithm operation on floating-point data (Single precision) (LOG(P)) .....	7 - 296
7.12.24	Natural logarithm operation on floating-point data (Double precision) (LOGD(P))... 7 - 298	



7.12.25	Common logarithm operation on floating-point data (Single precision) (LOG10(P)).....	7 - 300
7.12.26	Common logarithm operation on floating-point data (Double precision) (LOG10D(P)).....	7 - 302
7.12.27	Random number generation and series updates (RND(P),SRND(P)) .....	7 - 304
7.12.28	BCD 4-digit and 8-digit square roots (BSQR(P),BDSQR(P)) .....	7 - 306
7.12.29	BCD type SIN operation (BSIN(P)).....	7 - 309
7.12.30	BCD type COS operations (BCOS(P)) .....	7 - 311
7.12.31	BCD type TAN operation (BTAN(P)) .....	7 - 313
7.12.32	BCD type SIN <sup>-1</sup> operations (BASIN(P)).....	7 - 315
7.12.33	BCD type COS <sup>-1</sup> operation (BACOS(P)).....	7 - 317
7.12.34	BCD type TAN <sup>-1</sup> operations (BATAN(P)) .....	7 - 319
7.13	Data Control Instructions	7 - 321
7.13.1	Upper and lower limit controls for BIN 16-bit and BIN 32-bit data (LIMIT(P),DLIMIT(P)) .....	7 - 321
7.13.2	BIN 16-bit and 32-bit dead band controls (BAND(P),DBAND(P)) .....	7 - 324
7.13.3	Zone control for BIN 16-bit and BIN 32-bit data (ZONE(P),DZONE(P)).....	7 - 327
7.13.4	Scaling (Point-by-point coordinate data) (SCL(P),DSCL(P)).....	7 - 330
7.13.5	Scaling (Point-by-point coordinate data) (SCL2(P),DSCL2(P)).....	7 - 334
7.14	File register switching instructions	7 - 337
7.14.1	Switching file register numbers (RSET(P)).....	7 - 337
7.14.2	Setting files for file register use (QDRSET(P)) .....	7 - 339
7.14.3	File setting for comments (QCDSSET(P)) .....	7 - 342
7.15	Clock instructions	7 - 344
7.15.1	Reading clock data (DATERD(P)).....	7 - 344
7.15.2	Writing clock data (DATEWR(P)) .....	7 - 346
7.15.3	Clock data addition operation (DATE+(P)).....	7 - 348
7.15.4	Clock data subtraction operation (DATE-(P)).....	7 - 350
7.15.5	Time data conversion (from Hour/Minute/Second to Second) (SECOND(P)).....	7 - 352
7.15.6	Time data conversion (from Second to Hour/Minute/Second ) (HOUR(P)).....	7 - 354
7.15.7	Date comparison (DT=,DT<>,DT>,DT<=,DT<,DT>=) .....	7 - 356
7.15.8	Clock comparison (TM=,TM<>,TM>,TM<=,TM<,TM>=).....	7 - 361
7.16	Expansion Clock Instructions	7 - 366
7.16.1	Reading expansion clock data (S(P).DATERD) .....	7 - 366
7.16.2	Expansion clock data addition operation (S(P).DATE+).....	7 - 369
7.16.3	Expansion clock data subtraction operation (S(P).DATE-).....	7 - 372
7.17	Program control instructions	7 - 375
7.17.1	Program standby instruction (PSTOP(P)) .....	7 - 377
7.17.2	Program output OFF standby instruction (POFF(P)).....	7 - 378
7.17.3	Program scan execution registration instruction (PSCAN(P)) .....	7 - 380
7.17.4	Program low speed execution registration instruction (PLOW(P)) .....	7 - 382
7.17.5	Program execution status check instruction (PCHK).....	7 - 384
7.18	Other instructions	7 - 386
7.18.1	Resetting watchdog timer (WDT(P)).....	7 - 386
7.18.2	Timing pulse generation (DUTY) .....	7 - 388
7.18.3	Time check instruction (TIMCHK).....	7 - 390
7.18.4	Direct 1-byte read from file register (ZRRDB(P)).....	7 - 391

7.18.5	File register direct 1-byte write (ZRWRB(P))	7 - 393
7.18.6	Indirect address read operations (ADRSET(P))	7 - 395
7.18.7	Numerical key input from keyboard (KEY)	7 - 396
7.18.8	Batch save or recovery of index register (ZPUSH(P),ZPOP(P))	7 - 400
7.18.9	Reading Module Information (UNIRD(P))	7 - 402
7.18.10	Reading module model name(TYPERD(P))	7 - 406
7.18.11	Trace Set/Reset (TRACE,TRACER)	7 - 411
7.18.12	Writing Data to Designated File (SP.FWRITE)	7 - 413
7.18.13	Reading Data from Designated File (SP.FREAD)	7 - 424
7.18.14	Writing Data to Standard ROM (SP.DEVST)	7 - 436
7.18.15	Read Data from Standard ROM (S(P).DEVLD)	7 - 438
7.18.16	Load Program from Memory Card (PLOADP)	7 - 440
7.18.17	Unload Program from Program Memory (PUNLOADP)	7 - 443
7.18.18	Load + Unload (PSWAPP)	7 - 445
7.18.19	High-speed Block Transfer of File Register (RBMOV(P))	7 - 448

## Common Instructions 2/2

<b>8.</b>	<b>INSTRUCTIONS FOR DATA LINK</b>	<b>8 - 1 to 8 - 10</b>
8.1	Network refresh instructions	8 - 2
8.1.1	Refresh instruction for the designated module (S(P)/J(P)/G(P).ZCOM)	8 - 2
8.2	Reading/Writing Routing Information	8 - 6
8.2.1	Reading routing information (S(P)/Z(P).RTREAD)	8 - 6
8.2.2	Registering routing information (S(P)/Z(P).RTWRITE)	8 - 8
<b>9.</b>	<b>Multiple CPU dedicated instruction</b>	<b>9 - 1 to 9 - 18</b>
9.1	Writing to the CPU Shared Memory of Host CPU	9 - 2
9.1.1	Write to Host CPU Shared Memory (S(P).TO)	9 - 4
9.1.2	Writing to host station CPU shared memory (TO(P), DTO(P))	9 - 7
9.2	Reading from the CPU Shared Memory of another CPU	9 - 11
9.2.1	Reading from Other CPU Shared Memory (FROM(P), DFRO(P))	9 - 12
<b>10.</b>	<b>QCPU INSTRUCTIONS</b>	<b>10 - 1 to 10 - 20</b>
10.1	Overview	10 - 2
10.2	Writing Devices to Another CPU (D(P).DDWR)	10 - 13
10.3	Reading Devices from Another CPU (D(P).DDRD)	10 - 17
<b>11.</b>	<b>QCPU INSTRUCTIONS</b>	<b>11 - 1 to 11 - 4</b>
11.1	System Switching Instruction (SP.CONTSW)	11 - 2
<b>12.</b>	<b>ERROR CODES</b>	<b>12 - 1 to 12 - 84</b>
12.1	Error Code List	12 - 2
12.1.1	Error codes	12 - 3
12.1.2	Reading an error code	12 - 3
12.1.3	Error code list (1000 to 1999)	12 - 4

12.1.4	Error code list (2000 to 2999)	12 - 16
12.1.5	Error code list (3000 to 3999)	12 - 34
12.1.6	Error code list (4000 to 4999)	12 - 51
12.1.7	Error code list (5000 to 5999)	12 - 66
12.1.8	Error code list (6000 to 6999)	12 - 68
12.1.9	Error code list (7000 to 10000)	12 - 78
12.2	Canceling of Errors	12 - 83

<b>APPENDICES</b>	<b>App - 1 to App - 198</b>
-------------------	-----------------------------

<b>Appendix 1 OPERATION PROCESSING TIME</b>		<b>App - 2</b>
Appendix 1.1	Definition	App - 2
Appendix 1.2	Operation Processing Time of Basic Model QCPU	App - 3
Appendix 1.3	Operation Processing Time of High Performance Model QCPU/Process CPU/ Redundant CPU	App - 21
Appendix 1.4	Operation Processing Time of Universal Model QCPU	App - 50
Appendix 1.4.1	Subset instruction processing time	App - 50
Appendix 1.4.2	Processing time of instructions other than subset instruction	App - 66
<b>Appendix 2 CPU PERFORMANCE COMPARISON</b>		<b>App - 114</b>
Appendix 2.1	Comparison of Q with AnNCP, AnACPU, and AnUCPU	App - 114
Appendix 2.1.1	Usable devices	App - 114
Appendix 2.1.2	I/O control mode	App - 115
Appendix 2.1.3	Data that can be used by instructions	App - 115
Appendix 2.1.4	Timer comparison	App - 116
Appendix 2.1.5	Comparison of counters	App - 117
Appendix 2.1.6	Comparison of display instructions	App - 117
Appendix 2.1.7	Instructions whose designation format has been changed (Except dedicated instructions for AnACPU and AnUCPU)	App - 118
Appendix 2.1.8	AnACPU and AnUCPU dedicated instructions	App - 119
<b>Appendix 3 SPECIAL RELAY LIST</b>		<b>App - 120</b>
<b>Appendix 4 SPECIAL REGISTER LIST</b>		<b>App - 146</b>
<b>Appendix 5 APPLICATION PROGRAM EXAMPLES</b>		<b>App - 198</b>
Appendix 5.1	Concept of Programs which Perform Operations of $X^n$ , $\sqrt[n]{X}$	App - 198

<b>INDEX</b>	<b>Index - 1 to Index - 12</b>
--------------	--------------------------------

# MANUALS

To understand the main specifications, functions, and usage of the CPU module, refer to the basic manuals.  
 Read other manuals as well when using a different type of CPU module and its functions.  
 Order each manual as needed, referring to the following list.

The numbers in the "CPU module" and the respective modules are as follows.

Number	CPU module
1)	Basic model QCPU
2)	High Performance model QCPU
3)	Process CPU
4)	Redundant CPU
5)	Universal model QCPU

○:Basic manual, ●:Other CPU module manuals

Manual name < Manual number (model code) >	Description	CPU module				
		1)	2)	3)	4)	5)
<b>■User's manual</b>						
QCPU User's Manual (Hardware design, Maintenance and Inspection) < SH-080483ENG (13JR73) >	Specifications of the hardware (CPU modules, power supply modules, base units, extension cables, and memory cards), system maintenance and inspection, troubleshooting, and error codes	●	●	●	●	●
QnUCPU User's Manual (Function Explanation, Program Fundamentals) < SH-080807ENG (13JZ27) >	Functions, methods, and devices for programming					●
Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals) < SH-080808ENG (13JZ28) >	Functions, methods, and devices for programming	●	●	●	●	
QnUCPU User's Manual (Communication via Built-in Ethernet Port) < SH-080811ENG (13JZ29) >	Functions for the communication via built-in Ethernet port of the CPU module					○
<b>■Programming Manual</b>						
QCPU Programming Manual (Common Instructions) < SH-080809ENG (13JW10) >	How to use sequence instructions, basic instructions, and application instructions	●	●	●	●	●
QCPU (Q Mode)/QnACPU Programming Manual (SFC) < SH-080041 (13JF60) >	System configuration, performance specifications, functions, programming, debugging, and error codes for SFC (MELSAP3) programs	○	○	○	○	○
QCPU (Q Mode) Programming Manual (MELSAP-L) < SH-080072 (13JC03) >	Programming methods, specifications, and functions for SFC (MELSAP-L) programs	○	○	○	○	○
QCPU (Q Mode) Programming Manual (Structured Text) < SH-080366E (13JF68) >	Programming methods using structured languages	○	○	○	○	○
QCPU (Q Mode) / QnACPU Programming Manual (PID Control Instructions) < SH-080040 (13JF59) >	Dedicated instructions for PID control	○	○		○	○
QnPH/QnPRHCPU Programming Manual (Process Control Instructions) < SH-080316E (13JF59) >	Describes the dedicated instructions for performing process control.			○	○	

Related Manuals

Manual name < Manual number (model code) >	Description
CC-Link IE Controller Network Reference Manual < SH-080668ENG (13JV16) >	Specifications, procedures and settings before system operation, parameter setting, programming, and troubleshooting of the CC-Link IE controller network module
Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network) < SH-080049 (13JF92) >	Explains the specifications for a MELSECNET/H network system for PLC to PLC network. It explains the procedures and settings up to operation, setting the parameters, programming and troubleshooting.
Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network) < SH-080124 (13JF96) >	Explains the specifications for a MELSECNET/H network system for remote I/O network. It explains the procedures and settings up to operation, setting the parameters, programming and troubleshooting.
Type MELSECNET, MELSECNET/B Data Link System Reference Manual < IB-66530 (13JF70) >	Describes the general concept, specifications, and part names and settings for MELSECNET (II) and MELSECNET/B.
Q Corresponding Ethernet Interface Module User's Manual (Application) < SH-080010 (13JF70) >	Describes various functions of the Ethernet module: e-mail function, PLC CPU status monitoring, communication via MELSECNET/H or MELSECNET/10 network system, communication using data link instructions, file transfer (using FTP) and other functions.



# 8

# INSTRUCTIONS FOR DATA LINK

Category	Processing Details	Reference section
Network refresh instructions	Refreshes the specified network module.	Section 8.1
Routing information read/write instructions	Reading the data specified by routing parameters.	Section 8.2.1
	Writing routing data to the area specified by routing parameters.	Section 8.2.2

## Remark

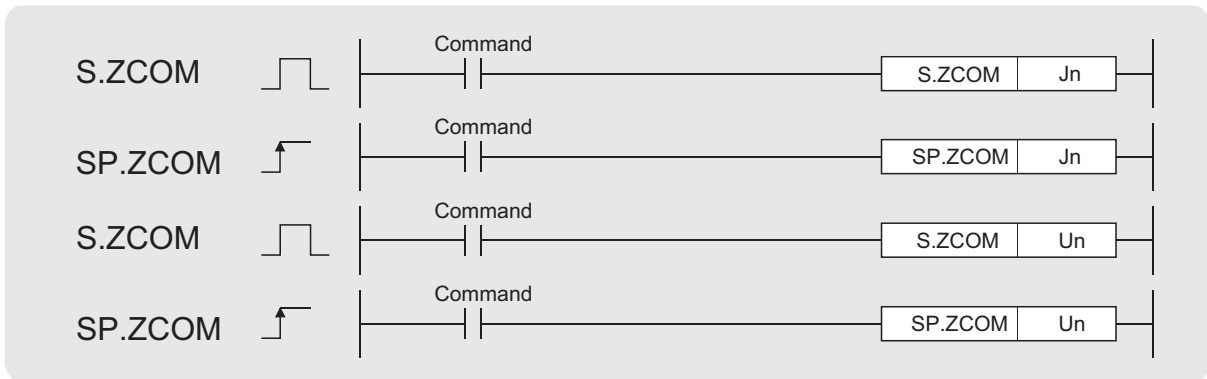
In this chapter, instruction names are abbreviated as follows if not specified particularly.

- S(P)/J(P)/G(P).ZCOM → ZCOM
- S(P)/Z(P).RTWRITE → RTWRITE
- S(P)/Z(P).RTREAD → RTREAD

# 8.1 Network refresh instructions

## 8.1.1 Refresh instruction for the designated module (S(P)/J(P)/G(P).ZCOM)

Basic High performance Process Redundant Universal



Jn : Network No. of host station (BIN 16 bits)

Un : Head I/O number of host station network module (BIN 16 bits)

Setting Data	Internal Devices		R, ZR	Jn		Un	Zn	Constants	Other
	Bit	Word		Bit	Word				
—									

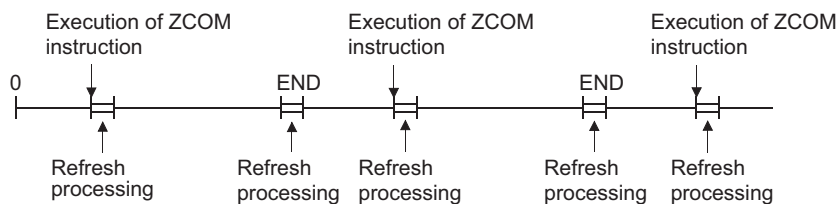
The ZCOM instruction is used to perform refresh at any timing during execution of a sequence program.

The targets of refresh performed by the ZCOM instruction are indicated below.

- Refresh of CC-Link IE controller network (when refresh parameters are set)
- Refresh of MELSECNET/H (when refresh parameters are set)
- Auto refresh of CC-Link (when refresh device is set)
- Auto refresh of intelligent function module (when auto refresh is set)

### ★ Function

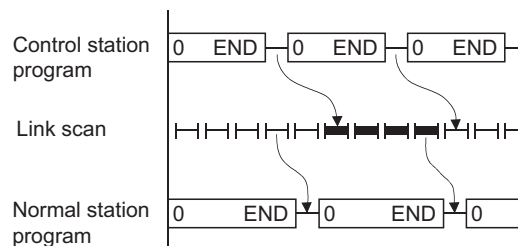
- (1) When the ZCOM instruction is executed, the CPU module temporarily suspends processing of the sequence program and conducts refresh processing of the network modules designated by Jn/Un.



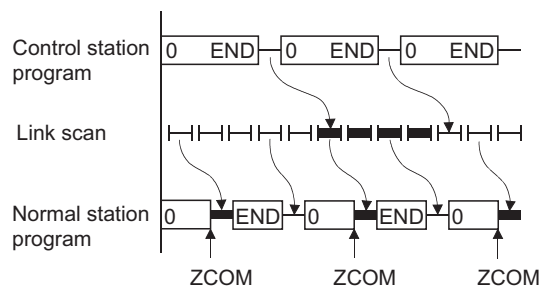


- (2) The ZCOM instruction does not perform the following processing.
- Communication processing between CPU module and programming tool
  - Monitor processing of other station
  - Read processing of buffer memory of other intelligent function module by serial communication module.
  - Low-speed cyclic data transmission of MELSECNET/H
- (3) PLC to PLC network<sup>\*1</sup>
- When the scan time for the sequence program of host station is longer than the scan time for the other station, the ZCOM instruction is used to ensure the data reception from the other station.

(1) Example of data communications when the ZCOM instruction is not used



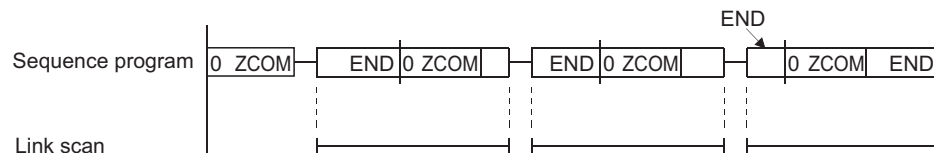
(2) Example of data communications when the ZCOM instruction is used



For details of the transmission delay time on the PLC to PLC network<sup>\*1</sup>, refer to the manual below:

- CC-Link IE Controller Network Reference Manual
- Q Corresponding MELSECNET/H Network System Reference Manual (PLC to PLC network)

- When the link scan time is longer than the sequence program scan time, data communications will not be faster even if the ZCOM instruction is used.



\*1 : Controller network in CC-Link IE controller network.

(4) Remote I/O network

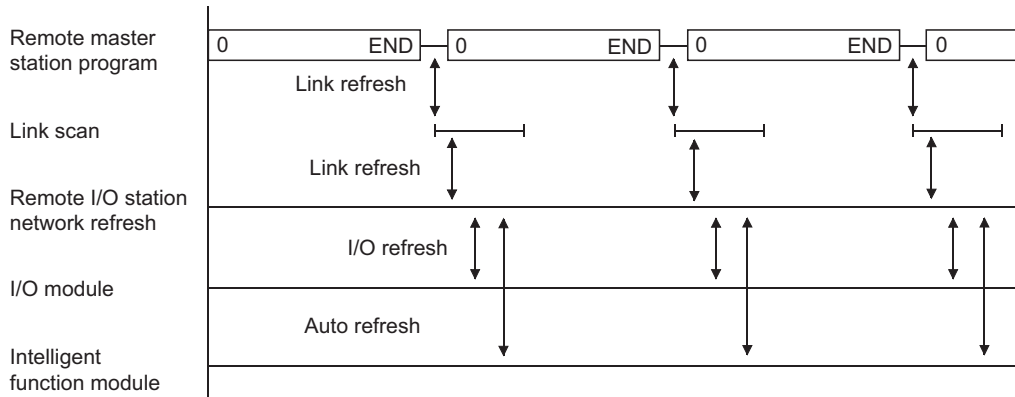
The link refresh of the remote master station is performed by the "END processing" of the CPU module.

Since link scan is performed at completion of link refresh, link scan 'synchronizes' with the program of the CPU module.

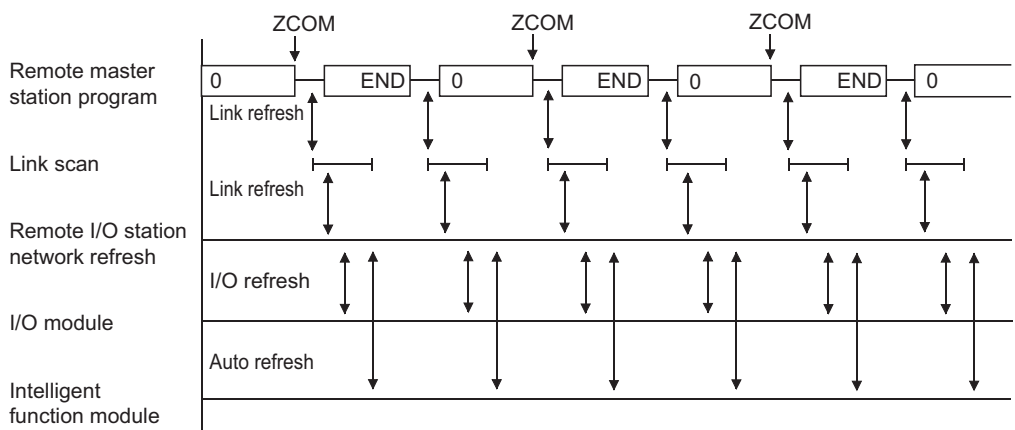
When the ZCOM instruction is used at the remote master station, link refresh is performed at the point of ZCOM instruction execution, and link scan is performed at completion of link refresh.

Hence, use of the ZCOM instruction at the remote master station speeds up send/receive processing to/from the remote I/O station.

(1) When the ZCOM instruction is not used



(2) When the ZCOM instruction is used



For details of the transmission delay time on the remote I/O network, refer to the manual below:

- Q Corresponding MELSECNET/H Network System Reference Manual (Remote I/O network)

(5) The ZCOM instruction can be used as many times as desired in sequence programs. However, note that each execution of a refresh operation will lengthen the sequence program scan time by the amount of time required for the refresh operation.

- (6) Designating "Un" in the argument enables the target designation of the intelligent function as well as the network modules.  
In this case, the auto refresh is performed for the buffer memory of the intelligent function modules. (It replaces the FROM/TO instructions.)
- (7) Only with the universal model QCPU, interruption of processing is enabled during the execution of the ZCOM instruction. However, when refresh data are used in an interrupted program, the data can split.

### POINT

1. The ZCOM instruction cannot be used in a fixed cycle execution type program or interrupt program.
2. The Redundant CPU has restrictions on use of the ZCOM instruction.  
Refer to the manual below for details.
  - QnPRHCPU User's Manual (Redundant System)

## ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- When the specified network number is not connected to the host station  
(Error code: 4102)
  - When the module specified with the head I/O number is not a network module or link module (Except Universal model QCPU)  
(Error code: 2111)
  - When the module specified with the head I/O number is not a network module or link module (Only Universal model QCPU)  
(Error code: 4102)

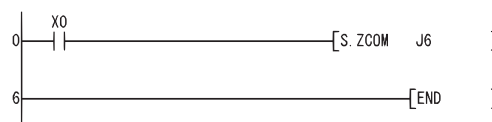
### POINT

To conduct only communication with peripheral device, use the COM instruction (refer to Section 7.6.9, 9.1).

## Program Example

- (1) The following program conducts a link refresh for the network module of network No. 6 while X0 is ON.

[Ladder Mode]

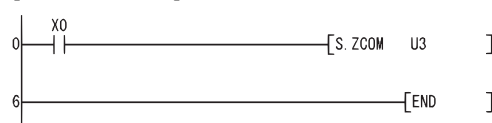


[List Mode]

Step	Instruction	Device
0	LD	X0
1	S. ZCOM	J6
6	END	

- (2) The following program conducts a link refresh for the network module mounted to the position whose head I/O number is a X/Y30 to X/Y4F while X0 is ON.

[Ladder Mode]

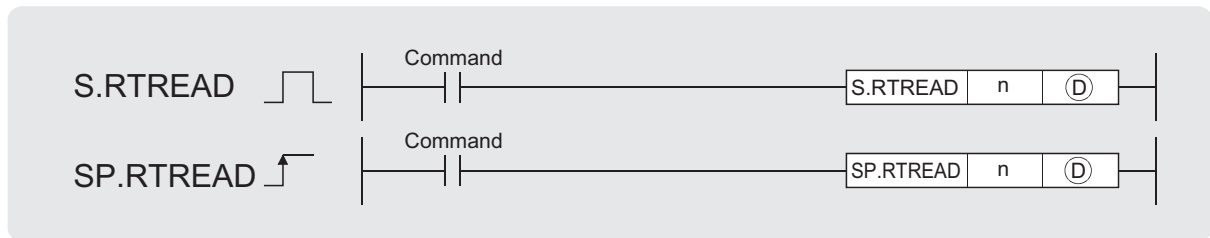


[List Mode]

Step	Instruction	Device
0	LD	X0
1	S. ZCOM	U3
6	END	

## 8.2 Reading/Writing Routing Information

### 8.2.1 Reading routing information (S(P)/Z(P).RTREAD)



n : Transfer destination network No. (1 to 239) (BIN 16 bits)

Ⓣ : Head number of the devices that stores the read data (Device name)

Setting Data	Internal Devices		R, ZR	JAD		UAG	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n	○	○				—		○	—
Ⓣ	—	○				—		—	—

### ★ Function

- (1) Reads data from transfer destination network number specified by n, using routing information set by the routing parameters, and stores it into the area starting from Ⓣ.
- (2) If no data for the transfer destination network number specified by n is set at the routing parameters, stores 0 into the area starting from Ⓣ.
- (3) The contents of the data stored in the area starting from Ⓣ is as indicated below.

(Individual data ranges)

Ⓣ+0	Relay network number	(1 to 239)
+1	Relay station number	(1 to 120)
+2	Dummy	

### ! Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - When data specified by n is other than 1 to 239. (Error code: 4100)
  - The device specified by Ⓣ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

# Program Example

- (1) The following program reads the routing information for the network number specified by D0 when X0 is turned ON.

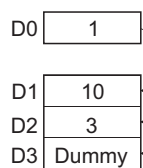
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	S.RTREAD	D0 D1
8	END	

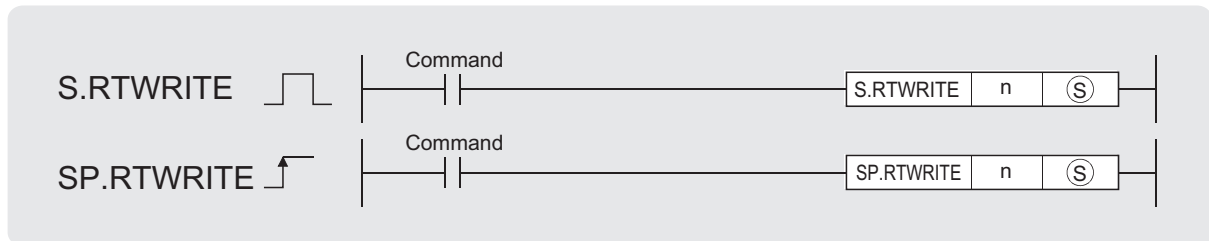
[Operation]



[Content of routing parameter setting]

Transfer destination network number	Relay network number	Relay station number
1	10	3
2	10	2
3	10	1

## 8.2.2 Registering routing information (S(P)/Z(P).RTWRITE)



n : Transfer destination network No. (1 to 239) (BIN 16 bits)

Ⓢ : Head number of the devices where the data to be written is stored (Device name)

Setting Data	Internal Devices		R, ZR	JOG		U, G	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n	<input type="radio"/>	<input type="radio"/>				—		<input type="radio"/>	—
Ⓢ	—	<input type="radio"/>				—		—	—

### ★ Function

- Registers routing data of Ⓢ or later in the area for the transfer destination network number specified by n in the routing parameters.
- The following shows the contents of data to be set at Ⓢ or later.

(Individual data ranges)

Ⓢ+0	Relay network number	(0 to 239)
+1	Relay station number	(0 to 120)
+2	Dummy	

- If data for the transfer destination network number specified by n is set in the routing parameters, it is used to update the data in the area starting from Ⓢ.
- If all data in Ⓢ or later (Ⓢ+0 to Ⓢ+2) is 0, the data for the transfer destination network number specified by n is deleted from the routing parameters.

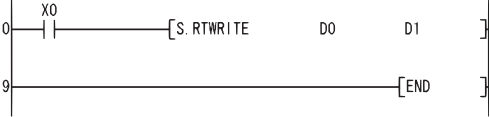
### ! Operation Error

- In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
  - When data specified by n is other than 1 to 239. (Error code: 4100)
  - When the data of Ⓢ or later exceeds each setting ranges. (Error code: 4100)
  - When the total number of routing information registered in the routing parameter of the network parameters and routing information registered with the RTWRITE instruction exceeds 64. (Error code: 4100)
  - The device specified by Ⓢ exceeds the range of the corresponding device. (For the Universal model QCPU only.) (Error code: 4101)

# Program Example

(1) The following program writes the routing information specified by D1 to D3 to the network module of the network number specified by D0 when X0 is turned ON.

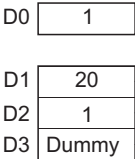
[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	S.RTWRITE	D0 D1
9	END	

[Operation]



[Content of routing parameter setting]

Transfer destination network number	Relay network number	Relay station number
1	20	1
2	10	2
3	10	1





# 9

# MULTIPLE CPU DEDICATED INSTRUCTION

---

Category	Processing Details	Reference section
Writing to the CPU shared memory of host CPU	Writes device data of the host CPU to the CPU shared memory of the host CPU module.	Section 9.1
Reading from the CPU shared memory of another CPU	Reads device data from the CPU shared memory of another CPU module to the host CPU.	Section 9.2

# 9.1 Writing to the CPU Shared Memory of Host CPU

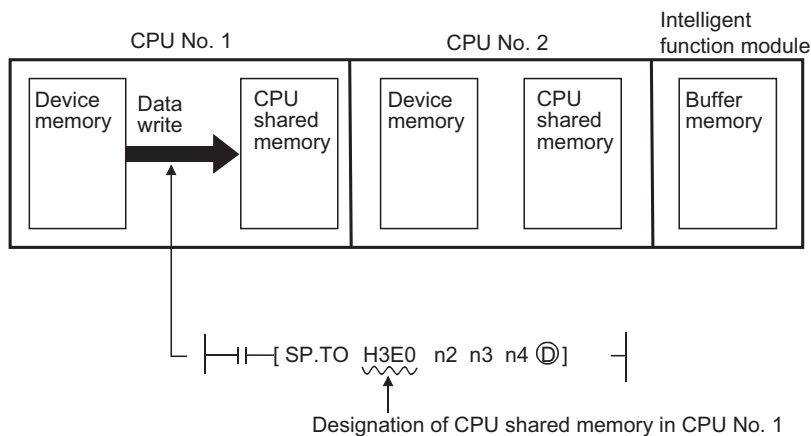
The S.TO or TO instruction is used to write to the CPU shared memory of the host station in the multiple CPU system.

The following table indicates the usability of the S.TO and TO instructions.

CPU Module Type Name		S.TO Instruction	TO Instruction
Basic model QCPU	Q00CPU, Q01CPU	Usable	Usable
High Performance model QCPU	Q02CPU, Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU	Usable	Unusable
Process CPU	Q02PHCPU, Q06PHCPU, Q12PHCPU, Q25PHCPU	Usable	Unusable
Redundant CPU	Q12PRHCPU, Q25PRHCPU	Unusable	Unusable
Universal model QCPU	Q00UCPU, Q01UCPU, Q02UCPU, Q03UDCPU, Q04UDHCPU, Q06UDHCPU, Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU, Q03UDECPU, Q04UDEHCPU, Q06UDEHCPU, Q10UDEHCPU, Q13UDEHCPU, Q20UDEHCPU, Q26UDEHCPU	Usable	Usable

### (1) Operation of S.TO instruction

The S.TO instruction can write data to the CPU shared memory of the host CPU module. The following figure shows the processing performed when the S.TO instruction is executed in CPU No. 1.

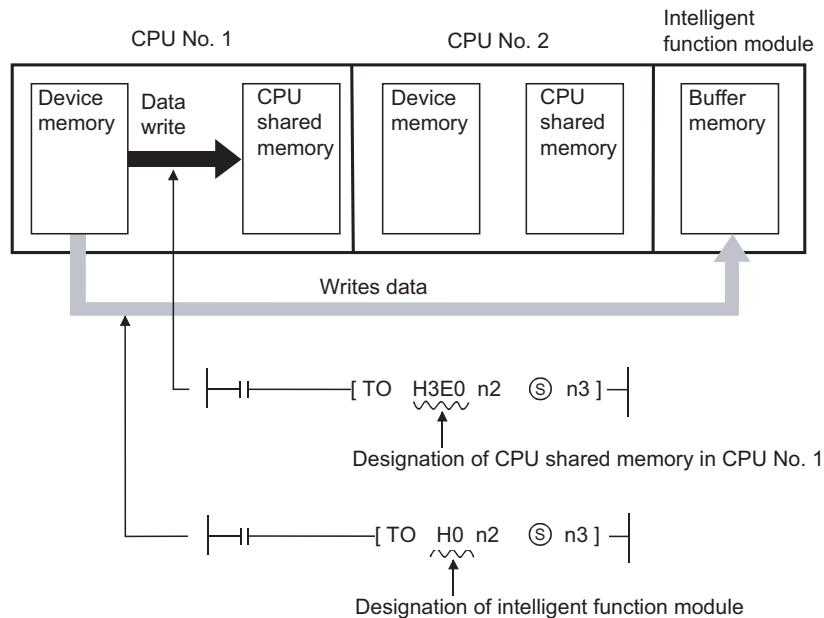


## (2) Operation of the TO instruction

The TO instruction can write device memory data to the following memories.

- CPU shared memory of host CPU module
- Buffer memory of intelligent function module

The following figure shows the processing performed when the TO instruction is executed in CPU No. 1.



### POINT

Both of the S.TO and TO instructions can be used for the Basic model QCPU (Q00CPU or Q01CPU) and Universal model QCPU to write data to the CPU shared memory. However, use of the TO instruction is recommended, since use of S.TO instruction reduces the number of steps and processing time.

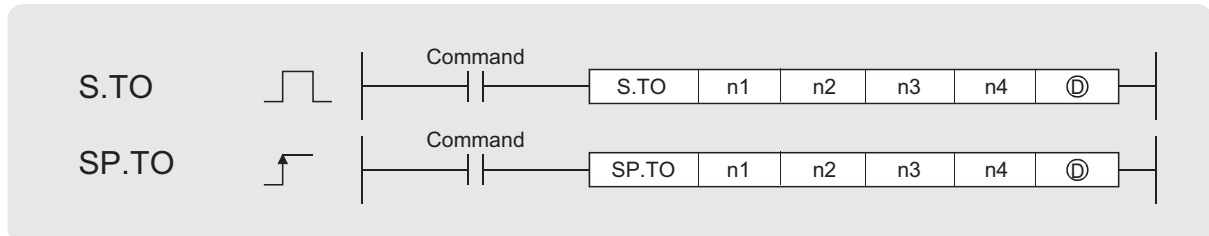
### Remark

Refer to Section 7.8.2 when writing to the buffer memory of the intelligent function module by the TO instruction.

## 9.1.1 Write to Host CPU Shared Memory (S(P).TO)



Basic model QCPU: The first 5 digits of serial No is "04122" or higher.  
 High performance model QCPU: Function version B or later.



- n1 : Head I/O number of the host CPU (BIN 16 bits)  
 n2 : CPU shared memory address of the write destination host CPU (BIN 16 bits)  
 •Basic model QCPU: 0 to 511  
 •High Performance model QCPU, Process CPU, Universal model QCPU: 0 to 4095  
 n3 : Head number of the devices where data to be written is stored (BIN 16 bits)  
 n4 : Number of data blocks to be written (BIN 16 bits)  
 •Basic model QCPU: 1 to 320  
 •High Performance model QCPU, Process CPU: 1 to 256  
 •Universal model QCPU: 1 to 2048

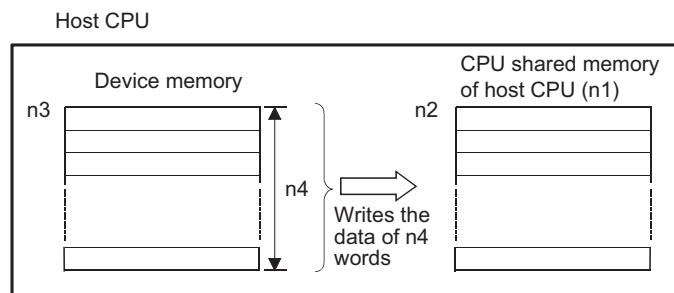
Ⓧ : Device of the host CPU which is turned ON for one scan by the completion of writing (bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
n1	—	○				—		○	—
n2	—	○				—		○	—
n3	—	○				—		—	—
n4	—	○				—		○	—
Ⓧ	○	○				○		—	—

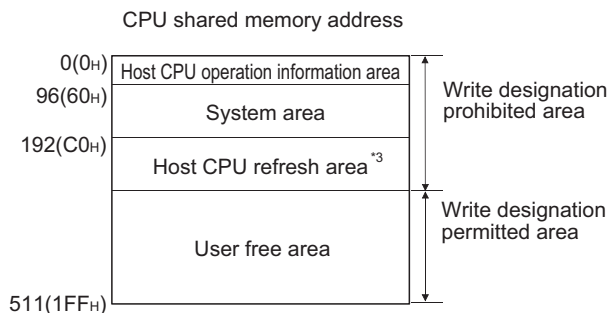
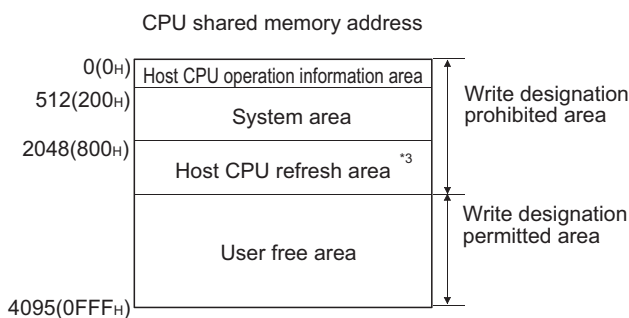
### ★ Function

- (1) Writes device data of words n3 to n4 to the CPU shared memory address specified by n2 of the host CPU module or later address.

When writing is completed, the completion bit specified by Ⓧ turns ON.



## (a) CPU shared memory address of the Basic model QCPU

(b) CPU shared memory address of the High Performance model QCPU, Process CPU and Universal model QCPU\*<sup>4</sup>

\*<sup>3</sup> : Usable as a user free area when auto refresh setting is not made.

In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

\*<sup>4</sup> : Data cannot be written to the multiple CPU high speed transmission area of the Universal model QCPU with the S(P).TO instruction.

- (2) When the number of write points is 0, no processing is performed and the completion device does not turn ON, either.
- (3) The S.TO instruction can be executed once to one scan for each CPU.  
 When execution condition is established at two or more places at the same time, the S.TO instruction executed later is not processed since handshake is established automatically.
- (4) The number of data that can be written varies depending on the target CPU module.

CPU module	Number of Write Points
Basic model QCPU	1 to 320
High Performance model QCPU Process CPU	1 to 256
Universal model QCPU	1 to 2048

### ☒ POINT

Writing data to CPU shared memory can be performed using the intelligent function module device.

For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

## Operation Error

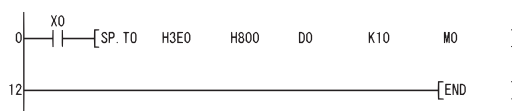
In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- (1) When the specified data is outside the following range (Error code: 4101)
  - When the number of write points (n4) is outside the specified range of the setting data.
  - When the head of the CPU shared memory address (n2) of the write destination host CPU exceeds the CPU shared memory address range
  - When the CPU shared memory address (n2) + the number of write points (n4) of the write destination host CPU exceeds the CPU shared memory address range
  - When the head number of the devices (n3) where the data to be written is stored + the number of write points (n4) exceeds the device range
- (2) When the host CPU operation information area, system area or host CPU refresh area is specified to the CPU shared memory address (n2) of the write destination  
 (High Performance model QCPU, Process CPU) (Error code: 4101)  
 (Basic model QCPU, Universal model QCPU) (Error code: 4111)
- (3) When the head I/O number (n1) of the host CPU is other than that of the host CPU  
 (High Performance model QCPU, Process CPU) (Error code: 2107)  
 (Basic model QCPU, Universal model QCPU) (Error code: 4112)
- (4) No CPU module is installed at the position specified by the head I/O number of the CPU module. (Error code: 2110)
- (5) When the head I/O number (n1) of the host CPU is other than 3E0H/3E1H/3E2H/3E3H (Error code: 4100)
- (6) When the specified instruction is improper (Error code: 4002)
- (7) When the specified number of devices is wrong (Error code: 4003)
- (8) When the unusable device is specified (Error code: 4004)

## Program Example

- (1) The following program stores 10 points of data from D0 into address 800<sub>H</sub> of the CPU shared memory of CPU No. 1 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	SP.T0	H3E0 H800 D0 K10 M0
12	END	

### Remark

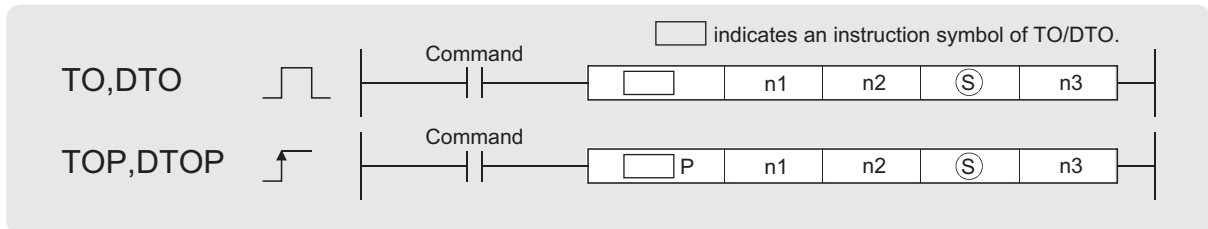
The n1 is specified by the first 3 digits of the hexadecimal 4 digits which represent the head I/O number of the slot mounted to the CPU module.

	CPU Slot	Slot 0	Slot 1	Slot 2
Head I/O number	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

## 9.1.2 Writing to host station CPU shared memory (TO(P), DTO(P))



Q00CPU/Q01CPU whose first 5 digits of the serial No. is "04122" or higher



n1 : Head I/O number of the host CPU (BIN 16 bits)

- Basic model QCPU : 3E0H
- Universal model QCPU: 3E0H to 3E3H

n2 : CPU shared memory address of the write destination host CPU (BIN 16 bits)

- Basic model QCPU : 192 to 511
- Universal model QCPU: 2048 to 4095, 10000 to 24335\*2

(S) : Data to be written or head number of the devices where the data to be written is stored (BIN 16 bits)

n3 : Number of data blocks to be written (BIN 16 bits)

- Basic model QCPU : TO(P): 1 to 320, DTP(P) : 1 to 160
- Universal model QCPU: TO(P): 1 to 14336\*2, DTP(P) : 1 to 7168\*2

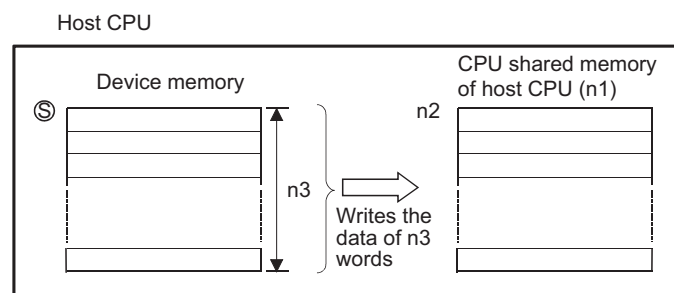
Setting Data	Internal Devices		R, ZR	J:G:G		U:G:G	Zn	Constants K, H	Other U
	Bit	Word		Bit	Word				
n1		○				○		○	○
n2		○				○		○	—
(S)		○				—		○	—
n3		○				○		○	—

\*2: The setting range varies depending on the auto refresh setting range of the multiple CPU high speed transmission function.

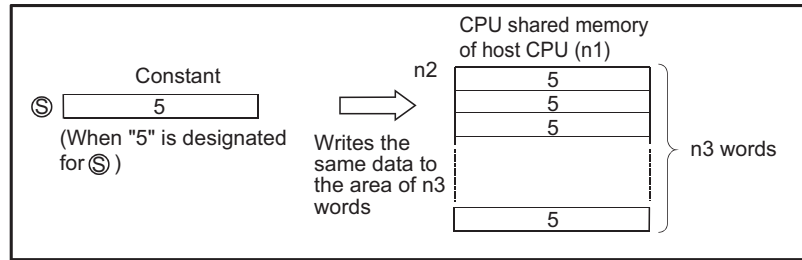
### ★ Function

#### TO

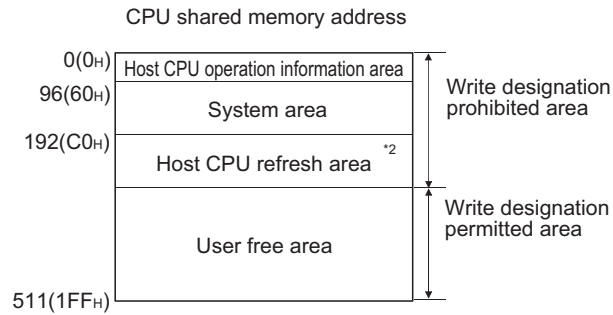
- (1) Writes device data of words (S) to n3 to the CPU shared memory address specified by n2 of the host CPU module or later address.



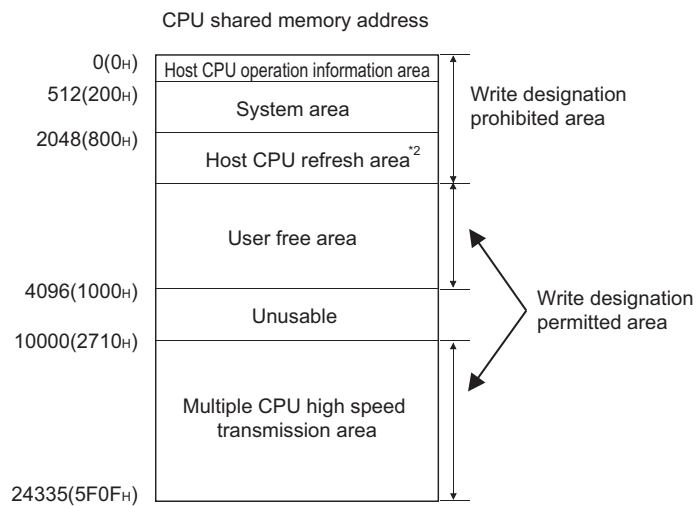
When a constant is specified to  $\textcircled{S}$ , writes the same data (value specified to  $\textcircled{S}$ ) to the area of n3 words from the specified CPU shared memory.



(a) CPU shared memory addresses of the Basic model QCPU



(b) CPU shared memory address of the Universal model QCPU\*3



\*2 : Usable as a user free area when auto refresh setting is not made.

In addition, even when auto refresh setting is made, the auto refresh send range or later is usable as a user free area.

\*3 : With Q02UCPU, data can not be written to the multiple CPU high speed transmission area.

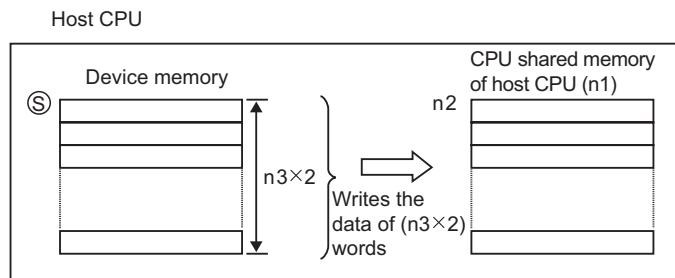
- (2) No processing is performed when the number of write points is 0.
- (3) The number of write data varies depending on the target CPU module.

CPU module	Number of Write Points
Basic model QCPU	1 to 320
Universal model QCPU	1 to 14336

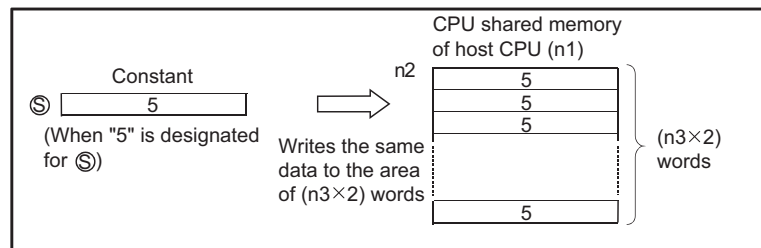


## DTO

- (1) Writes device data of words  $\textcircled{S}$  to  $(n3 \times 2)$  to the CPU shared memory address specified by  $n2$  of the host CPU module or later address.



When a constant is specified to  $\textcircled{S}$ , writes the same data (value specified to  $\textcircled{S}$ ) to the area of  $(n3 \times 2)$  words from the specified CPU shared memory.



- (2) No processing is performed when the number of write points is 0.
- (3) The number of data that can be written varies depending on the target CPU module.

CPU mode	Number of Write Points
Basic model QCPU	1 to 160
Universal model QCPU	1 to 7168

### POINT

Writing data to CPU shared memory can be performed using the intelligent function module device.

For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

## Operation Error

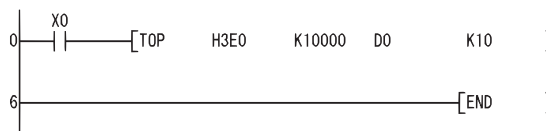
In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- (1) When the specified data is outside the following range (Error code: 4101)
  - When the number of write points (n3) is outside the specified range of the setting data.
  - When the CPU shared memory address (n2) of the write destination host CPU + the number of write points (n3) exceeds the CPU shared memory range
  - When the head number of the devices that stores the data to be written (Ⓢ) + the number of write points (n3) exceeds the device range
  - When the head of CPU shared memory address (n2) of the write destination host CPU is outside the write permitted area.
- (2) When the head of CPU shared memory address (n2) of the write destination host CPU is an invalid value. (Error code: 4111)
- (3) When the I/O number specified in (n1) is other than that of the host CPU (Exclude the case when the multiple CPU high speed transmission area of other CPU is used.) (Error code: 4112)
- (4) No CPU module is installed at the position specified by the head I/O number of the CPU module. (Error code: 2110)

## Program Example

- (1) The following program stores 10 points of data from D0 into address 10000 of the CPU shared memory of CPU No. 1 when X0 is turned ON.

[Ladder Mode]

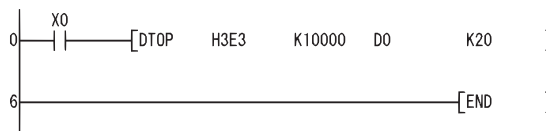


[List Mode]

Step	Instruction	Device
0	LD	X0
1	TOP	H3E0 K10000 D0 K10
6	END	

- (2) The following program stores 20 points of data from D0 into address 10000 of the CPU shared memory of CPU No. 4 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DTOP	H3E3 K10000 D0 K20
6	END	

### Remark

The n1 is specified by the first 3 digits of the hexadecimal 4 digits which represent the head I/O number of the slot mounted to the CPU module.

	CPU Slot	Slot 0	Slot 1	Slot 2
Head I/O number	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

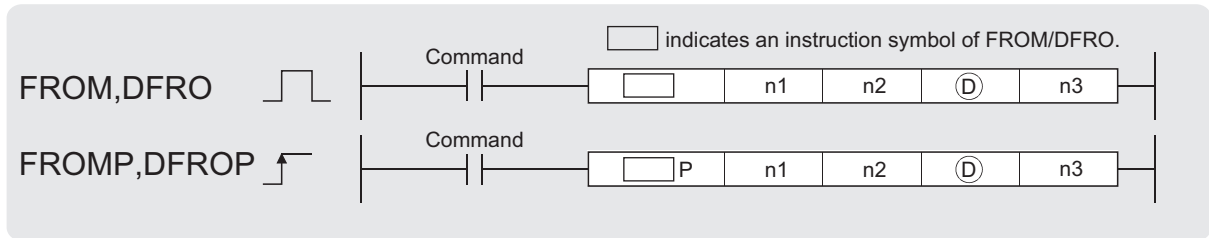


## 9.2.1 Reading from Other CPU Shared Memory (FROM(P), DFRO(P))



Basic model QCPU: The first 5 digits of serial No is "04122" or higher.  
 High performance model QCPU: Function version B or later.

(1) When Basic model QCPU, Universal model QCPU is used



- n1 : Head I/O number of the reading target CPU module (BIN 16 bits)
  - Basic model QCPU : 3E0H to 3E2H
  - Universal model QCPU: 3E0H to 3E3H
- n2 : Head address of data to be read (BIN 16 bits)
  - Basic model QCPU : 0 to 512
  - Universal model QCPU: 0 to 4095, 10000 to 24335\*<sup>3</sup>
- (D) : Head number of the devices where the read data is stored (BIN 16 bits)
- n3 : Number of read data (BIN 16 bits)
  - Basic model QCPU : FROM(P): 1 to 512, DFRO(P): 1 to 256
  - Universal model QCPU: FROM(P): 1 to 14336\*<sup>3</sup>, DRRO(P): 1 to 7168\*<sup>3</sup>

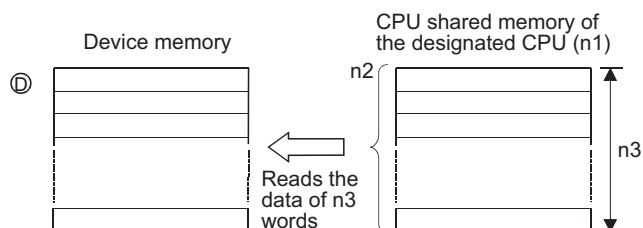
Setting Data	Internal Devices		R, ZR	JMO		UJGO	Zn	Constants K, H	Other U
	Bit	Word		Bit	Word				
n1	—	○			○			○	○
n2	—	○			○			○	—
(D)	—	○			—			—	—
n3	—	○			○			○	—

\*3: The setting range varies depending on the auto refresh setting range of the multiple CPU high speed communication function.

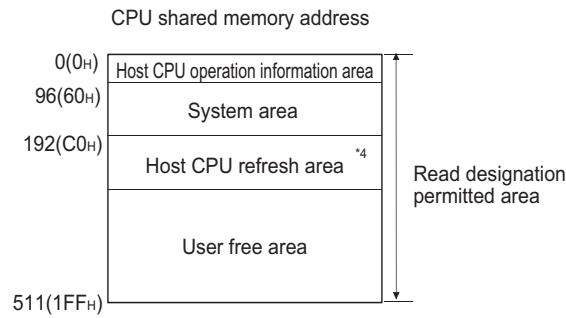
### ★ Function

#### FROM

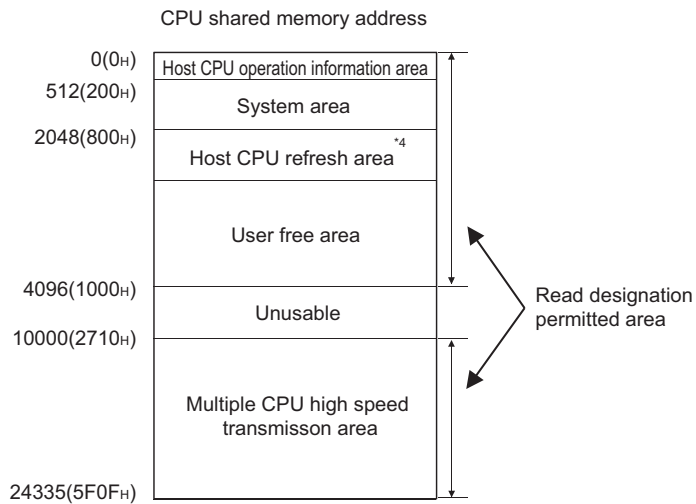
(1) Reads the data of n3 words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by (D).



## (a) CPU shared memory address of the Basic model QCPU



## (b) CPU shared memory address of the Universal model QCPU\*5



\*4 : Usable as a user free area when auto refresh setting is not made.

When auto refresh setting is made, the auto refresh send range and later are usable as a user free area.

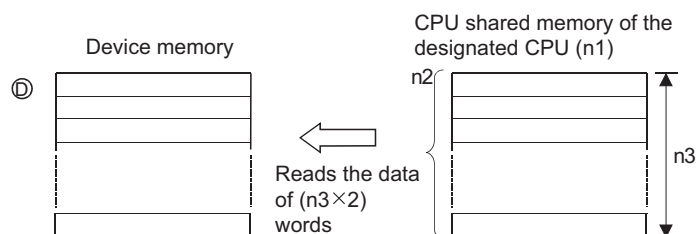
\*5 : With Q02UCPU, data can not be written to the multiple CPU high speed transmission area.

- (2) When 0 is specified in n3 as the number of data to be read, no processing is performed.
- (3) The number of data to be read changes depending on the target CPU module.

CPU Module	Number of Read Points
Basic model QCPU	1 to 512
Universal model QCPU	1 to 14336

**DFRO**

- (1) Reads the data of  $(n3 \times 2)$  words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by ①.



- (2) When 0 is specified in n3 as the number of data to be read, no processing is performed.
- (3) The number of data to be read changes depending on the target CPU module.

CPU Module	Number of Read Points
Basic model QCPU	1 to 256
Universal model QCPU	1 to 7168

## POINT

Read of data from the CPU shared memory can also be performed using the intelligent function module devices.

For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).



## Operation Error

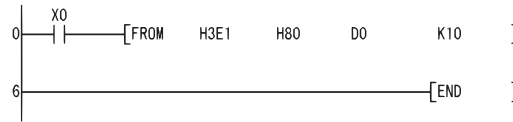
In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- (1) When the specified data is outside the following range. (Error code: 4101)
- The head of the CPU shared memory address (n2) which performs reading is outside the CPU shared memory range.
  - The address of the CPU shared memory (n2) which performs reading plus the number of read points (n3) is outside the CPU shared memory range.
  - The read data storage device number (D) plus the number of read points (n3) is outside the specified device range.
- (2) The CPU module does not exist in the position specified by the CPU module head I/O number. (Error code: 2110)
- (3) When the head of the CPU shared memory address (n2) which performs reading is an invalid value. (4097 to 9999) (Error code: 4101)

## Program Example

- (1) The following program stores 10 points of data from address C0H of the CPU shared memory of CPU No. 2 into the area starting from D0 when X0 is turned ON.

[Ladder Mode]

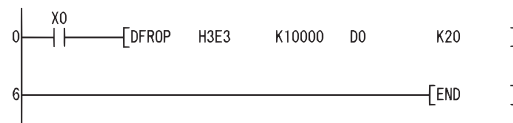


[List Mode]

Step	Instruction	Device
0	LD	X0
1	FROM	H3E1 H80 D0 K10
6	END	

- (2) The following program stores 20 points of data from address 10000 of the CPU shared memory of CPU No. 4 into the area starting from D0 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	DFROP	H3E3 K10000 D0 K20
6	END	

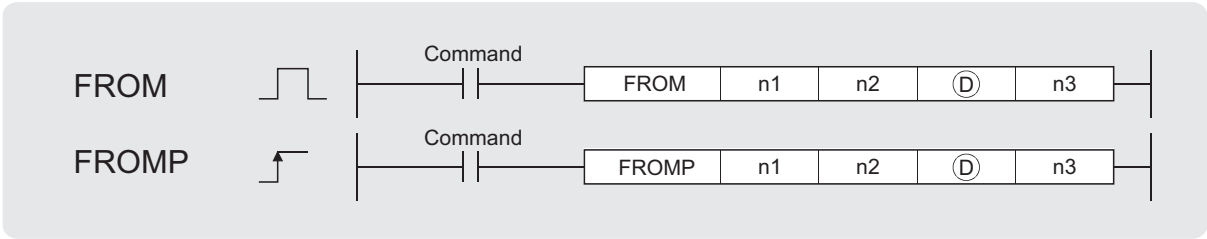
### Remark

The n1 is specified by the first 3 digits of the hexadecimal 4digits which represent the head I/O number of the slot mounted to the CPU module.

	CPU Slot	Slot 0	Slot 1	Slot 2
Head I/O number	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

The QCPU provides automatic interlocks for the FROM and TO instructions.

(2) When High Performance model QCPU, Process CPU is used

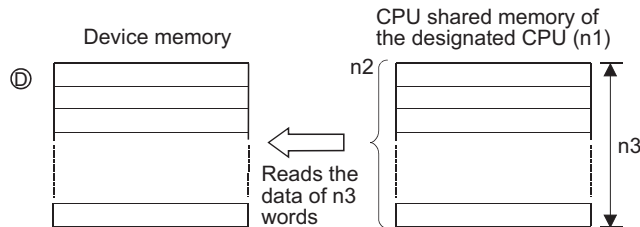


n1 : Head I/O number of the reading target CPU module (BIN 16 bits)  
 n2 : Head address of data to be read (BIN 16 bits)  
 (D) : Head number of the devices where the read data is stored (BIN 16 bits)  
 n3 : Number of read data (BIN 16 bits)

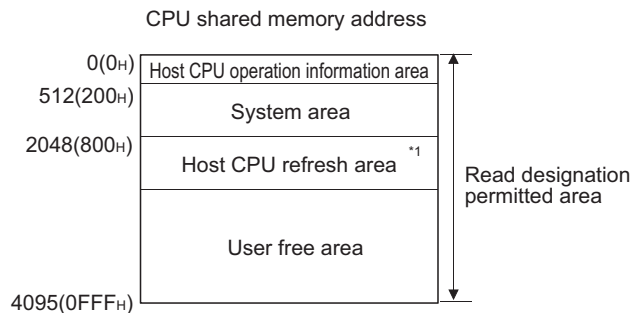
Setting Data	Internal Devices		R, ZR	J		U	Zn	Constants K, H	Other U
	Bit	Word		Bit	Word				
n1	—	○			○			○	○
n2	—	○			○			○	—
(D)	—	○			—			—	—
n3	—	○			○			○	—

★ Function

- (1) Reads the data of n3 words from the CPU shared memory address designated by n2 of the CPU module designated by n1, and stores that data into the area starting from the device designated by (D).



CPU shared memory address of the High Performance model QCPU and Process CPU



\*1 : Usable as a user free area when auto refresh setting is not made.  
 When auto refresh setting is made, the auto refresh send range and later are usable as a user free area.

- (2) When 0 is specified in n3 as the number of data to be read, no processing is performed.
- (3) The number of data to be read changes depending on the target CPU module.

CPU Module	Number of Read Points
High Performance model QCPU Process CPU	1 to 4096



**POINT**

Read of data from the CPU shared memory can also be performed using the intelligent function module devices.

For intelligent function module device, refer to the QnUCPU User's Manual (Function Explanation, Program Fundamentals) or Qn(H)/QnPH/QnPRHCPU User's Manual (Function Explanation, Program Fundamentals).

**! Operation Error**

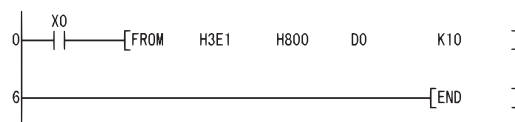
In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.

- (1) When the specified data is outside the following range. (Error code: 4101)
  - The head address of the CPU shared memory (n2) from which read will be performed is outside the CPU shared memory range.
  - The address of the CPU shared memory (n2) from which data is read plus the number of read points (n3) is outside the CPU shared memory range.
  - The read data storage device number (D) plus the number of read points (n3) is outside the specified device range.
- (2) The CPU module does not exist in the position specified by the CPU module head I/O number. (Error code: 2110)
- (3) When the head of read CPU shared memory address (n2) is an invalid value. (4097 to 9999) (Error code: 4101)

**Program Example**

- (1) The following program stores data of 10 points from address 800H of the CPU shared memory of CPU No. 2. into the area starting from D0 when X0 is turned ON.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	X0
1	FROM	H3E1 H800 D0 K10
6	END	

**Remark**

The n1 is specified by the first 3 digits of the hexadecimal 4digits which represent the head I/O number of the slot mounted to the CPU module.

	CPU Slot	Slot 0	Slot 1	Slot 2
Head I/O number	3E00	3E10	3E20	3E30
n1	3E0	3E1	3E2	3E3

The QCPU provides automatic interlocks for the FROM and TO instructions.



# 10

## QCPU INSTRUCTIONS

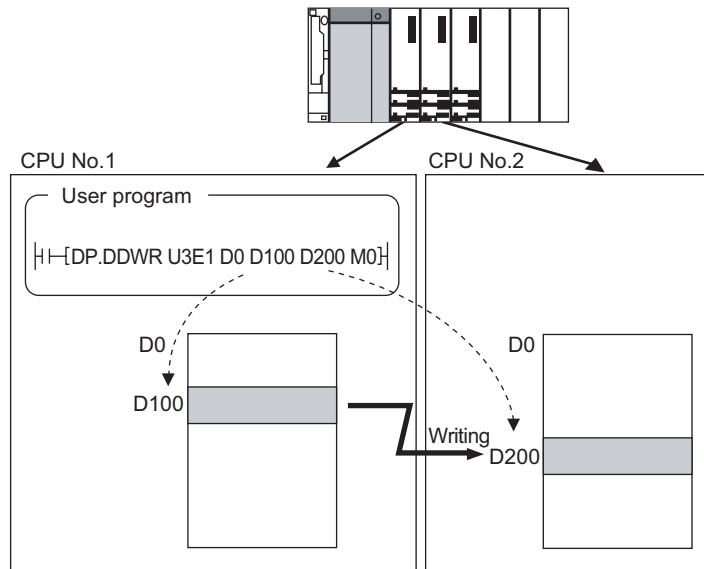
---

Category	Processing Details	Reference section
Write instruction to another CPU	Writes devices to another CPU.	Section 10.2
Read instruction from another CPU	Reads devices from another CPU.	Section 10.3

## 10.1 Overview

The multiple CPU high-speed transmission dedicated instruction directs the Universal model QCPU to write/read device data to/from the Universal model QCPU in another CPU.

The following shows an operation when CPU No.1 writes device data to CPU No.2 with the multiple CPU high-speed transmission dedicated instruction.



### POINT

The multiple CPU high-speed transmission dedicated instruction in either host CPU or another CPU (target CPU module of instruction) is available only for the following CPU modules.

- Q03UDCPU, Q04UDHCPU, Q06UDHCPU  
The first five digits of serial number is 10012 or higher.
- Q10UDHCPU, Q13UDHCPU, Q20UDHCPU, Q26UDHCPU
- QnUDE (H) CPU

- (1) Parameter setting and system configuration to execute the multiple CPU high-speed transmission dedicated instruction

The multiple CPU high-speed transmission dedicated instruction can be executed in the following parameter setting and system configuration.

- CPU No.1 uses QnUD(H)CPU or QnUDE(H)CPU.
- The multiple CPU high speed main base unit (Q3□DB) is used.
- "Use multiple CPU high speed transmission" is selected in the Multiple CPU settings screen of PLC parameter.

## (2) Writable/readable devices

## (a) Writable/readable device names

The following table shows the devices that can be written to/read from the Univesal model QCPU in another CPU with the multiple CPU high-speed transmission dedicated instruction.

Category	Type	Device name	Setting of target device	Remarks
Internal user device	Bit device	X, Y, M, L, B, F, SB	△	Requirements for the setting • Digits are specified by 16 bits (4 digits). • The start bit device is multiples of 16(10H).
	Word device	T, ST, C, D, W, SW	○	—
Internal system device	Bit device	SM	△	Requirements for the setting • Digits are specified by 16 bits (4 digits). • The start bit device is multiples of 16(10H).
	Word device	SD	○	—
File register	Word device	R, ZR	○	—

○ :Settable △ :Settable with conditions

### POINT

SB, SW, SM, and SD include system information area. Take care not to destroy the system information when writing data to the devices above with the D(P).DDWR instruction of the multiple CPU high-speed transmission dedicated instruction.

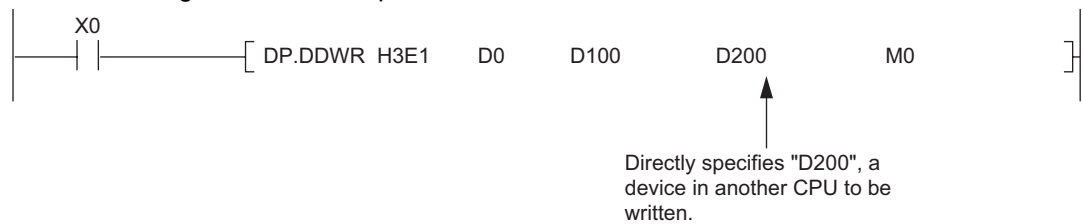
## (3) Specification method of a device and writable/readable device range

There are two methods for specifying a device in another CPU: device specification and string specification. They differ in writable/readable device range to another CPU.

## (a) Device specification

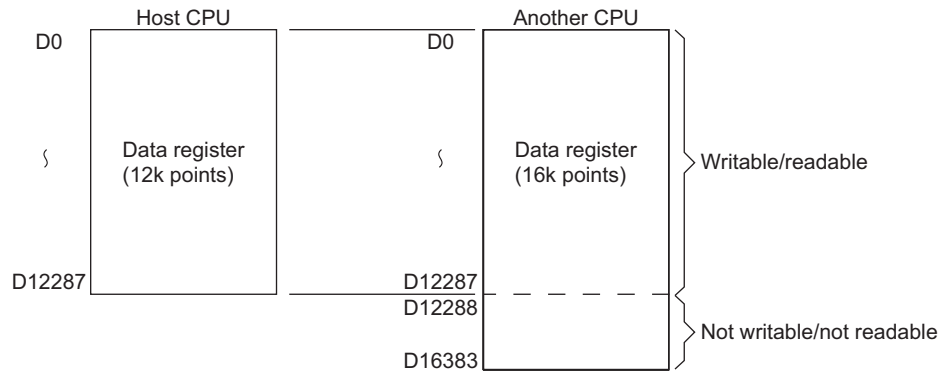
The device specification is a method to directly specify a device in another CPU to be written/read.

Program for device specification with the DP.DDWR instruction



In the device specification, data can be written/read within the device range of host CPU. For example, when data register in host CPU is 12k points and data register in another CPU is 16k points, data can be written/read by 12k points from the start of the data register in another CPU.

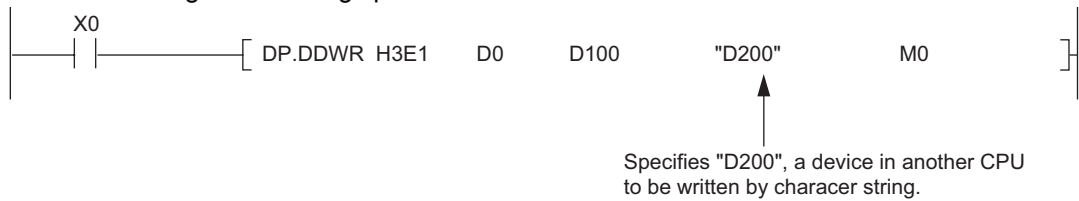
Writable/readable device range in device specification



(b) String specification

The string specification is a method to specify a device in another CPU to be written/read by character string.

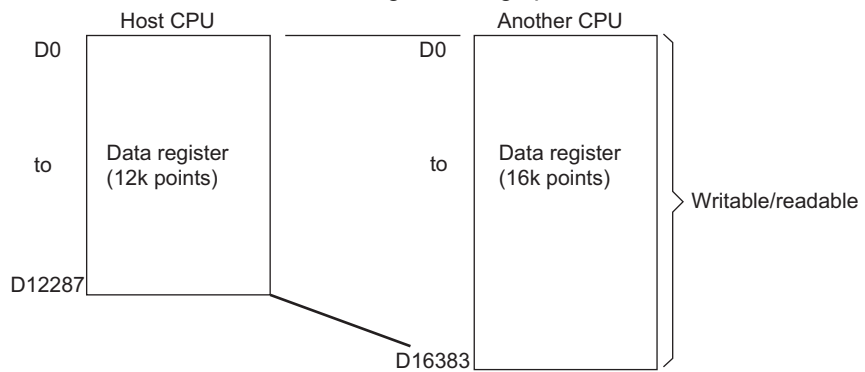
Program for string specification with the DP.DDWR instruction



In the string specification, data can be written to/read from all device ranges of another CPU.

For example, when data register in host CPU is 12k points and data register in another CPU is 16k points, data can be written/read by 16k points from the start of the data register in another CPU.

Writable/readable device range in string specification



**Remark**

The following explains precautions for string specification.

- The number of characters that can be specified is 32.
- Whether "0" is appended at the start of the device number or not, the devices are processed as the same.
- For example, both "D1" and "D0001" are processed as "D1".
- Whether a device is specified by upper case character or lower-case character, they are processed as the same.
- For example, both "D1" and "d1" are processed as "D1".
- If a device not existing in another CPU is specified by a character string, the instruction will be completed abnormally.

(4) Managing the multiple CPU high speed transmission area

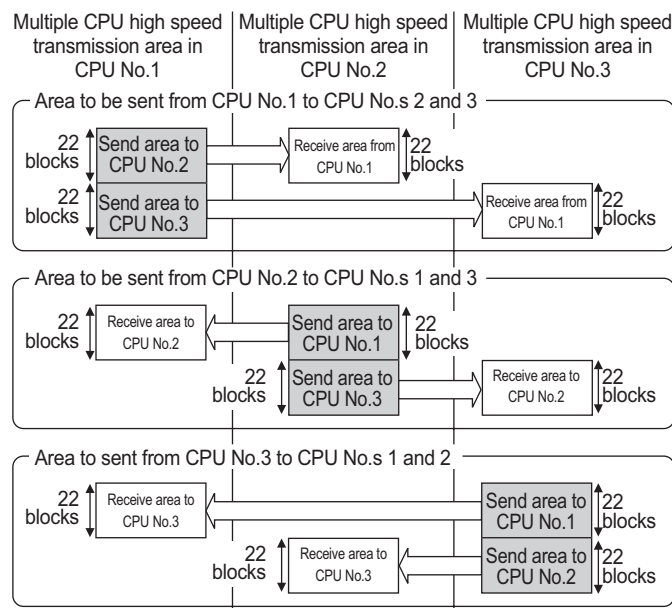
- (a) The multiple CPU high speed transmission area is managed by blocks in units of 16 words.

The following table shows the number of blocks that can be used in each CPU and the number of blocks used in the instruction.

C Number of CPU modules	System area *1	
	1k points	2k points
2	46	110
3	22	54
4	14	35

\*1: For setting of the system area, refer to the QCPU User's Manual (Multiple CPU System).

- (b) The following shows configuration of the multiple CPU high speed transmission area when the multiple CPU system is configured with three CPU modules and the system area size is 1k word.



- (5) The number of blocks used for the instruction

The number of blocks used for the instruction depends on the number of write points.

The following table shows the number of blocks used for the instruction.

Number of write/read points specified by the instruction	D(P).DDWR instruction	D(P).DDR D instruction
1 to 4	1	1
5 to 20	2	
21 to 36	3	
37 to 52	4	
53 to 68	5	
69 to 84	6	
85 to 100	7	

- (6) The multiple CPU high-speed transmission dedicated instructions that can be executed concurrently

For the Universal model QCPU, the multiple CPU high-speed transmission dedicated instructions can be concurrently executed within the range satisfying the following formula.

$$\left[ \begin{array}{c} \text{The number of blocks that} \\ \text{can be used in each CPU} \end{array} \right] \geq \left[ \begin{array}{c} \text{Total number of blocks used for the} \\ \text{instructions concurrently executed} \end{array} \right]$$

When the number of blocks used for the multiple CPU high-speed transmission dedicated instructions exceeds the total number of blocks in the multiple CPU high speed transmission area, the instruction will not be executed in the scan (no processing) but executed at the next scan.

Note that the instruction will be completed abnormally when the number of empty blocks in the multiple CPU high speed transmission area is less than the setting values of SD796 to SD799 (maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting) at the execution of the instruction.

The following table shows execution possibility of the multiple CPU high-speed transmission dedicated instructions when the number of empty blocks in the multiple CPU high speed transmission area is less than the number of blocks used for the multiple CPU high-speed transmission dedicated instructions or the setting values of SD796 to SD799.

Magnitude relation between the number of blocks used for the instructions*1 and the number of empty blocks	Number of blocks used for the instruction*1 $\leq$ Number of empty blocks*2	Number of blocks used for the instruction*1 $>$ Number of empty blocks*2
	Executed	Not executed (no processing)
Magnitude relation between SD setting value and the number of empty blocks	Completed abnormally	
SD setting value*3 $\leq$ Number of empty blocks*2		
SD setting value*3 $>$ Number of empty blocks*2		

\*1:The number of blocks used for the multiple CPU high-speed transmission dedicated instruction.

\*2:The number of empty blocks in the multiple CPU high-speed transmission area.

\*3:Setting values from SD796 of SD799.

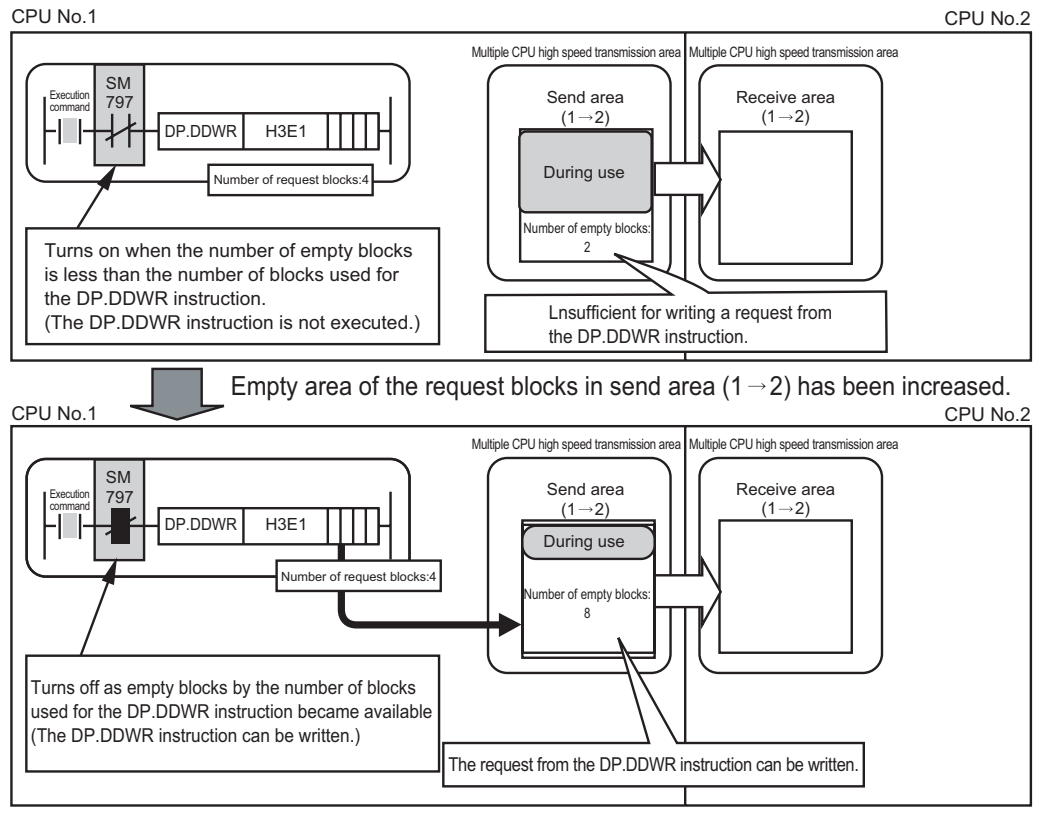


- (7) Interlock when using the multiple CPU high-speed transmission dedicated instruction
  - (a) Special relays SM796 to SM799 (maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting) can be used as an interlock for the multiple CPU high-speed transmission dedicated instruction.  
When executing the multiple CPU high-speed transmission dedicated instructions concurrently, use SM796 to SM799 as an interlock for the instructions.

**POINT**

When using special relays SM796 to SM799, set the maximum number of blocks for the instruction used for each CPU to special registers SD796 to SD799. (For example, when the maximum number of blocks for the multiple CPU high-speed transmission dedicated instruction to be executed to CPU No.3 is 5, set 5 to SD798.)

When the multiple CPU high speed transmission area becomes equal to or less than the number of blocks set at SD796 to SD799, the corresponding special relay (SM796 to SM799) turns on.



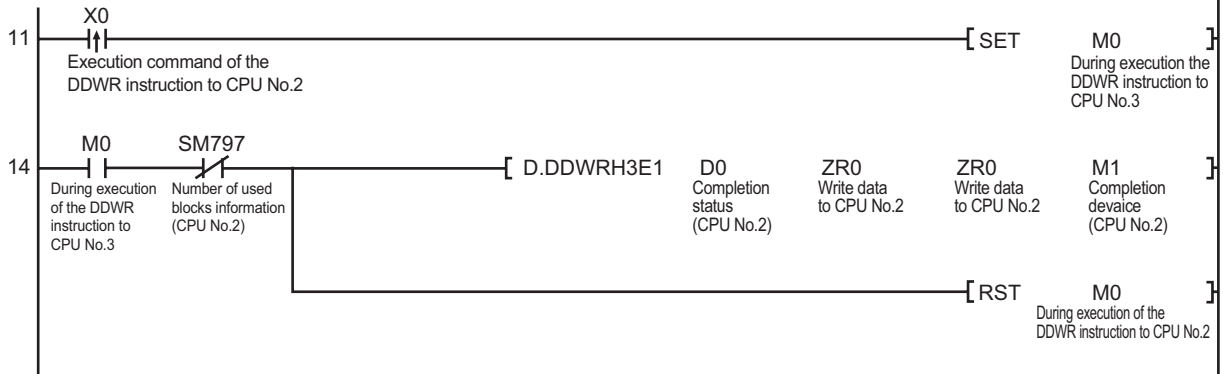
(b) Program example when SM796 to SM799 are used as an interlock

The following shows a program that executes the D.DDWR instruction to CPU No.2 at the rise of X0, and executes the D.DDWR instruction to CPU No.3 at the rise of X1.

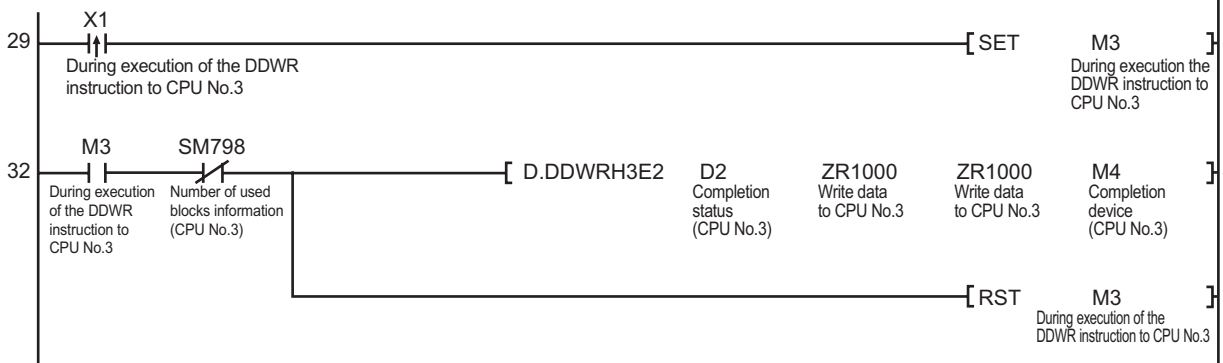
The maximum number of used blocks for multiple CPU high speed transmission dedicated



The DDWR instruction is executed to CPU No.2 at the rise of X0

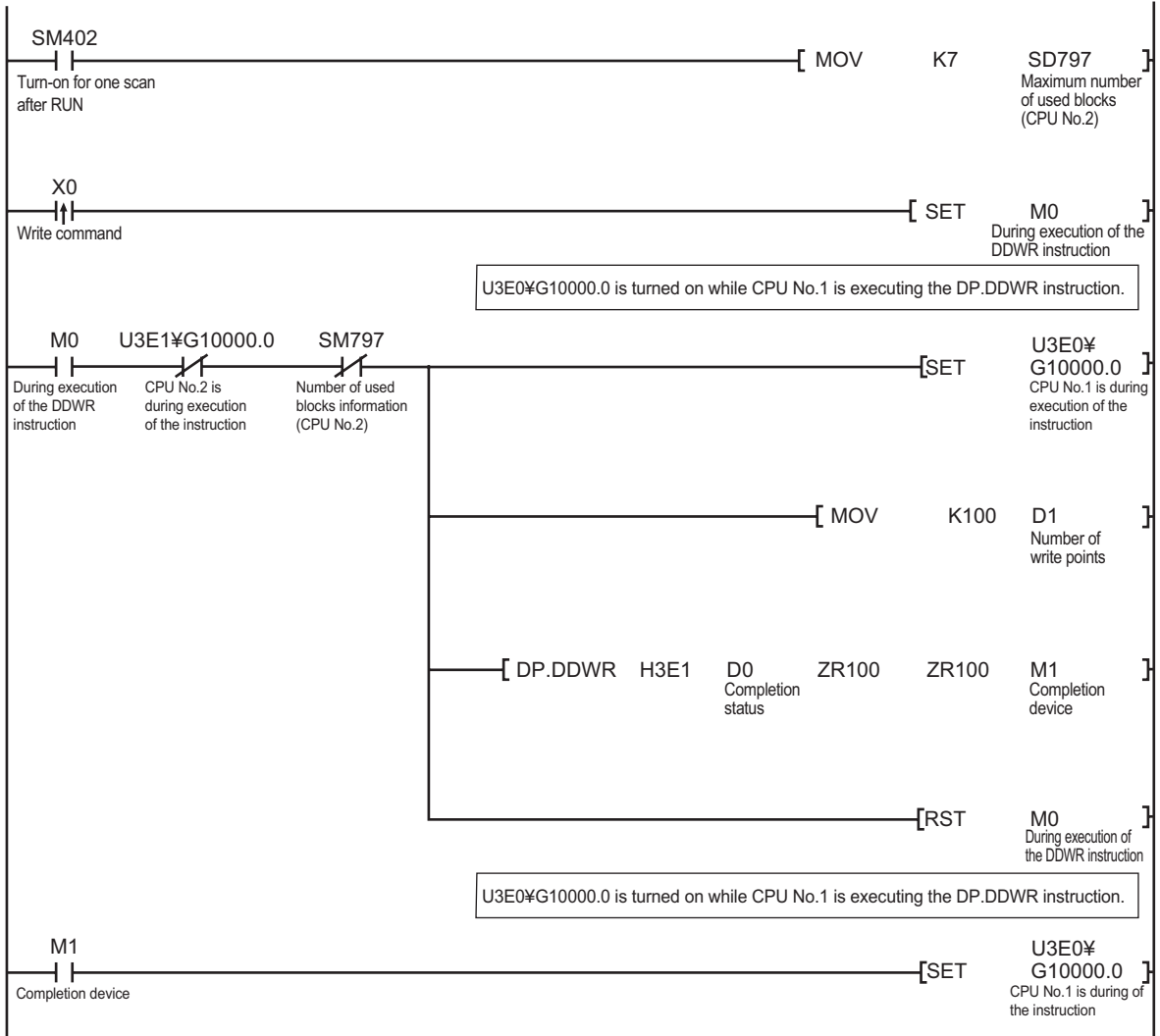


The DDWR instruction is executed to CPU No.3 at the rise of X1

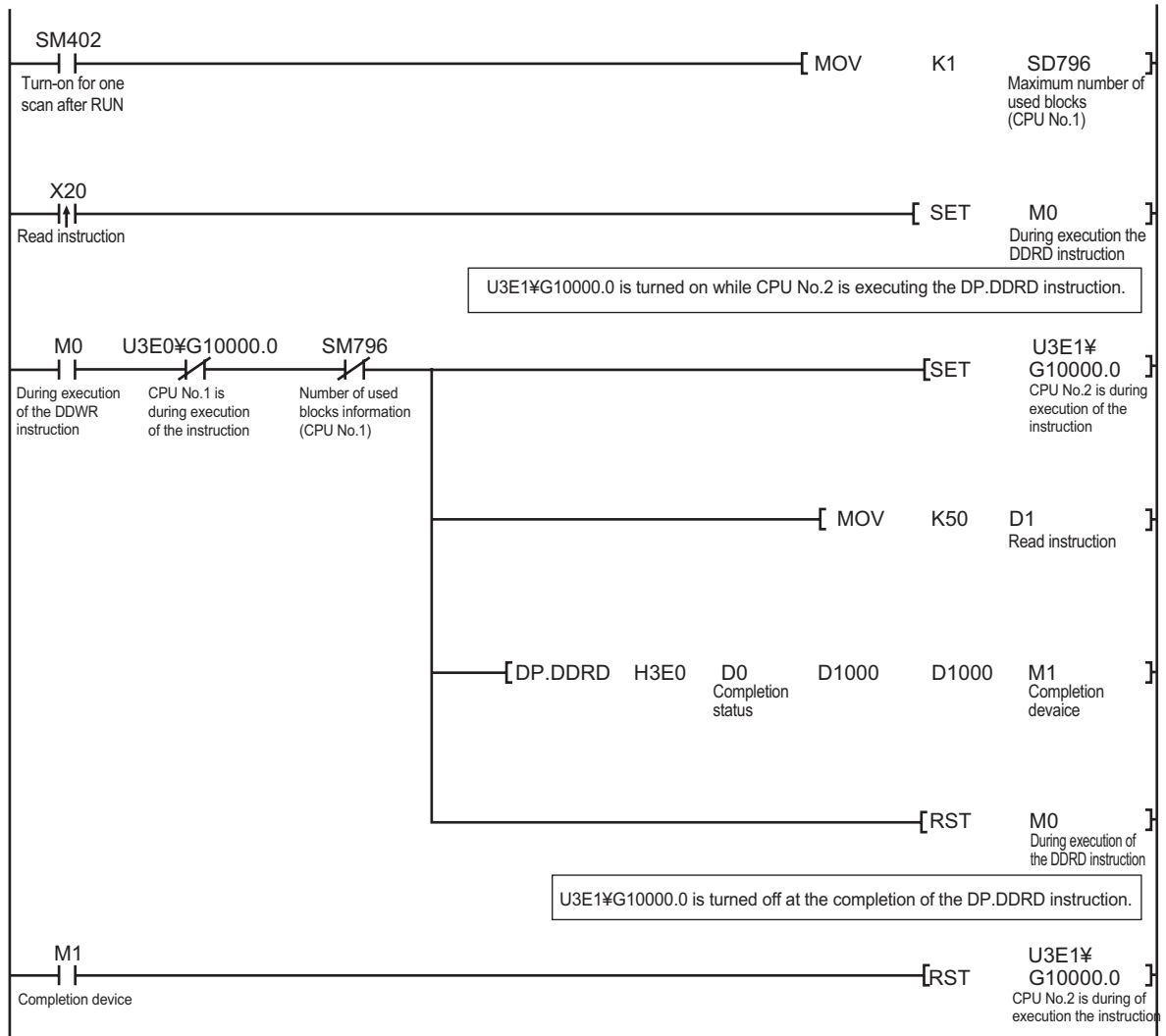


- (8) Program example when the multiple CPU high-speed transmission dedicated instructions are executed to CPU modules by turns
- When the multiple CPU high-speed transmission dedicated instructions are executed to Universal model QCPUs by turns, release an interlock to prevent the concurrent execution. Use the cyclic transmission area device (from U3En¥G10000) as an interlock.
- The following shows a program example when the multiple CPU high-speed transmission dedicated instructions are executed at CPU No.s 1 and 2 by turns.

Program example when the multiple CPU high-speed transmission dedicated instruction is executed at CPU No.1



Program example when the multiple CPU high-speed transmission dedicated instruction is executed at CPU No.2



(9) Program example when data exceeding 100 words are written/read with the multiple CPU high-speed transmission dedicated instruction

The maximum number of write/read points that can be processed with the multiple CPU high-speed transmission dedicated instruction is 100 words. Data exceeding 100 words can be written/read by executing the multiple CPU high-speed transmission dedicated instruction at several times.

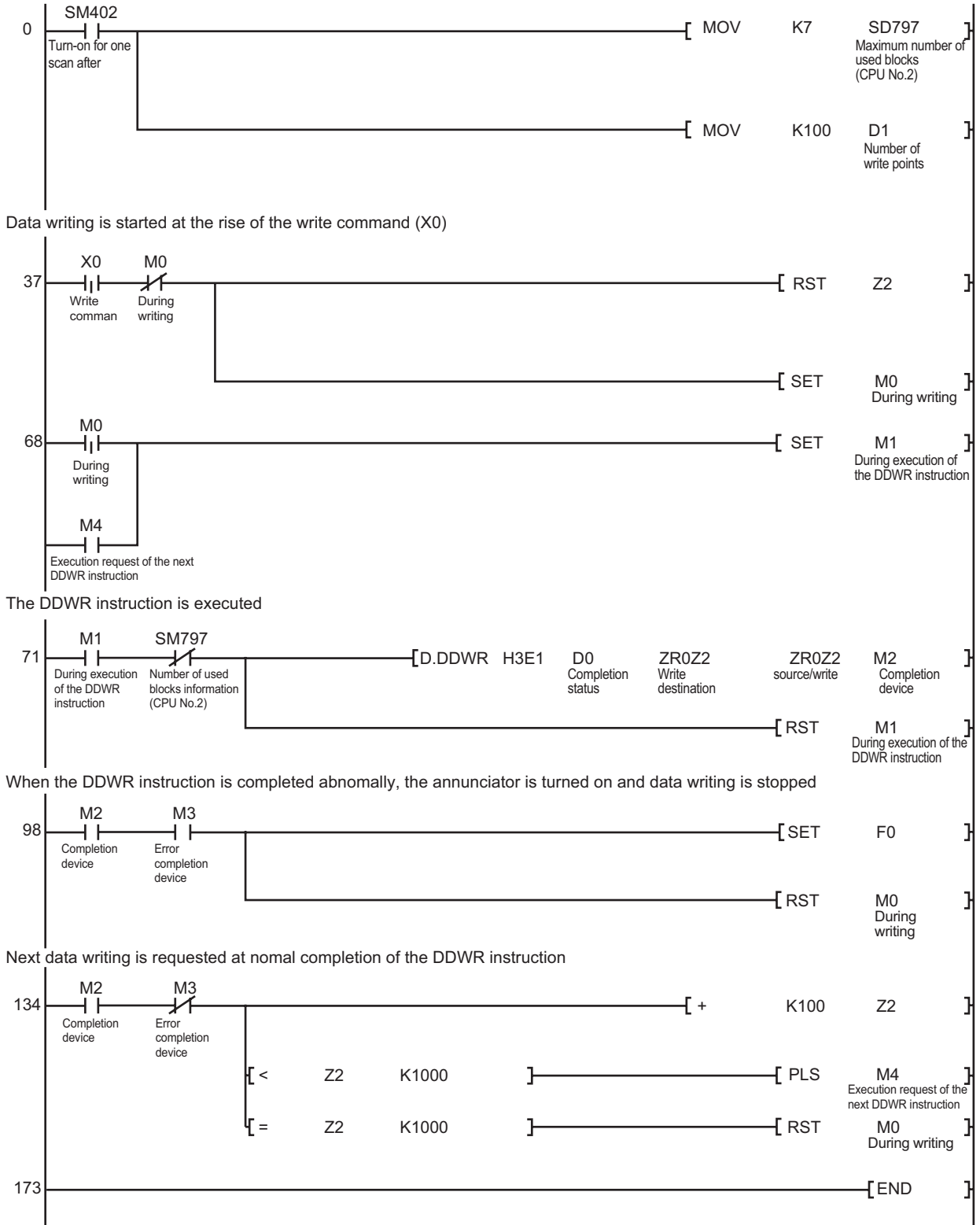
The following shows a program example using the D(P).DDWR instruction of the multiple CPU high-speed transmission dedicated instruction. The similar program can be used when using the D(P).DDR D instruction of the multiple CPU high-speed transmission dedicated instruction.

(a) Program example when one D(P).DDWR instruction is executed  
 The following shows a program example that writes ZR0 to ZR999 (1000 points) in CPU No.1 to ZR0 to ZR999 in CPU No.2 with the D.DDWR instruction.

In the following program example, the next D.DDWR instruction is executed after the completion device of the D.DDWR instruction (M2) turns on so that only one D.DDWR instruction may be executed.

Program example when one D(P).DDWR instruction is executed

The maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting is set to CPU No.2



(b) Program example when the D(P).DDWR instructions are executed concurrently

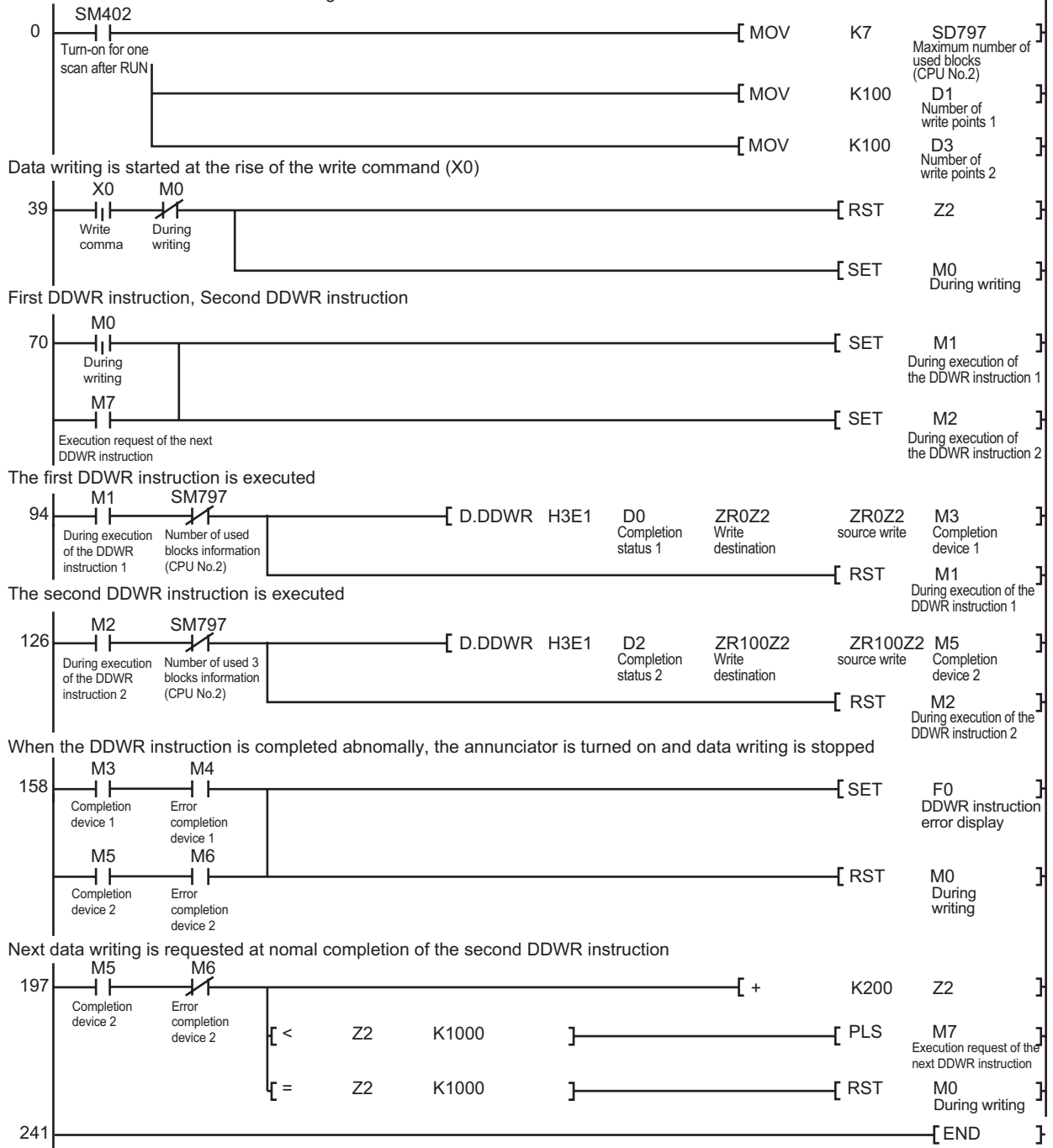
The following shows a program example that writes ZR0 to ZR999 (1000 points) in CPU No.1 to ZR0 to ZR999 in CPU No.2 with the D.DDWR instruction.

As shown on the program example, multiple CPU device write/read instructions can be executed concurrently.

When reading/writing devices with the multiple CPU high-speed transmission dedicated instructions concurrently, the more the total number of blocks in the multiple CPU high speed transmission area (send area), the more the time taken to complete reading/writing with the multiple CPU high-speed transmission dedicated instruction can be shortened.

Program example when the D(P).DDWR instructions are executed concurrently

The maximum number of used blocks for multiple CPU high-speed transmission dedicated instruction setting is set to CPU No.2

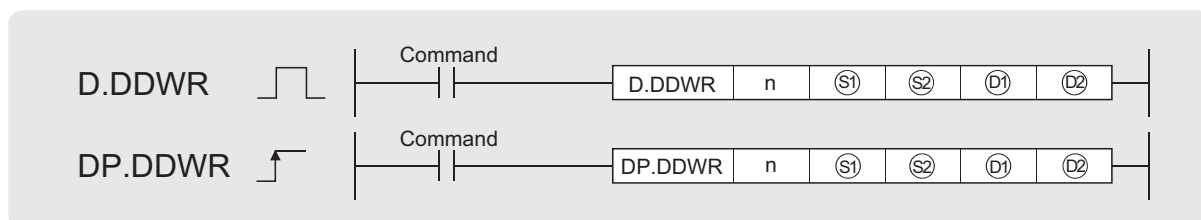


## 10.2 Writing Devices to Another CPU (D(P).DDWR)



10

Q03UDCPU, Q04UDHCPU, Q06UDHCPU: that the first 5 digits of serial number is 10012 or higher QnUDE(H)CPU.



Setting data	Internal device		R, ZR	JIO		UIGO	Zn	Constant K, H	Others
	Bit	Word <sup>*6</sup>		Bit	Word				
n <sup>*2</sup>	—	○	○			—		○	—
(S1) <sup>*3</sup>	—	△ <sup>*4</sup>	△ <sup>*5</sup>			—		—	—
(S2) <sup>*3</sup>	—	○	○			—		—	—
(D1) <sup>*3</sup>	—	○	○			—		—	—
(D2) <sup>*3</sup>	△ <sup>*7</sup>	—	△ <sup>*5</sup>			—		—	—

\*2: Index modification cannot be made to setting data n.

\*3: Index modification cannot be made to setting data from (S1) to (D2).

\*4: Local devices cannot be used.

\*5: File registers cannot be used per program.

\*6: FD @□ (indirect specification) cannot be used.

\*7: FX and FY cannot be used.

### Set Data

Setting data	Description	Data type
n	The result of dividing the start I/O number of another CPU by 16 CPU No.1: 3E0H, CPU No.2: 3E1H, CPU No.3: 3E2H, CPU No.4: 3E3H	BIN 16 bits
(S1)	Start device of the host CPU that stores control data	Device name
(S2)	Start device of the host CPU that stores data to be written	
(D1)	Start device of another CPU that stores write data	Device <sup>*6</sup> Character string <sup>*7</sup>
(D2)	Completion device	Bit

\*6: By specifying a file register (R, ZR), data can be written to devices in another CPU, outside the range of host CPU.

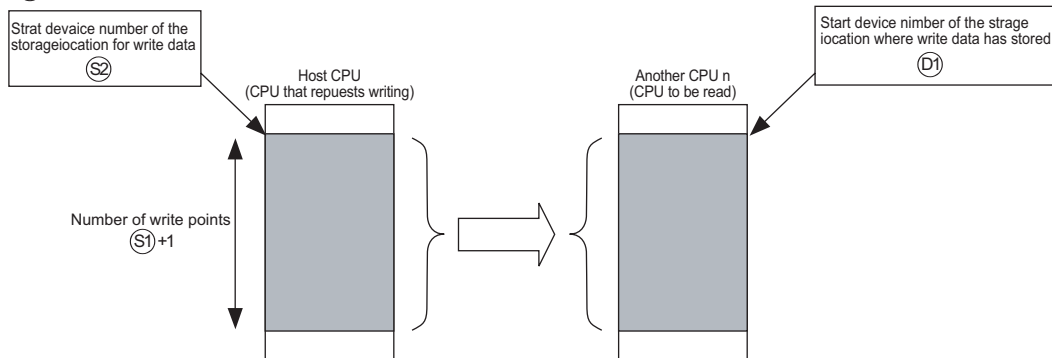
\*7: By specifying the start device by " ", devices can be written to devices in another CPU, outside the range of host CPU.

## Control Data

Device	Item	Setting data	Setting range	Set by
(S1)+0	Completion status	An execution result upon completion of the instruction is stored. 0000(H): No errors (normal completion) Other than 0000(H): Error code (error completion)	—	System
(S1)+1	Number of write points	Set the number of write points in units of words.	1 to 100	User

## Function

- (1) In multiple CPU system, data stored in a device specified by host CPU (S2) or later is stored by the number of write points specified by (S1)+1 into a device specified by another CPU (n) (D1) or later.



- (2) Whether to complete the D(P).DDWR instruction normally can be checked by the completion device (D2)+0 and completion status display device (D2)+1.
- Completion device (D2)+0  
Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing.
  - Completion status display device (D2)+1  
This device turns on/off depending on the status upon completion of the instruction.
    - Normal completion: Off
    - Error completion: Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing (At error completion, an error code is stored at control data (S1)+0: Completion status)).



- (3) The number of blocks used for the instruction depends on the number of write points (refer to Section 12.1).

Number of blocks used for the instruction	
Number of write points specified by the instruction	D(P).DDWR instruction
1 to 4	1
5 to 20	2
21 to 36	3
37 to 52	4
53 to 68	5
69 to 84	6
85 to 100	7

- (4) The instruction will be completed abnormally when there are no empty blocks in the multiple CPU high speed transmission area.

Set the number of blocks used for the instruction at special registers (SD796 to SD799), and use the special relays (SM796 to SM799) as an interlock prevent error completion (refer to Section 12.1).

## Operation Error

In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.

- (1) Specified another CPU is wrong or the multiple CPU high-speed transmission dedicated instruction cannot be used in the setting (Error code: 4350)
  - The reserved CPU has been specified.
  - Unmounted CPU has been specified.
  - Another CPU start I/O number divided by 16n is out of 3E0H to 3E3H.
  - The instruction was executed without setting "Use multiple CPU high speed transmission".
  - The instruction was executed with the Q02UCPU.
  - Host CPU has been specified.
  - The CPU where the instruction cannot be executed has been specified.
- (2) The instruction cannot be executed with the CPU. (Error code: 4351)
  - Another CPU does not support this instruction.
- (3) The number of devices is wrong. (Error code: 4352)
- (4) The device that cannot be used for the instruction has been specified. (Error code: 4353)
- (5) A device has been specified by the character string that cannot be used. (Error code: 4354)
- (6) The number of write points ( $\text{Ⓢ}+1$ ) is other than 0 to 100. (Error code 4354)

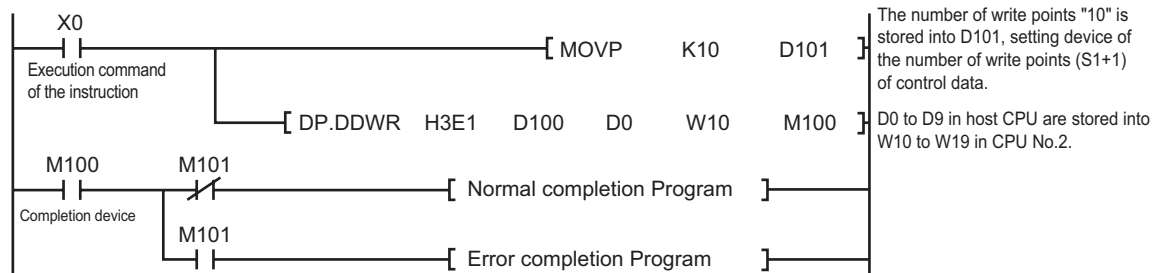
In any of the following cases, the instruction is completed abnormally, and an error code is stored into a device specified at completion status storage device (S1+0).

- (1) The request of the instruction to the target CPU is more than the acceptable value (no empty blocks exist in the multiple CPU high speed transmission area).  
(Error code: 0010H)
- (2) A device for another CPU specified at S1 cannot be used at another CPU, or is out of device range.  
(Error code: 1001H)
- (3) The number of write points set with the D(P).DDWR instruction is 0.  
(Error code: 1080H)
- (4) The response of the instruction from another CPU cannot be returned (no empty blocks exist in the multiple CPU high speed transmission area).  
(Error code: 1003H)

## Program Example

- (1) This program stores data by 10 words starting from D0 in host CPU into W10 or later in CPU No.2 when XO turns on.

[Ladder mode]



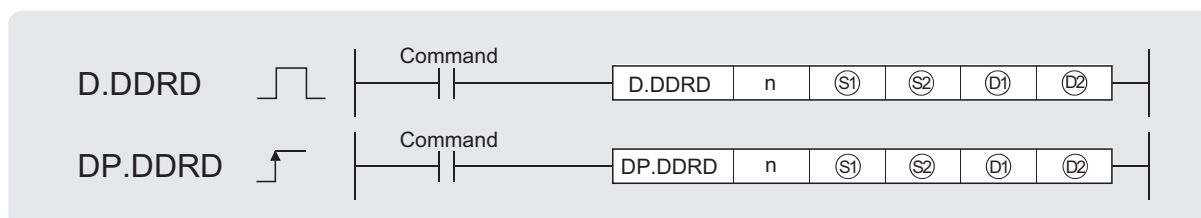
## Caution

- (1) Digit specification of bit device is possible for n, S2, and D1. Note that when the digit specification of bit device is made to S2 or D1, the following conditions must be met.
  - Digits are specified by 16 bits (4 digits).
  - The start bit device is multiples of 16 (10H).
- (2) Execute this instruction after checking that the write target CPU is powered on. Not doing so may end up no processing.
- (3) If changing a range of the device specified at setting data between after execution of the instruction and turn-on of the completion device, data to be stored by system (completion status, completion device) cannot be stored normally.
- (4) SB, SW, SM, and SD include system information area. Take care not to destroy the system information when writing data to the devices above with the D(P).DDWR instruction of the multiple CPU high-speed transmission dedicated instruction.

## 10.3 Reading Devices from Another CPU (D(P).DDR)



Q03UDCPU, Q04UDHCPU, Q06UDHCPU: that the first 5 digits of serial number is 10012 or higher  
QnUDE(H)CPU.



Setting data	Internal device		R, ZR	Jn		Uj	Zn	Constant K, H	Others
	Bit	Word		Bit	Word				
n <sup>*2</sup>	—	○	○			—		○	—
S1 <sup>*3</sup>	—	△ <sup>*3</sup>	△ <sup>*4</sup>			—		—	—
S2 <sup>*3</sup>	—	○	○			—		—	—
D1 <sup>*3</sup>	—	△ <sup>*3</sup>	△ <sup>*4</sup>			—		—	—
D2 <sup>*3</sup>	△ <sup>*3</sup>	—	△ <sup>*4</sup>			—		—	—

\*2: Index modification cannot be made to setting data n.

\*3: Index modification cannot be made to setting data from S1 to D2.

\*4: Local devices cannot be used.

\*5: File registers cannot be used per program.

\*6: FD @□ (indirect specification) cannot be used.

\*7: FX and FY cannot be used.

### Set Data

Setting data	Description	Data type
n	The result of dividing the start I/O number of another CPU by 16 CPU No.1: 3E0H, CPU No.2: 3E1H, CPU No.3: 3E2H, CPU No.4: 3E3H	BIN 16 bits
S1	Start device of the host CPU that stores control data	Device name
S2	Start device of another CPU that stores data to be read	
D1	Start device of the host CPU that stores read data	Device <sup>*6</sup> Character string <sup>*7</sup>
D2	Completion device	Bit

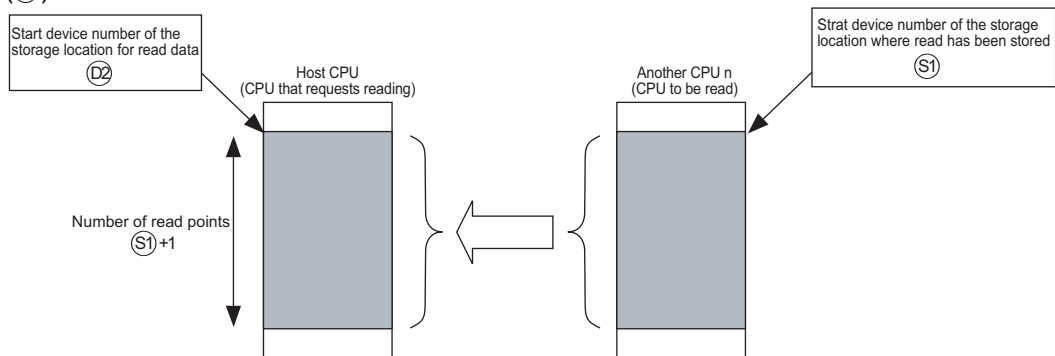
\*6: By specifying a file register (R, ZR), data can be read to devices in another CPU, outside the range of host CPU.

\*7: By specifying the start device by " ", devices can be read to devices in another CPU, outside the range of host CPU.

## Control Data

Device	Item	Setting data	Setting range	Set by
(S1)+0	Completion status	An execution result upon completion of the instruction is stored. 0000(H): No errors (normal completion) Other than 0000(H): Error code (error completion)	—	System
(S1)+1	Number of read points	Set the number of read points in units of words.	1 to 100	User

- (1) In multiple CPU system, data stored in a device specified by another CPU (n) (D1) or later is stored by the number of read points specified by (S1)+1 into a device specified by host CPU (S2) or later.



- (2) Whether to complete the D(P).DDRD instruction normally can be checked by the completion device (D2)+0 and completion status display device (D2)+1.
- END processing in scan data that CPU completed the instruction turns on the device and the next END processing turns off the device.
  - This device turns on/off depending on the status upon completion of the instruction.
    - Normal completion: Off
    - Error completion: Turns on at END processing in the scan where the instruction has been completed, and turns off at the next END processing (At error completion, an error code is stored at control data (S1)+0: Completion status)).
- (3) The number of blocks used for the instruction depends on the number of read points (refer to Section 12.1).

Number of blocks used for the instruction	
Number of read points specified by the instruction	D(P).DDRD instruction
1 to 100	1

- (4) The instruction will be completed abnormally when there are no empty blocks in the multiple CPU high speed transmission area.  
Set the number of blocks used for the instruction at special registers (SD796 to SD799), and use the special relays (SM796 to SM799) as an interlock prevent error completion (refer to Section 12.1).

## Operation Error

In any of the following cases, an operation error occurs, the error flag (SM0) turns on, and an error code is stored into SD0.

- (1) Specified another CPU is wrong or the multiple CPU high-speed transmission dedicated instruction cannot be used in the setting (Error code: 4350).
  - The reserved CPU has been specified.
  - Unmounted CPU has been specified.
  - The result of dividing the start I/O number of another CPU by 16n is outside the range of 3E0H to 3E3H.
  - The instruction was executed without setting "Use multiple CPU high speed transmission".
  - The instruction was executed with the Q02UCPU.
  - Host CPU has been specified.
  - The CPU where the instruction cannot be executed has been specified.
- (2) The instruction cannot be executed with the CPU. (Error code: 4351)
  - Another CPU does not support this instruction.
- (3) The number of devices is wrong. (Error code: 4352)
- (4) The device that cannot be used for the instruction has been specified. (Error code: 4353)
- (5) A device has been specified by the character string that cannot be used. (Error code: 4354)
- (6) The number of read points ( $\text{S1}+1$ ) is other than 0 to 100. (Error code: 4355)

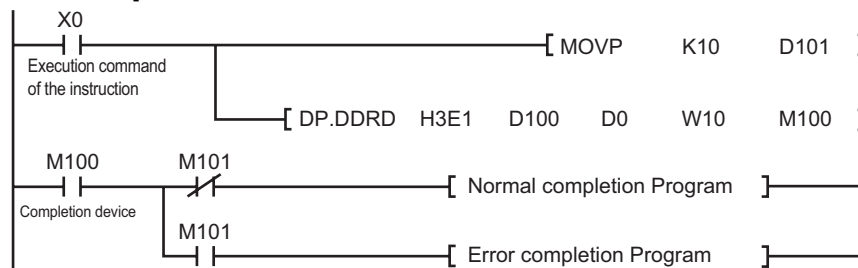
In any of the following cases, the instruction is completed abnormally, and an error code is stored into a device specified at completion status storage device ( $\text{S1}+0$ ).

- (1) The request of the instruction to the target CPU is more than the acceptable value (no empty blocks exist in the multiple CPU high speed transmission area). (Error code: 0010H)
- (2) A device for another CPU specified at  $\text{S2}$  cannot be used at another CPU, or is out of device range. (Error code: 1001H)
- (3) The number of read points set with the D(P).DDRDRD instruction is 0. (Error code: 1081H)
- (4) The response of the instruction from another CPU cannot be returned (no empty blocks exist in the multiple CPU high speed transmission area). (Error code: 1003H)

## Program Example

- (1) This program stores data by 10 words starting from D0 in CPU No.2 into W10 or later in host CPU when XO turns on.

[Ladder mode]



## Caution

- (1) Digit specification of bit device is possible for n,  $\text{\textcircled{S}}$ , and  $\text{\textcircled{D}}$ . Note that when the digit specification of bit device is made to  $\text{\textcircled{S}}$  or  $\text{\textcircled{D}}$ , the following conditions must be met.
  - Digits are specified by 16 bits (4 digits).
  - The start bit device is multiples of 16 (10H).
- (2) Execute this instruction after checking that the read target CPU is powered on. Not doing so may end up no processing.
- (3) If changing a range of the device specified at setting data between after execution of the instruction and turn-on of the completion device, data to be stored by system (completion status, completion device) cannot be stored normally.

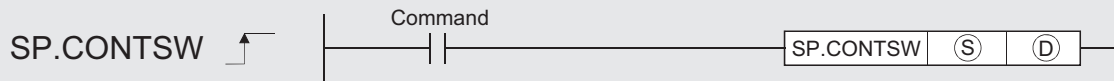
# 11

## QCPU INSTRUCTIONS

---

Category	Processing Details	Reference section
System switching instruction	Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.	Section 11.1

# 11.1 System Switching Instruction (SP.CONTSW)



Ⓢ : Value other than 0 and used to identify the processing that issued the system switching request (BIN 16 bits)

Ⓣ : Error completion device number (bits)

Setting Data	Internal Devices		R, ZR	JMO		UNGO	Zn	Constants K, H	Other
	Bit	Word		Bit	Word				
Ⓢ	—	○						○	—
Ⓣ	○	○*1						—	—

\*1: The bit specification for the word device is available.

## ★ Function

- (1) Switches between the control system and standby system at the END processing of the scan executed with the SP.CONTSW instruction.
- (2) When using the SP.CONTSW instruction for system switching, the "manual switching enable flag (SM1592)" must have been turned ON (enabled) in advance.
- (3) Ⓢ is provided to identify the processing block of the program where system switching occurred when multiple SP.CONTSW instructions are used.

At Ⓢ, specify a value within the ranges -32768 to -1 and 1 to 32767 (1H to FFFFH).

The Ⓢ value specified for the SP.CONTSW instruction is stored into the "system switching instruction argument (SD6)" of the error common information when the system switching is normally completed. \*2

When multiple SP.CONTSW instructions are executed during the same scan, the argument of the SP.CONTSW instruction executed first is stored into the system switching instruction argument (SD6).

- (4) When system switching is normally completed, the Ⓢ value specified for the SP.CONTSW instruction is stored into the "system switching instruction argument (SD1602)" of the new control system CPU module. \*3

By reading the SD1602 value from the new control system CPU module, which the SP.CONTSW instruction was used for system switching can be confirmed.

\*2 : The Ⓢ value specified for the SP.CONTSW instruction can be confirmed in the error common information of the PLC diagnostics dialog box on GX Developer.

\*3 : The new control system CPU module means the CPU module that was switched from the standby system to the control system by the SP.CONTSW instruction.



- (5) The error completion device is turned ON by the control system CPU module when system switching by the SP.CONTSW instruction was unsuccessful.
- (a) When OPERATION ERROR is detected due to any of the following reasons at the execution of the SP.CONTSW instruction, the error completion device is turned ON during the instruction execution.
- 0 is specified at  $\textcircled{S}$  of the executed SP.CONTSW instruction.
  - The "manual switching enable flag (SM1592)" is OFF.
  - The SP.CONTSW instruction was executed by the standby system in the separate mode.
  - The SP.CONTSW instruction was executed in the debug mode.
- (b) If systems could not be switched due to any of the reasons given in the following table, the error completion device turns ON when system switching is executed in the END processing.

Reason No.	Reasons for System Switching Failure
0	Normally completed
1	Tracking cable is disconnected or faulty.
2	Hardware fault, power-off, reset or watchdog timer error occurred in the standby system.
3	Watchdog timer error occurred in the control system.
4	Preparations being made for tracking transfer.
5	Communication time-out.
6	Stop error occurred in the standby system. (Excluding watchdog timer error)
7	Operating status different between the control system and standby system.
8	Memory copy being executed from the control system to the standby system.
9	Write during RUN being executed.
10	Network fault detected by the standby system.

When the error completion device was turned ON due to unsuccessful system switching, 16 is stored into the "reason(s) for system switching (SD1588)" and the reason No. of the above table into the "reason(s) for system switching failure (SD1589)".

- (6) Use a user program or GX Developer to turn OFF the error completion bit that has turned ON.
- If normal system switching is performed by the execution of the SP.CONTSW instruction with the error completion device ON, the error completion device of the new standby system CPU module is also turned OFF.
- When system switching is performed due to a factor other than the SP.CONTSW instruction, however, the error completion device is not turned OFF.

## Operation Error

- (1) In any of the following cases, an operation error occurs, the error flag (SM0) turns ON, and an error code is stored into SD0.
- The value specified at  $\textcircled{S}$  is 0 at execution of the SP.CONTSW instruction. (Error code: 4100)
  - The manual switching enable flag (SM1592) is OFF (disable) at execution of the SP.CONTSW instruction. (Error code: 4120)
  - The SP.CONTSW instruction was executed by the standby system CPU module in the separate mode. (Error code: 4121)
  - The SP.CONTSW instruction was executed in the debug mode. (Error code: 4121)

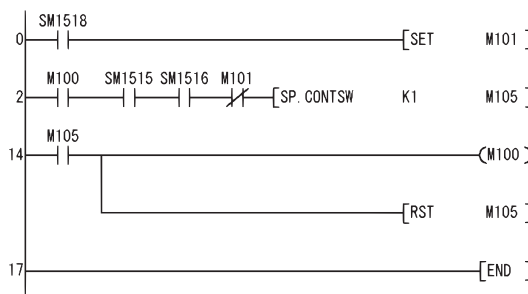
- (2) If system switching was unsuccessful, the error flag (SM0) is turned ON and an error code is stored into SD0.
- The tracking cable is disconnected or faulty. (Error code: 6220)
  - Hardware fault, power-off, reset or watchdog timer error occurred in the standby system. (Error code: 6220)
  - Watchdog timer error occurred in the control system. (Error code: 6220)
  - Preparations are being made for tracking transfer. (Error code: 6220)
  - Communication time-out occurred. (Error code: 6220)
  - Stop error, excluding watchdog timer error, occurred in the standby system. (Error code: 6220)
  - The operating status differs between the control system and standby system. (Error code: 6220)
  - Memory copy is being executed from the control system to the standby system. (Error code: 6220)
  - Write during RUN is being executed. (Error code: 6220)
  - Network fault was detected by the standby system. (Error code: 6220)

## Program Example

- (1) The following program executes system switching on the leading edge of the system switching command (M100).

If the system switching command (M100) remains ON, the SP.CONTSW instruction is also executed by the new control system CPU module after system switching. Therefore, M101 is added to the execution conditions as a consecutive switching prevention flag.

[Ladder Mode]



[List Mode]

Step	Instruction	Device
0	LD	SM1518
1	SET	M101
2	LD	M100
3	AND	SM1515
4	AND	SM1516
5	ANI	M101
6	SP. CONTSW	K1 M105
14	LD	M105
15	OUT	M100
16	RST	M105
17	END	



# ERROR CODES

---



## 12.1 Error Code List

---

The CPU module uses the self diagnostics function to display error information (on the LED) and stores the information into the special relay SM and special register SD, when an error occurs in the following situations:

- When the Programmable Controller is powered ON.
- When the CPU module is switched from STOP to RUN.
- While the CPU module is running.

If an error occurs when a communication request is issued from the peripheral device, intelligent function module or network system to the CPU module, the CPU module returns the error code (4000<sub>H</sub> to 4FFF<sub>H</sub>) to the request source.

The following describes the description of errors which occur in the CPU module and the corrective actions for the errors.

### (1) How to read the error code list

The following describes how to read Section 12.1.3 Error code list (1000 to 1999) to Section 12.1.9 Error code list (7000 to 10000).

#### (a) Error code, common information and individual information

Alphanumeric characters in the parentheses of the titles indicate the special register numbers where each information is stored.

#### (b) Compatible CPU

QCPU	: Indicates all the Q series CPU modules.
Q00J/Q00/Q01	: Indicates the Basic model QCPU.
Qn(H)	: Indicates the High Performance model QCPU.
QnPH	: Indicates the Process CPU.
QnPRH	: Indicates the Redundant CPU.
QnU	: Indicates the Universal model QCPU.
Each CPU module model name	: Indicates the relevant specific CPU module. (Example: Q02U)

## 12.1.1 Error codes

Errors are detected by the self diagnostic function of the CPU module or detected during communication with the CPU module.

The relation between the error detection pattern, error detection location and error code is shown in Table 12.1.

Table 12.1 Reference destination

Error detection pattern	Error detection location	Error code	Reference
Detection by the self diagnostics function of CPU module	CPU module	1000 to 10000*1*2	Section 12.1.3 to 12.1.9
Detection at communication with CPU module	CPU module	4000H to 4FFFH	• QCPU User's Manual (Hardware design, Maintenance and Inspection)
	Serial communication module, etc.	7000H to 7FFFH	Serial Communication User's Manual, etc.
	CC-Link module	B000H to BFFFH	CC-Link System Master/Local Module User's Manual
	Ethernet module	C000H to CFFFH	Ethernet Interface Module User's Manual
	CC-Link IE controller network	E000H to EFFFH	CC-Link IE Controller Network Reference Manual
	MELSECNET/H network module	F000H to FFFFH	• MELSECNET/H mode Q Corresponding MELSECNET/H Network System Reference Manual • MELSECNET/10 mode For QnA/Q4AR MELSECNET/10 Network System Reference Manual

\*1: CPU module error codes are classified into minor, moderate, major errors as shown below.

- Minor error: Errors that may allow the CPU module to continue the operation, e.g., battery error. (Error code: 1300 to 10000)
- Moderate error: Errors that may cause the CPU module to stop the operation, e.g., WDT error. (Error code: 1300 to 10000)
- Major error: Errors that may cause the CPU module to stop the operation, e.g., RAM error. (Error code: 1000 to 1299)

Determine the error level, i.e. whether the operation can be continued or stopped, by referring to "Operating Statuses of CPU" described in Section 12.1.3 to 12.1.9 "Error Code List"

\*2: When detected an error code without being noted in the reference table, please contact your local Mitsubishi representative.

## 12.1.2 Reading an error code

When an error occurs, reading an error code, error message or the like can be executed with GX Developer.

For the details of the operation method, refer to the operating manual for GX Developer.

## 12.1.3 Error code list (1000 to 1999)

The following shows the error messages from the error code 1000 to 1999, the contents and causes of the errors, and the corrective actions for the errors.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1000	<p><b>[MAIN CPU DOWN]</b> Runaway or failure of CPU module or failure of main CPU</p> <ul style="list-style-type: none"> <li>• Malfunctioning due to noise or other reason</li> <li>• Hardware fault</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:--</li> <li>• Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again.If the same error is displayed again, this suggests a CPU module hardware fault.(Contact your local Mitsubishi representative.)</li> </ul>		
1001	<p><b>[MAIN CPU DOWN]</b> Runaway or failure of CPU module or failure of main CPU</p> <ul style="list-style-type: none"> <li>• Malfunctioning due to noise or other reason</li> <li>• Hardware fault</li> <li>• Accessed to outlying devices with the device range checks disabled (SM237 is turned on)(This error occurs only when BMOV, FMOV, and DFMOV instructions are executed.) (Universal model QCPU only)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:--</li> <li>• Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again.If the same error is displayed again, this suggests a CPU module hardware fault.(Contact your local Mitsubishi representative.)</li> <li>• Check the devices specified by BMOV, FMOV, and DFMOV instructions and correct the device settings. (Universal model QCPU only)</li> </ul>	RUN: Off	QCPU
1002	<p><b>[MAIN CPU DOWN]</b> Runaway or failure of CPU module or failure of main CPU</p> <ul style="list-style-type: none"> <li>• Malfunctioning due to noise or other reason</li> <li>• Hardware fault</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:--</li> <li>• Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again.If the same error is displayed again, this suggests a CPU module hardware fault.(Contact your local Mitsubishi representative.)</li> </ul>	ERR.: Flicker	
1003				
1004	<p><b>[MAIN CPU DOWN]</b> Runaway or failure of CPU module or failure of main CPU</p> <ul style="list-style-type: none"> <li>• Malfunctioning due to noise or other reason</li> <li>• Hardware fault</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:--</li> <li>• Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again.If the same error is displayed again, this suggests a CPU module hardware fault.(Contact your local Mitsubishi representative.)</li> </ul>	CPU Status: Stop	
1005	<p><b>[MAIN CPU DOWN]</b> Runaway or failure of CPU module or failure of main CPU</p> <ul style="list-style-type: none"> <li>• Malfunctioning due to noise or other reason</li> <li>• Hardware fault</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:--</li> <li>• Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again.If the same error is displayed again, this suggests a CPU module hardware fault.(Contact your local Mitsubishi representative.)</li> </ul>		
	<p><b>[MAIN CPU DOWN]</b> Boot operation was performed in the transfer destination without formatting.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:--</li> <li>• Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON</li> </ul>	<ul style="list-style-type: none"> <li>• Before performing boot operation by the parameter, select "Clear program memory" to clear the program memory.</li> </ul>		Qn(H) QnPH QnPRH

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1006	<b>[MAIN CPU DOWN]</b>			
1007	Runaway or failure of CPU module or failure of main CPU			
1008	<ul style="list-style-type: none"> <li>Malfunctioning due to noise or other reason</li> <li>Hardware fault</li> </ul> <b>■Collateral information</b> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <b>■Diagnostic Timing</b> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Take noise reduction measures.</li> <li>Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		Qn(H) QnPH QnPRH
1009	<b>[MAIN CPU DOWN]</b> <ul style="list-style-type: none"> <li>A failure is detected on the power supply module, CPU module, main base unit, extension base unit or extension cable.</li> <li>When using the redundant base unit, the redundant power supply module failure in both systems and/or the redundant base unit failure are detected.</li> </ul> <b>■Collateral information</b> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <b>■Diagnostic Timing</b> <ul style="list-style-type: none"> <li>Always</li> </ul>	Reset the CPU module and RUN it again. If the same error is detected again, it is considered that the power supply module, CPU module, main base unit, extension base unit or extension cable is faulty. (Contact your local Mitsubishi representative.)		Q00J/Q00/Q01* <sup>4</sup> Qn(H) <sup>6</sup> QnPH QnPRH QnU
1010	<b>[END NOT EXECUTE]</b> Entire program was executed without the execution of an END instruction. <ul style="list-style-type: none"> <li>When the END instruction is executed it is read as another instruction code, e.g. due to noise.</li> <li>The END instruction has been changed to another instruction code somehow.</li> </ul> <b>■Collateral information</b> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <b>■Diagnostic Timing</b> <ul style="list-style-type: none"> <li>When an END instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>Take noise reduction measures.</li> <li>Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	QCPU
1020	<b>[SFCP. END ERROR]</b> The SFC program cannot be normally terminated due to noise or other reason. <ul style="list-style-type: none"> <li>The SFC program cannot be normally terminated due to noise or any similar cause.</li> <li>The SFC program cannot be normally terminated for any other reason.</li> </ul> <b>■Collateral information</b> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <b>■Diagnostic Timing</b> <ul style="list-style-type: none"> <li>When SFC program is executed</li> </ul>	<ul style="list-style-type: none"> <li>Take noise reduction measures.</li> <li>Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		Q00J/Q00/Q01* <sup>4</sup> QnPH QnU
1035	<b>[MAIN CPU DOWN]</b> Runaway or error of the CPU module was detected. <ul style="list-style-type: none"> <li>Malfunction due to noise etc.</li> <li>Hardware failure</li> </ul> <b>■Collateral information</b> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <b>■Diagnostic Timing</b> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Take measures against noise.</li> <li>Reset the CPU module and run it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		QnU

\*4 Function version is B or later.

\*6 The module whose first 5 digits of serial No. is "04101" or later.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1101	<p><b>[RAM ERROR]</b> The sequence program storing program memory in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset/ When an END instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		
1102	<p><b>[RAM ERROR]</b></p> <ul style="list-style-type: none"> <li>• The work area RAM in the CPU module is faulty.</li> <li>• The standard RAM and extended RAM in the CPU module are faulty.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset/ When an END instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU
1103	<p><b>[RAM ERROR]</b> The device memory in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• When indexing is performed, check the value of index register to see if it is within the device range.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	
	<p><b>[RAM ERROR]</b></p> <ul style="list-style-type: none"> <li>• The device memory in the CPU module is faulty.</li> <li>• The device out of range is accessed due to indexing, and the device for system is overwritten.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset/ When an END instruction executed</li> </ul>			Qn(H) <sup>*8</sup> QnPH <sup>*8</sup> QnPRH <sup>*9</sup>
1104	<p><b>[RAM ERROR]</b> The address RAM in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU
1105	<p><b>[RAM ERROR]</b> The CPU memory in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, this suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		Q00J/Q00/Q01 QnU
	<p><b>[RAM ERROR]</b> The CPU shared memory in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>			Qn(H) <sup>*4</sup> QnPH QnPRH QnU

\*4 Function version is B or later.

\*8 The module whose first 5 digits of serial No. is "08032" or later.

\*9 The module whose first 5 digits of serial No. is "09012" or later.



Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU	
1106	<p><b>[RAM ERROR]</b> The battery is dead. The program memory in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN/When an END instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Check the battery to see if it is dead or not. If dead, replace the battery.</li> <li>• Take noise reduction measures.</li> <li>• Format the program memory, write all files to the PLC, then reset the CPU module, and RUN it again.</li> </ul> <p>If the same error is displayed again, the possible cause is a CPU module hardware fault. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	Qn(H) QnPH <sup>*7</sup> QnPRH	
1107	<p><b>[RAM ERROR]</b> The work area RAM in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<p>This suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</p>		QnPRH	
1108	<p><b>[RAM ERROR]</b> The work area RAM in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>			Qn(H) <sup>*8</sup> QnPH <sup>*8</sup> QnPRH <sup>*9</sup>	
1109	<p><b>[RAM ERROR]</b> The work area RAM in the CPU module is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>			<p>This suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</p>	QnPRH
1110	<p><b>[TRK. CIR. ERROR]</b> A fault was detected by the initial check of the tracking hardware.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>				
1111	<p><b>[TRK. CIR. ERROR]</b> A tracking hardware fault was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>				
1112	<p><b>[TRK. CIR. ERROR]</b> A tracking hardware fault was detected during running.</p> <ul style="list-style-type: none"> <li>• The tracking cable was disconnected and reinserted without the standby system being powered off or reset.</li> <li>• The tracking cable is not secured by the connector fixing screws.</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• During running</li> </ul>		<ul style="list-style-type: none"> <li>• Start after checking that the tracking cable is connected.</li> </ul> <p>If the same error is displayed again, the cause is the hardware fault of the tracking cable or CPU module. (Please contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p> <ul style="list-style-type: none"> <li>• Confirm the redundant system startup procedure, and execute a startup again. For details, refer to the QnPRHCPU User's Manual (Redundant System).</li> </ul>	QnPRH	
1113	<p><b>[TRK. CIR. ERROR]</b> A tracking hardware fault was detected during running.</p> <ul style="list-style-type: none"> <li>• The tracking cable was disconnected and reinserted without the standby system being powered off or reset.</li> <li>• The tracking cable is not secured by the connector fixing screws.</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• During running</li> </ul>	<ul style="list-style-type: none"> <li>• Start after checking that the tracking cable is connected.</li> </ul> <p>If the same error is displayed again, the cause is the hardware fault of the tracking cable or CPU module. (Please contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p> <ul style="list-style-type: none"> <li>• Confirm the redundant system startup procedure, and execute a startup again. For details, refer to the QnPRHCPU User's Manual (Redundant System).</li> </ul>	QnPRH		

\*7 The module whose first 5 digits of serial No. is "07032" or later.

\*8 The module whose first 5 digits of serial No. is "08032" or later.

\*9 The module whose first 5 digits of serial No. is "09012" or later.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1115	<p><b>[TRK. CIR. ERROR]</b> A fault was detected by the initial check of the tracking hardware.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<p>This suggests a CPU module hardware fault. (Contact your nearest Mitsubishi representative.)</p>		
1116	<p><b>[TRK. CIR. ERROR]</b> A tracking hardware fault was detected during running.</p> <ul style="list-style-type: none"> <li>• The tracking cable was disconnected and reinserted without the standby system being powered off or reset.</li> <li>• The tracking cable is not secured by the connector fixing screws.</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• During running</li> </ul>	<ul style="list-style-type: none"> <li>• Start after checking that the tracking cable is connected.</li> </ul> <p>If the same error is displayed again, the cause is the hardware fault of the tracking cable or CPU module. (Please contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p> <ul style="list-style-type: none"> <li>• Confirm the redundant system startup procedure, and execute a startup again. For details, refer to the QnPRHCPU User's Manual (Redundant System).</li> </ul>		QnPRH
1150	<p><b>[RAM ERROR]</b> The memory of the CPU module in the Multiple CPU high speed transmission area is faulty.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Reset the CPU module and RUN it again.</li> </ul> <p>If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative, explaining a detailed description of the problem.</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnU*10
1160	<p><b>[RAM ERROR]</b> The program memory in the CPU module is overwritten.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At program execution</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Format the program memory, write all files to the PLC, then reset the CPU module, and RUN it again.</li> </ul> <p>If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative, explaining a detailed description of the problem.</p>		
1161	<p><b>[RAM ERROR]</b> The data of the device memory built in the CPU module is overwritten.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At program execution</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> </ul> <p>If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative, explaining a detailed description of the problem.</p>		QnU
1162	<p><b>[RAM ERROR]</b> The error of the data held by the battery in the CPU module is detected. (It occurs when the automatic format is not set.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Change the CPU main body or SRAM card battery.</li> </ul> <p>If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative, explaining a detailed description of the problem.</p>		

\*10 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1164	<p><b>[RAM ERROR]</b> The destruction of the data stored in the standard RAM is detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> </ul> <p>If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative, explaining a detailed description of the problem.</p>		QnU*11
1200	<p><b>[OPE. CIRCUIT ERR.]</b> The operation circuit for index modification in the CPU module does not operate normally.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<p>This suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	QCPU
1201	<p><b>[OPE. CIRCUIT ERR.]</b> The hardware (logic) in the CPU module does not operate normally.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>			
1202	<p><b>[OPE. CIRCUIT ERR.]</b> The operation circuit for sequence processing in the CPU module does not operate normally.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>			
1203	<p><b>[OPE. CIRCUIT ERR.]</b> The operation circuit for index modification in the CPU module does not operate normally.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>			QnPRH
1204	<p><b>[OPE. CIRCUIT ERR.]</b> The hardware (logic) in the CPU module does not operate normally.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>			
1205	<p><b>[OPE. CIRCUIT ERR.]</b> The operation circuit for sequence processing in the CPU module does not operate normally.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>			

\*11 The Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, .Q26UD(E)HCPU only.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1300	<p><b>[FUSE BREAK OFF]</b> There is an output module with a blown fuse.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(Slot No.) [For Remote I/O network]Network No./Station No.</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Check FUSE. LED of the output modules and replace the module whose LED is lit. (The module with a blown fuse can also be identified using GX Developer. Check the special registers SD1300 to SD1331 to see if the bit corresponding to the module is "1".)</li> <li>When a GOT is bus-connected to the main base unit or extension base unit, check the connection status of the extension cable and the earth status of the GOT.</li> </ul>	<p>RUN: Off/On</p> <p>ERR.: Flicker/On</p>	<p>Qn(H) QnPH QnPRH QnU</p>
	<p><b>[FUSE BREAK OFF]</b> There is an output module with a blown fuse.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(Slot No.) [For Remote I/O network]Network No./Station No.</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<p>Check ERR. LED of the output modules and replace the module whose LED is lit. (The module with a blown fuse can also be identified using GX Developer. Check the special registers SD130 to SD137 to see if the bit corresponding to the module is "1".)</p>	<p>CPU Status: Stop/ Continue*1</p>	<p>Q00J/Q00/Q01</p>
1310	<p><b>[I/O INT. ERROR]</b> An interruption has occurred although there is no interrupt module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>During interrupt</li> </ul>	<p>Any of the mounted modules is experiencing a hardware fault. Therefore, check the mounted modules and change the faulty module. (Contact your local Mitsubishi representative.)</p>		<p>QCPU</p>
1311	<p><b>[I/O INT. ERROR]</b> An interrupt request from other than the interrupt module was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>During interrupt</li> </ul>	<p>Take action so that an interrupt will not be issued from other than the interrupt module.</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>Q00J/Q00/Q01*4 QnU</p>
	<p><b>[I/O INT. ERROR]</b> An interrupt request from the module where interrupt pointer setting has not been made in the PLC parameter dialog box was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>During interrupt</li> </ul>	<ul style="list-style-type: none"> <li>Correct the interrupt pointer setting in the PLC system setting of the PLC parameter dialog box.</li> <li>Take measures so that an interrupt is not issued from the module where the interrupt pointer setting in the PLC system setting of the PLC parameter dialog box has not been made. Correct the interrupt setting of the network parameter. Correct the interrupt setting of the intelligent function module buffer memory. Correct the basic program of the QD51.</li> </ul>		<p>Q00J/Q00/Q01*5 QnPRH QnU</p>
1320	<p><b>[LAN CTRL.DOWN]</b> The H/W self-diagnostics detected a LAN controller failure.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	<p>This suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>QnU*13</p>
1321	<p><b>[LAN CTRL.DOWN]</b> The H/W self-diagnostics detected a LAN controller failure.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	<p>This suggests a CPU module hardware fault. (Contact your local Mitsubishi representative.)</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>QnU*13</p>

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*4 Function version is B or later.

\*5 Function version is A.

\*13 This applies to the Built-in Ethernet port QCPU.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1401	<p><b>[SP. UNIT DOWN]</b></p> <ul style="list-style-type: none"> <li>There was no response from the intelligent function module/special function module in the initial processing.</li> <li>The size of the buffer memory of the intelligent function module/special function module is invalid.</li> <li>The unsupported module is mounted.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/ At reset/When intelligent function module is accessed</li> </ul>	<p>When the unsupported module is mounted, remove it.</p> <p>When the corresponding module is supported, this suggests the intelligent function module/special function module, CPU module and/or base unit is expecting a hardware fault (Contact your local Mitsubishi representative.)</p>		
1402	<p><b>[SP. UNIT DOWN]</b></p> <p>The intelligent function module/special function module was accessed in the program, but there was no response.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When an intelligent function module access instruction is executed</li> </ul>	<p>This suggests the intelligent function module/special function module , CPU module and/or base unit is expecting a hardware fault (Contact your local Mitsubishi representative.)</p>	<p>RUN: Off/On</p> <p>ERR.: Flicker/On</p>	
1403	<p><b>[SP. UNIT DOWN]</b></p> <ul style="list-style-type: none"> <li>The unsupported module is mounted.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When an END instruction executed</li> </ul>	<p>When the unsupported module is mounted, remove it.</p> <p>When the corresponding module is supported, this suggests the intelligent function module/special function module , CPU module and/or base unit is expecting a hardware fault (Contact your local Mitsubishi representative.)</p>	<p>CPU Status: Stop/ Continue*2</p>	QCPU
	<p><b>[SP. UNIT DOWN]</b></p> <ul style="list-style-type: none"> <li>There was no response from the intelligent function module/special function module when the END instruction is executed.</li> <li>An error is detected at the intelligent function module/special function module.</li> <li>The I/O module (intelligent function module/special function module) is nearly removed, completely removed, or mounted during running.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<p>The CPU module, base module and/or the intelligent function module/special function module that was accessed is experiencing a hardware fault. (Contact your local Mitsubishi representative.)</p>		
1411	<p><b>[CONTROL-BUS. ERR.]</b></p> <p>When performing a parameter I/O allocation the intelligent function module/special function module could not be accessed during initial communications.</p> <p>(On error occurring, the head I/O number of the corresponding intelligent function module/special function module is stored in the common information.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	<p>Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module/special function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)</p>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	

\*2 In the QCPU, either error stop or continue can be selected for each intelligent function module by the parameters.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1412	<p><b>[CONTROL-BUS. ERR.]</b> The FROM/TO instruction is not executable, due to a control bus error with the intelligent function module/special function module. (On error occurring, the program error location is stored in the individual information.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• During execution of FROM/TO instruction set</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module/special function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)	RUN: Off ERR.: Flicker  CPU Status: Stop	QCPU
1413	<p><b>[CONTROL-BUS. ERR.]</b> In a multiple CPU system, a CPU module incompatible with the multiple CPU system is mounted.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Remove the CPU module incompatible with the multiple CPU system from the main base unit, or replace the CPU module incompatible with the multiple CPU system with a CPU module compatible with the multiple CPU system.</li> <li>• The intelligent function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)</li> </ul>		Q00J/Q00/Q01*4 Qn(H)*4 QnPH
	<p><b>[CONTROL-BUS. ERR.]</b> An error is detected on the system bus.</p> <ul style="list-style-type: none"> <li>• Self-diagnosis error of the system bus.</li> <li>• Self-diagnosis error of the CPU module</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)		QCPU
1414	<p><b>[CONTROL-BUS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• Fault of a loaded module was detected.</li> <li>• In a multiple CPU system, a CPU module incompatible with the multiple CPU system is mounted.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Remove the CPU module incompatible with the multiple CPU system from the main base unit, or replace the CPU module with a CPU module compatible with the multiple CPU system.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)</li> </ul>		Q00J/Q00/Q01*4 Qn(H)*4 QnPH QnU
	<p><b>[CONTROL-BUS. ERR.]</b> An error is detected on the system bus.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)	Q00J/Q00/Q01*4 Qn(H) QnPH QnPRH QnU	

\*4 Function version is B or later.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1415	<p><b>[CONTROL-BUS. ERR.]</b> Fault of the main or extension base unit was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)		Q00J/Q00/Q01 Qn(H) <sup>*4</sup> QnPH QnPRH QnU
	<p><b>[CONTROL-BUS. ERR.]</b> Fault of the main or extension base unit was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ When an END instruction executed</li> </ul>			Qn(H) <sup>*8</sup> QnPH <sup>*8</sup>
1416	<p><b>[CONTROL-BUS. ERR.]</b> System bus fault was detected at power-on or reset.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the intelligent function module, CPU module or base unit is faulty. (Contact your local Mitsubishi representative.)	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*4</sup> QnPH QnU
	<p><b>[CONTROL-BUS. ERR.]</b> In a multiple CPU system, a bus fault was detected at power-on or reset.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>			Q00/Q01 <sup>*4</sup> QnU
1417	<p><b>[CONTROL-BUS. ERR.]</b> A reset signal error was detected on the system bus.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>			QnPRH
1418	<p><b>[CONTROL-BUS.ERR.]</b> In the redundant system, at power-on/reset or switching system, the control system cannot access the extension base unit since it failed to acquire the access right.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ At Switching execution</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module, the Q6□WRB, or hardware of extension cable is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		QnPRH <sup>*9</sup>
1430	<p><b>[MULTI-C.BUS ERR.]</b> The error of host CPU is detected in the Multiple CPU high speed bus.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		QnU <sup>*10</sup>

\*4 Function version is B or later.

\*8 The module whose first 5 digits of serial No. is "08032" or later.

\*9 The module whose first 5 digits of serial No. is "09012" or later.

\*10 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1431	<p><b>[MULTI-C.BUS ERR.]</b> The communication error with other CPU is detected in the Multiple CPU high speed bus.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Check the main base unit mounting status of the CPU module.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	QnU* <sup>10</sup>
1432	<p><b>[MULTI-C.BUS ERR.]</b> The communication time out with other CPU is detected in the Multiple CPU high speed bus.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		
1433	<p><b>[MULTI-C.BUS ERR.]</b> The communication error with other CPU is detected in the Multiple CPU high speed bus.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Check the main base unit mounting status of the CPU module.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		
1434				
1436	<p><b>[MULTI-C.BUS ERR.]</b> The error of the Multiple CPU high speed main base unit is detected. (The error of the Multiple CPU high speed bus is detected.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		
1437	<p><b>[MULTI-C.BUS ERR.]</b> The error of the Multiple CPU high speed main base unit is detected. (The error of the Multiple CPU high speed bus is detected.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Take noise reduction measures.</li> <li>• Check the main base unit mounting status of the CPU module.</li> <li>• Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		
1439	<p><b>[MULTI-C.BUS ERR.]</b> An error of the multiple CPU high speed main base unit was detected. (An error of the multiple CPU high speed bus was detected.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	Reset the CPU module and RUN it again. If the same error is displayed again, the CPU module has hardware failure. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		
1500	<p><b>[AC/DC DOWN]</b></p> <ul style="list-style-type: none"> <li>• A momentary power supply interruption has occurred.</li> <li>• The power supply went off.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Check the power supply.	RUN: On ERR.: Off  CPU Status: Continue	QCPU

\*10 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.



Error Code	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
1510	<p><b>[SINGLE PS. DOWN]</b> The power supply voltage of either of redundant power supply modules on the redundant base unit dropped.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Base No./ Power supply No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Check the power supplied to the redundant power supply modules mounted on the redundant base unit.	RUN: On ERR.: On	Qn(H) <sup>*6</sup> QnPH <sup>*6</sup> QnPRH QnU <sup>*12</sup>
1520	<p><b>[SINGLE PS. ERROR]</b> On the redundant base unit, the one damaged redundant power supply module was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Base No./ Power supply No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Hardware fault of the redundant power supply module. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)	CPU Status: Continue	
1600	<p><b>[BATTERY ERROR<sup>*3</sup>]</b></p> <ul style="list-style-type: none"> <li>• The battery voltage in the CPU module has dropped below stipulated level.</li> <li>• The lead connector of the CPU module battery is not connected.</li> <li>• The lead connector of the CPU module battery is not securely engaged.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Drive Name</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Change the battery.</li> <li>• If the battery is for program memory, standard RAM or for the back-up power function, install a lead connector.</li> <li>• Check the lead connector of the CPU module for looseness. Firmly engage the connector if it is loose.</li> </ul>	RUN: On ERR.: Off	QCPU
1601	<p><b>[BATTERY ERROR<sup>*3</sup>]</b> Voltage of the battery on memory card has dropped below stipulated level.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Drive Name</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Change the battery.	CPU Status: Continue	Qn(H) QnPH QnPRH QnU <sup>*14</sup>
1610	<p><b>[FLASH ROM ERROR]</b> The number of writing to flash ROM (standard ROM and system securement area) exceeds 100,000 times. (Number of writings &gt;100,000 times)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When writing to ROM</li> </ul>	Change the CPU module.	RUN: On ERR.: On  CPU Status: Continue	QnU

\*3 BAT. LED is displayed at BATTERY ERROR.

\*6 The module whose first 5 digits of serial No. is "04101" or later.

\*12 The module whose first 5 digits of serial No. is "10042" or later.

\*14 The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

## 12.1.4 Error code list (2000 to 2999)

The following shows the error messages from the error code 2000 to 2999, the contents and causes of the errors, and the corrective actions for the errors.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2000	<p><b>[UNIT VERIFY ERR.]</b> In a multiple CPU system, a CPU module incompatible with the multiple CPU system is mounted.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.) [For Remote I/O network] Network No./Station No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When an END instruction executed</li> </ul>	Replace the CPU module incompatible with the multiple CPU system with a CPU module compatible with the multiple CPU system.		Qn(H) <sup>*3</sup> QnPH
	<p><b>[UNIT VERIFY ERR.]</b> The I/O module status is different from the I/O module information at power ON.</p> <ul style="list-style-type: none"> <li>I/O module (or intelligent function module) is not installed properly or installed on the base unit.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.) [For Remote I/O network] Network No./Station No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When an END instruction executed</li> </ul>	<p>Read the error common information at the GX Developer, and check and/or change the module that corresponds to the numerical value (module number) there.</p> <p>Alternatively, monitor special registers SD150 to SD157 using GX Developer, and check and replace the module where the bit of its data is "1".</p>	<p>RUN: Off/On</p> <p>ERR.: Flicker/On</p> <p>CPU Status: Stop/ Continue<sup>*1</sup></p>	Q00J/Q00/Q01
	<p><b>[UNIT VERIFY ERR.]</b> I/O module information power ON is changed.</p> <ul style="list-style-type: none"> <li>I/O module (or intelligent function module/special function module) not installed properly or installed on the base unit.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.) [For Remote I/O network] Network No./Station No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When an END instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>Read the common information of the error using the peripheral device, and check and/or change the module that corresponds to the numerical value (module number) there.</li> <li>Alternatively, monitor the special registers SD1400 to SD1431 at a peripheral device, and change the fuse at the output module whose bit has a value of "1".</li> <li>When a GOT is bus-connected to the main base unit or extension base unit, check the connection status of the extension cable and the grounding status of the GOT.</li> </ul>		Qn(H) QnPH QnPRH QnU
2001	<p><b>[UNIT VERIFY ERR.]</b> During operation, a module was mounted on the slot where the empty setting of the CPU module was made.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (CPU No.)</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When an END instruction executed</li> </ul>	During operation, do not mount a module on the slot where the empty setting of the CPU module was made.	<p>RUN: Off/On</p> <p>ERR.: Flicker/On</p> <p>CPU Status: Stop/ Continue<sup>*2</sup></p>	Q00J/Q00/Q01 <sup>*3</sup> QnU
2010	<p><b>[BASE LAY ERROR]</b></p> <ul style="list-style-type: none"> <li>More than applicable number of extension base units have been used.</li> <li>When a GOT was bus-connected, the CPU module was reset while the power of the GOT was OFF.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Base No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Use the allowable number of extension base units or less.</li> <li>Power on the Programmable Controller and GOT again.</li> </ul>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	Q00J/Q00/Q01 <sup>*3</sup> QnPRH Q00UJ Q00U/Q01U Q02U

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*2 Either error stop or continue can be selected for each module by the parameters.

\*3 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2011	<p><b>[BASE LAY ERROR]</b> The QA1S6□B, QA6□B, or QA6ADP+A5□B/A6□B was used as the base unit.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Base No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Do not use the QA1S6□B, QA6□B, or QA6ADP+A5□B/A6□B as the base unit.		Q00J/Q00/Q01* <sup>3</sup> QnPH QnPRH QnU
2012	<p><b>[BASE LAY ERROR]</b> The GOT is bus-connected to the main base unit of the redundant system. The following errors are detected in the CPU redundant system compatible with the extension base unit.</p> <ul style="list-style-type: none"> <li>• The base unit other than the Q6□WRB is connected to the extension stage No.1.</li> <li>• The base unit is connected to any one of the extension stages No.2 to No.7, although the Q6□WRB does not exist in the extension stage No.1 .</li> <li>• The other system CPU module is incompatible with the extension base unit.</li> <li>• The Q5□B, QA1S6□B, QA6□B or QA6ADP+A5□B/A6□B is connected.</li> <li>• The number of slots of the main base unit for both systems is different. Information of the Q6□WRB cannot be read correctly.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Base No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Remove a bus connection cable for GOT connection connected to the main base unit.</li> <li>• Use the Q6□WRB (fixed to the extension stage No.1)</li> <li>• Use the CPU module compatible with the extension base unit for the other system.</li> <li>• Do not use the Q5□B, QA1S6□B, QA6□B or QA6ADP+A5□B/A6□B for the base unit.</li> <li>• Use the main base unit which has the same number of slots.</li> <li>• Hardware failure of the Q6□WRB. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnPRH* <sup>6</sup>
2013	<p><b>[BASE LAY ERROR]</b> Stage number of the Q6□WRB is recognized as other than extension stage No.1 in the redundant system.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Base No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Hardware failure of the Q6□WRB. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		
2020	<p><b>[EXT.CABLE ERR.]</b> The following errors are detected in the redundant system.</p> <ul style="list-style-type: none"> <li>• At power-on/reset, the standby system has detected the error in the path between the control system and the Q6□WRB.</li> <li>• The standby system has detected the error in the path between the host system CPU and the Q6□WRB at END processing.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/At reset/ When an END instruction executed</li> </ul>	<p>Check to see if the extension cable between the main base unit and the Q6□WRB is connected correctly. If not, connect it after turning OFF the main base unit where the extension cable will be connected.</p> <p>If the cable is connected correctly, hardware of the CPU module, Q6□WRB, or extension cable is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p>		QnPRH* <sup>6</sup>

\*3 The function version is B or later.

\*6 The module whose first 5 digits of serial No. is "09012" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2100	<p><b>[SP. UNIT LAY ERR.]</b> The slot to which the QI60 is mounted is set to other than Inteli (intelligent function module) or Interrupt (interrupt module) in the I/O assignment of PLC parameter.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Make setting again to match the PLC parameter I/O assignment with the actual loading status.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*3</sup> QnPH QnPRH
	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>• In the I/O assignment setting of PLC parameter, Inteli (intelligent function module) was allocated to an I/O module or vice versa.</li> <li>• In the I/O assignment setting of PLC parameter, a module other than CPU (or nothing) was allocated to the location of a CPU module or vice versa.</li> <li>• In the I/O assignment setting of the PLC parameter, switch setting was made to the module that has no switch setting.</li> <li>• In the I/O assignment setting of the PLC parameter dialog box, the number of points assigned to the intelligent function module is less than the number of points of the mounted module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Make the PLC parameter's I/O assignment setting again so it is consistent with the actual status of the intelligent function module and the CPU module.</li> <li>• Delete the switch setting in the I/O assignment setting of the PLC parameter.</li> </ul>		Qn(H) QnPH QnPRH QnU
	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>• In the parameter I/O allocation settings, an Inteli (intelligent function module) was allocated to a location reserved for an I/O module or vice versa.</li> <li>• In the parameter I/O allocation settings, a module other than CPU (or nothing) was allocated to a location reserved for a CPU module or vice versa.</li> <li>• In the I/O assignment setting of the PLC parameter dialog box, the number of points assigned to the intelligent function module is less than the number of points of the mounted module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Reset the parameter I/O allocation setting to conform to the actual status of the intelligent function module and the CPU module.		Q00J/Q00/Q01
2101	<p><b>[SP. UNIT LAY ERR.]</b> 13 or more A-series special function modules (except for the A1SI61) that can initiate an interrupt to the CPU module have been installed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Reduce the A series special function modules (except the A1SI61) that can make an interrupt start to the CPU module to 12 or less.	Qn(H)	

\*3 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2102	<p><b>[SP. UNIT LAY ERR.]</b> Seven or more A1SD51S have been installed.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Keep the number of A1SD51S to six or fewer.		Qn(H)
2103	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>• Two or more QI60/A1SI61 modules are mounted in a single CPU system.</li> <li>• Two or more QI60/A1SI61 modules are set to the same control CPU in a multiple CPU system.</li> <li>• Two or more A1SI61 modules are loaded in a multiple CPU system.</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Reduce the number of QI60/A1SI61 modules mounted in the single CPU system to one.</li> <li>• Change the number of QI60/A1SI61 modules set to the same control CPU to only one in the multiple CPU system.</li> <li>• Reduce the number of A1SI61 modules to only one in the multiple CPU system. When using an interrupt module with each QCPU in a multiple CPU system, replace it with the QI60. (Use one A1SI61 module + max. three QI60 modules or only the QI60 modules.)</li> </ul>	RUN: Off	Qn(H) <sup>*3</sup> QnPH
	<p><b>[SP. UNIT LAY ERR.]</b> Two or more QI60, A1SI61 interrupt modules have been mounted.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Install only 1 QI60, A1SI61 module.	ERR.: Flicker	Qn(H) QnPRH
	<p><b>[SP. UNIT LAY ERR.]</b> Two or more QI60 modules are mounted.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Reduce the QI60 modules to one.	CPU Status: Stop	Q00J/Q00/Q01 <sup>*5</sup>
	<p><b>[SP. UNIT LAY ERR.]</b> Two or more QI60 modules where interrupt pointer setting has not been made are mounted.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Reduce the QI60 modules to one.</li> <li>• Make interrupt pointer setting to the second QI60 module and later.</li> </ul>		Q00J/Q00/Q01 <sup>*3</sup> QnU

\*3 The function version is B or later.

\*5 The module whose first 5 digits of serial No. is "04101" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2106	<p><b>[SP.UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Two or more MELSECNET/H modules are mounted.</li> <li>Two or more CC-Link IE controller network modules are mounted.</li> <li>Two or more Ethernet modules are mounted.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the number of MELSECNET/H modules to one.</li> <li>Reduce the number of CC-Link IE controller network modules to one.</li> <li>Reduce the number of Ethernet modules to one.</li> </ul>		Q00UJ
	<p><b>[SP.UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Five or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system.</li> <li>Two or more MELSECNET/H modules are mounted in the entire system.</li> <li>Two or more CC-Link IE controller network modules are mounted in the entire system.</li> <li>Two or more Ethernet modules are mounted in the entire system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the number of MELSECNET/H and CC-Link IE controller network modules to four or less in total in the entire system.</li> <li>Reduce the number of MELSECNET/H modules to one in the entire system.</li> <li>Reduce the number of CC-Link IE controller network modules to one in the entire system.</li> <li>Reduce the number of Ethernet modules to one in the entire system.</li> </ul>		Q00U/Q01U
	<p><b>[SP.UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Three or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system.</li> <li>Three or more Ethernet interface modules are mounted in the entire system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the MELSECNET/H and CC-Link IE controller network modules up to two or less in the entire system.</li> <li>Reduce the Ethernet interface modules up to two or less in the entire system.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q02U
	<p><b>[SP.UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Five or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system.</li> <li>Five or more Ethernet interface modules are mounted in the entire system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the MELSECNET/H and CC-Link IE controller network modules up to four or less in the entire system.</li> <li>Reduce the Ethernet interface modules up to four or less in the entire system.</li> </ul>		QnU*7
	<p><b>[SP.UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Three or more CC-Link IE controller network modules are mounted in the entire system.</li> <li>Five or more MELSECNET/H and CC-Link IE controller network modules in total are mounted in the entire system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the CC-Link IE controller network modules up to two or less in the entire system.</li> <li>Reduce the total number of the MELSECNET/H and CC-Link IE controller network modules up to four or less in the entire system.</li> </ul>		Qn(H)*6 QnPH*9 QnPRH*9

\*6 The module whose first 5 digits of serial No. is "09012" or later.

\*7 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*9 The module whose first 5 digits of serial No. is "10042" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2106	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Five or more MELSECNET/H modules have been installed.</li> <li>Five or more Ethernet interface modules have been installed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the number of MELSECNET/H modules to four or less.</li> <li>Reduce the number of Ethernet modules to four or less.</li> </ul>		Qn(H) QnPH QnPRH
	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Two or more MELSECNET/H modules were installed.</li> <li>Two or more Ethernet modules were installed.</li> <li>Three or more CC-Link modules were installed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reduce the MELSECNET/H modules to one or less.</li> <li>Reduce the Ethernet modules to one or less.</li> <li>Reduce the CC-Link modules to two or less.</li> </ul>		Q00J/Q00/Q01
	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>The same network number or same station number is duplicated in the MELSECNET/H network system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Check the network number and station number.</li> </ul>		RUN: Off ERR.: Flicker  CPU Status: Stop
2107	<p><b>[SP. UNIT LAY ERR.]</b></p> <p>The start X/Y set in the PLC parameter's I/O assignment settings is overlapped with the one for another module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<p>Make the PLC parameter's I/O assignment setting again so it is consistent with the actual status of the intelligent function module/special function modules.</p>		QCPU
2108	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>Network module A1SJ71LP21, A1SJ71BR11, A1SJ71AP21, A1SJ71AR21, or A1SJ71AT21B dedicated for the A2USCPU has been installed.</li> <li>Network module A1SJ71QLP21 or A1SJ71QBR11 dedicated for the Q2AS has been installed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<p>Replace the network module for the A2USCPU or the network module for the Q2ASCPU with the MELSECNET/H module.</p>		Qn(H)

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2110	<p><b>[SP. UNIT ERROR]</b></p> <ul style="list-style-type: none"> <li>The location designated by the FROM/TO instruction set is not the intelligent function module/special function module.</li> <li>The module that does not include buffer memory has been specified by the FROM/TO instruction.</li> <li>The intelligent function module/special function module, Network module being accessed is faulty.</li> <li>Station not loaded was specified using the instruction whose target was the CPU share memory.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>Read the individual information of the error using the GX Developer, check the FROM/TO instruction that corresponds to that numerical value (program error location), and correct when necessary.</li> <li>The intelligent function module/special function module that was accessed is experiencing a hardware fault. Therefore, change the faulty module. Alternatively, contact your local Mitsubishi representative.</li> </ul>		<p>Q00J/Q00/Q01</p> <p>Qn(H)<sup>*3</sup></p> <p>QnPH</p> <p>QnPRH</p> <p>QnU</p>
2111	<p><b>[SP. UNIT ERROR]</b></p> <ul style="list-style-type: none"> <li>The location designated by a link direct device (J□□□) is not a network module.</li> <li>The I/O module (intelligent function module/special function module) was nearly removed, completely removed, or mounted during running.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>		<p>RUN: Off/On</p> <p>ERR.: Flicker/On</p> <p>CPU Status: Stop/ Continue<sup>*1</sup></p>	
2112	<p><b>[SP. UNIT ERROR]</b></p> <ul style="list-style-type: none"> <li>The module other than intelligent function module/special function module is specified by the intelligent function module/special function module dedicated instruction.</li> <li>Or, it is not the corresponding intelligent function module/special function module.</li> <li>There is no network No. specified by the network dedicated instruction.</li> <li>Or the relay target network does not exist.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed/STOP → RUN</li> </ul>	<p>Read the individual information of the error using a peripheral device, and check the special function module /special function module dedicated instruction (network instruction) that corresponds to the value (program error part) to make modification.</p>		<p>QCPU</p>
2113	<p><b>[SP. UNIT ERROR]</b></p> <p>The module other than network module is specified by the network dedicated instruction.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:FFFF<sub>H</sub> (fixed)</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed/STOP → RUN</li> </ul>			<p>Qn(H)</p> <p>QnPH</p>

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*3 The function version is B or later.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2114	<p><b>[SP. UNIT ERROR]</b> An instruction, which on execution specifies other stations, has been used for specifying the host CPU. (An instruction that does not allow the host CPU to be specified).</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed/ STOP → RUN</li> </ul>	Read the individual information of the error using the GX Developer, check the program corresponding that value (program error location), and make correction.	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/Continue	Q00J/Q00/Q01* <sup>3</sup> Qn(H)* <sup>3</sup> QnPH QnU
2115	<p><b>[SP. UNIT ERROR]</b> An instruction, which on execution specifies the host CPU, has been used for specifying other CPUs. (An instruction that does not allow other stations to be specified).</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed/ STOP → RUN</li> </ul>			Q00J/Q00/Q01* <sup>3</sup> Qn(H)* <sup>3</sup> QnPH
2116	<p><b>[SP. UNIT ERROR]</b> • An instruction that does not allow the under the control of another CPU to be specified is being used for a similar task.</p> <p>• Instruction was executed for the A or QnA module under control of another CPU.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed/ STOP → RUN</li> </ul>			Q00J/Q00/ Q01* <sup>3</sup> Qn(H)* <sup>3</sup> QnPH QnU
2117	<p><b>[SP. UNIT ERROR]</b> A CPU module that cannot be specified in the instruction dedicated to the multiple CPU system was specified.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed/ STOP → RUN</li> </ul>			Qn(H)* <sup>3</sup> QnPH QnU* <sup>7</sup>
2118	<p><b>[SP. UNIT ERROR]</b> When the online module change setting is set to be "enabled" in the PLC parameter in a multiple CPU system, intelligent function module controlled by other CPU using the FROM instruction/intelligent function module device (U□\□G) is specified.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			<ul style="list-style-type: none"> <li>• When performing the online module change in a multiple CPU system, correct the program so that access will not be made to the intelligent function module controlled by the other CPU.</li> <li>• When accessing the intelligent function module controlled by the other CPU in a multiple CPU system, set the online module change setting to be "disabled" by parameter.</li> </ul>

\*3 The function version is B or later.

\*7 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2120	<p><b>[SP. UNIT LAY ERR.]</b> The locations of the Q5□B/Q6□B, QA1S6□B/QA6□B, and QA6ADP+A5□B/A6□B are improper.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Check the location of the base unit.		Q00J/Q00/Q01*4 Qn(H) QnPH
2121	<p><b>[SP. UNIT LAY ERR.]</b> The CPU module is installed to other than the CPU slot and slots 0 to 2.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Check the loading position of the CPU module and reinstall it at the correct slot.		Qn(H) QnPH
2122	<p><b>[SP. UNIT LAY ERR.]</b> The QA1S6□B/QA6□B and QA6ADP+A5□B/A6□B are used for the main base unit.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Replace the main base unit with a usable one.		Qn(H) QnPH QnPRH
2124	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>• A module is mounted on the 65th slot or later slot.</li> <li>• A module is mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting.</li> <li>• A module is mounted on the slot whose number of I/O points exceeds 4096 points.</li> <li>• A module is mounted on the slot whose number of I/O points strides 4096 points.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Remove the module mounted on the 65th slot or later slot.</li> <li>• Remove the module mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting.</li> <li>• Remove the module mounted on the slot whose number of I/O points exceeds 4096 points.</li> <li>• Replace the module with the one whose number of occupied points does not exceed 4096 points.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) QnPH QnPRH QnU*7
	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>• A module is mounted on after the 25th slot (on after the 17th slot for the Q00UJ).</li> <li>• A module is mounted on the slot whose number is later than the one set in the "Base setting" on the I/O assignment tab of PLC parameter in GX Developer.</li> <li>• A module is mounted on the slot for which I/O points greater than 1024 (greater than 256 for the Q00UJ) is assigned.</li> <li>• A module is mounted on the slot for which I/O points is assigned from less than 1024 to greater than 1024 (from less than 256 to greater than 256 for the Q00UJ).</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Remove the module mounted on after the 25th (on after the 17th slot for the Q00UJ).</li> <li>• Remove the module mounted on the slot whose number is later than the one set in the "Base setting" on the I/O assignment tab of PLC parameter in GX Developer.</li> <li>• Remove the module mounted on the slot for which I/O points greater than 1024 (greater than 256 for the Q00UJ) is assigned.</li> <li>• Replace the end module with the one whose number of occupied points is within 1024 (within 256 for the Q00UJ).</li> </ul>		Q00UJ Q00U/Q01U

\*4 The function version is A.

\*7 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2124	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>A module is mounted on the 37th slot or later slot.</li> <li>A module is mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting.</li> <li>A module is mounted on the slot whose number of I/O points exceeds 2048 points.</li> <li>A module is mounted on the slot whose number of I/O points strides 2048 points.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Remove the module mounted on the 37th slot or later slot.</li> <li>Remove the module mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting.</li> <li>Remove the module mounted on the slot whose number of I/O points exceeds 2048 points.</li> <li>Replace the module with the one whose number of occupied points does not exceed 2048 points.</li> </ul>		Q02U
	<p><b>[SP. UNIT LAY ERR.]</b></p> <ul style="list-style-type: none"> <li>A module is mounted on the 25th slot or later slot. (The 17th slot or later slot for the Q00J.)</li> <li>A module is mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting.</li> <li>A module is mounted on the slot whose number of I/O points exceeds 1024 points. (256 points for the Q00J.)</li> <li>A module is mounted on the slot whose number of I/O points strides 1024 points. (256 points for the Q00J.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Remove the module mounted on the 25th slot or later slot. (The 17th slot or later slot for the Q00J.)</li> <li>Remove the module mounted on the slot whose number is greater than the number of slots specified at [Slots] in [Standard setting] of the base setting.</li> <li>Remove the module mounted on the slot whose number of I/O points exceeds 1024 points. (256 points for the Q00J.)</li> <li>Replace the module with the one whose number of occupied points does not exceed 1024 points. (256 points for the Q00J.)</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00J/Q00/Q01
	<p><b>[SP. UNIT LAY ERR.]</b></p> <p>5 or more extension base units were added. (3 bases for Q00J)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	Remove 5 or more extension base units. (3 bases for Q00J)		Q00J/Q00/Q01*4
2125	<p><b>[SP. UNIT LAY. ERR.]</b></p> <ul style="list-style-type: none"> <li>A module which the QCPU cannot recognise has been installed.</li> <li>There was no response from the intelligent function module/special function module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No. (Slot No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>Install a usable module.</li> <li>The intelligent function module/special function module is experiencing a hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU

\*4 The function version is A.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2126	<p><b>[SP. UNIT LAY. ERR.]</b> CPU module locations in a multiple CPU system are either of the following.</p> <ul style="list-style-type: none"> <li>• There are empty slots between the QCPU and QCPU/motion controller.</li> <li>• A module other than the High Performance model QCPU/Process CPU (including the motion controller) is mounted on the left-hand side of the High Performance model QCPU/Process CPU.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Mount modules on the available slots so that the empty slots will be located on the right-hand side of the CPU module.</li> <li>• Remove the module mounted on the left-hand side of the High Performance model QCPU/Process CPU, and mount the High Performance model QCPU/Process CPU on the empty slot. Mount the motion CPU on the right-hand side of the High Performance model QCPU/Process CPU.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*3</sup> QnPH
2128	<p><b>[SP.UNIT LAY ERR.]</b> The unusable module is mounted on the extension base unit in the redundant system.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Remove the unusable module from the extension base unit.</li> </ul>		QnPRH <sup>*6</sup>
2150	<p><b>[SP. UNIT VER. ERR.]</b> In a multiple CPU system, the control CPU of the intelligent function module incompatible with the multiple CPU system is set to other than CPU No.1.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Change the intelligent function module for the one compatible with the multiple CPU system (function version B).</li> <li>• Change the setting of the control CPU of the intelligent function module incompatible with the multiple CPU system to CPU No.1.</li> </ul>		Q00J/Q00/Q01 QnPH QuU <sup>*10</sup>
2151	<p><b>[SP. UNIT VER. ERR.]</b> Either of the following modules incompatible with the redundant system has been mounted in a redundant system.</p> <ul style="list-style-type: none"> <li>• CC-Link IE controller network modules</li> <li>• MELSECNET/H modules</li> <li>• Ethernet modules</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No. (Slot No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ At writing to progurammable controller</li> </ul>	Use either of the following modules compatible with the redundant system. <ul style="list-style-type: none"> <li>• CC-Link IE controller network modules</li> <li>• MELSECNET/H modules</li> <li>• Ethernet modules</li> </ul>		QnPRH

\*3 The function version is B or later.

\*6 The module whose first 5 digits of serial No. is "09012" or later.

\*10 The Universal model QCPU except the Q00JCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2200	<p><b>[MISSING PARA.]</b> There is no parameter file in the drive specified as valid parameter drive by the DIP switches.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information: Drive Name</li> <li>• Individual Information: –</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Check and correct the valid parameter drive settings made by the DIP switches.</li> <li>• Set the parameter file to the drive specified as valid parameter drive by the DIP switches.</li> </ul>	<p>RUN: Off ERR.: Flicker  CPU Status: Stop</p>	Qn(H) QnPH QnPRH
	<p><b>[MISSING PARA.]</b> There is no parameter file at the program memory.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information: Drive Name</li> <li>• Individual Information: –</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	Set the parameter file to the program memory.		Q00J/Q00/Q01
	<p><b>[MISSING PARA.]</b> Parameter file does not exist in all drives where parameters will be valid.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information: Drive Name</li> <li>• Individual Information: –</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	Set a parameter file in a drive to be valid.		QuU
2210	<p><b>[BOOT ERROR]</b> The contents of the boot file are incorrect.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information: Drive name</li> <li>• Individual Information: –</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	Check the boot setting.	<p>RUN: Off ERR.: Flicker  CPU Status: Stop</p>	Q00J/Q00/Q01* <sup>3</sup> Qn(H) QnPH QnPRH QnU
2211	<p><b>[BOOT ERROR]</b> File formatting is failed at a boot.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information: Drive name</li> <li>• Individual Information: –</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Reboot.</li> <li>• CPU module hardware fault. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		Qn(H) QnPRH QnU
2220	<p><b>[RESTORE ERROR]</b></p> <ul style="list-style-type: none"> <li>• The device information (number of points) backed up by the device data backup function is different from the number of device points of the PLC parameter.</li> </ul> <p>After this error occurred, perform restore per power-on/reset until the number of device points is identical to the number of device points in the PLC parameter, or until the backup data is deleted.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information: File name/ Drive name</li> <li>• Individual Information: –</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Set the number of device points at the time of backup to the device point setting in [PLC parameter]. Then, turn ON from OFF power supply, or reset the CPU and cancel reset.</li> <li>• Delete the backed up data, and turn ON from OFF power supply, or reset the CPU and cancel reset.</li> </ul>		QnU

12.1 Error Code List  
12.1.4 Error code list (2000 to 2999)

\*3 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2221	<p><b>[RESTORE ERROR]</b></p> <ul style="list-style-type: none"> <li>The device information backed up by the device data backup function is incomplete. (Turning power supply OFF or reset is suspected.)</li> </ul> <p>Do not return the data when this error occurs. Also, delete the incomplete device information at the time of this error occurrence.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	Reset the CPU module and run it again.		
2225	<p><b>[RESTORE ERROR]</b></p> <p>The model name of the restoration destination CPU module is different from the one of the backup source CPU module.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	Execute a restore for the CPU module whose name is same as the backup source CPU module.	RUN: Off ERR.: Flicker	QnU
2226	<p><b>[RESTORE ERROR]</b></p> <ul style="list-style-type: none"> <li>The backup data file is destroyed. (The content of the file is different from the check code.</li> <li>Reading the backup data from the memory card is not successfully completed.</li> <li>Since the write protect switch of the SRAM card is set to on (write inhibited), the checked "Restore for the first time only" setting cannot be performed.</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	<ul style="list-style-type: none"> <li>Execute a restore of other backup data because the backup data may be destructed.</li> <li>Set the write protect switch of the SRAM card to off (write enabled).</li> </ul>	CPU Status: Stop	
2227	<p><b>[RESTORE ERROR]</b></p> <p>Writing the backup data to the restoration destination drive is not successfully completed.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>Common Information:File name/Drive name</li> <li>Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>At power ON/ At reset</li> </ul>	Execute a restore for the other CPU module too because the CPU module may be damaged.		
2300	<p><b>[ICM. OPE. ERROR]</b></p> <ul style="list-style-type: none"> <li>A memory card was removed without switching the memory card in/out switch OFF.</li> <li>The memory card in/out switch is turned ON although a memory card is not actually installed.</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>Common Information:Drive name</li> <li>Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>When memory card is inserted or removed</li> </ul>	<ul style="list-style-type: none"> <li>Remove memory card after placing the memory card in/out switch OFF.</li> <li>Turn on the card insert switch after inserting a memory card.</li> </ul>	RUN: Off/On ERR.: Flicker/On CPU Status: Stop/ Continue*1	Qn(H) QnPH QnPRH QnU*11

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*11 The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2301	<p><b>[ICM. OPE. ERROR]</b></p> <ul style="list-style-type: none"> <li>The memory card has not been formatted.</li> <li>Memory card format status is incorrect.</li> <li>The QCPU file does not exist in the Flash card.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Drive name</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When memory card is inserted or removed/When memory card is inserted</li> </ul>	<ul style="list-style-type: none"> <li>Format memory card.</li> <li>Reformat memory card.</li> <li>Write the QCPU file the Flash card</li> </ul>		Qn(H) QnPH QnPRH QnU* <sup>11</sup>
	<p><b>[ICM. OPE. ERROR]</b></p> <p>SRAM card failure is detected. (It occurs when automatic format is not set.) Writing parameters was performed during setting file registers.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Drive name</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When memory card is inserted or removed/When memory card is inserted</li> </ul>	<p>Format SRAM card after changing battery of SRAM card.</p> <p>Write a parameter, which set the file register at "Not available", in CPU, and then perform the operation.</p>	<p>RUN: Off/On ERR.: Flicker/On</p> <p>CPU Status: Stop/ Continue*<sup>1</sup></p>	QnU* <sup>11</sup>
2302	<p><b>[ICM. OPE. ERROR]</b></p> <p>A memory card that cannot be used with the CPU module has been installed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Drive name</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When memory card is inserted or removed</li> </ul>	<ul style="list-style-type: none"> <li>Format memory card.</li> <li>Reformat memory card.</li> <li>Check memory card.</li> </ul>		Qn(H) QnPH QnPRH QnU* <sup>11</sup>
2400	<p><b>[FILE SET ERROR]</b></p> <p>Automatic write to standard ROM was performed on the CPU module that is incompatible with automatic write to standard ROM. (Memory card where automatic write to standard ROM was selected in the boot file was fitted and the parameter enable drive was set to the memory card.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/ At writing to programmable controller</li> </ul>	<ul style="list-style-type: none"> <li>Execute automatic write to standard ROM on the CPU module which is compatible with automatic write to standard ROM.</li> <li>Using GX Developer, perform write of parameters and programs to standard ROM.</li> <li>Change the memory card for the one where automatic write to standard ROM has not been set, and perform boot operation from the memory card.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	Qn(H) <sup>3</sup> QnPH QnPRH
	<p><b>[FILE SET ERROR]</b></p> <p>The file designated at the PLC file settings in the parameters cannot be found.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/ At writing to programmable controller</li> </ul>	<ul style="list-style-type: none"> <li>Read the individual information of the error using peripheral device, check to be sure that the parameter drive name and file name correspond to the numerical values there (parameter number), and correct.</li> <li>Create a file created using parameters, and load it to the CPU module.</li> </ul>		QCPU

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*3 The function version is B or later.

\*11 The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2401	<p><b>[FILE SET ERROR]</b> Program memory capacity was exceeded by performing boot operation or automatic write to standard ROM.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ At writing to programmable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Check and correct the parameters (boot setting).</li> <li>• Delete unnecessary files in the program memory.</li> <li>• Choose "Clear program memory" for boot in the parameter so that boot is started after the program memory is cleared.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*3</sup> QnPH QnPRH
	<p><b>[FILE SET ERROR]</b> Program memory capacity was exceeded by performing boot operation.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ At writing to programmable controller</li> </ul>			QnU
	<p><b>[FILE SET ERROR]</b> The file specified by parameters cannot be made.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ At writing to programmable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check to be sure that the parameter drive name and file name correspond to the numerical values there (parameter number), and correct.</li> <li>• Check the space remaining in the memory card.</li> </ul>		QCPU
	<p><b>[FILE SET ERROR]</b></p> <ul style="list-style-type: none"> <li>• Although setting is made to use the device data storage file, there is no empty capacity required for creating the device data storage file in the standard ROM.</li> <li>• When the latch data backup function (to standard ROM) is used, there is no empty capacity required for storing backup data in standard ROM. (The parameter number "FFFF<sub>H</sub>" is displayed for the error individual information.)</li> <li>• Standard RAM capacity is insufficient that error history of the module cannot be stored.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ At writing to programmable controller</li> </ul>	Secure the empty capacity of the standard ROM.		QnU
2410	<p><b>[FILE OPE. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The specified program does not exist in the program memory. This error may occur when the ECALL, EFCALL, PSTOP, PSCAN, POFF or PLOW instruction is executed.</li> <li>• The specified file does not exist.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check to be sure that the program corresponds to the numerical values there (program location), and correct. Create a file created using parameters, and load it to the CPU module.</li> <li>• In case a specified file does not exist, write the file to a target memory and/or check the file specified with the instruction again.</li> </ul>	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue <sup>*1</sup>	Qn(H) QnPH QnPRH QnU

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*3 The function version is B or later.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2411	<p><b>[FILE OPE. ERROR]</b></p> <ul style="list-style-type: none"> <li>The file is the one which cannot be specified by the sequence program (such as comment file).</li> <li>The specified program exists in the program memory, but has not been registered in the program setting of the Parameter dialog box. This error may occur when the ECALL, EFCALL, PSTOP, PSCAN or POFF instruction is executed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the individual information of the error using the peripheral device, check to be sure that the program corresponds to the numerical values there (program location), and correct.	RUN: Off/On ERR.: Flicker/On	Qn(H) QnPH QnPRH QnU
2412	<p><b>[FILE OPE. ERROR]</b></p> <p>The SFC program file is one that cannot be designated by the sequence program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the individual information of the error using the peripheral device, check to be sure that the program corresponds to the numerical values there (program location), and correct.	CPU Status: Stop/ Continue*1	Qn(H) QnPH QnPRH QnU
2413	<p><b>[FILE OPE. ERROR]</b></p> <p>No data has been written to the file designated by the sequence program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Program error location</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the individual information of the error using the peripheral device, check to be sure that the program corresponds to the numerical values there (program location), and correct. Check to ensure that the designated file has not been write protected.		Qn(H) QnPH QnPRH
2500	<p><b>[CAN'T EXE. PRG.]</b></p> <ul style="list-style-type: none"> <li>There is a program file that uses a device that is out of the range set in the PLC parameter device setting.</li> <li>After the PLC parameter setting is changed, only the parameter is written into the PLC.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Read the common information of the error using the peripheral device, check to be sure that the parameter device allocation setting and the program file device allocation correspond to the numerical values there (file name), and correct if necessary.</li> <li>If PLC parameter device setting is changed, batch-write the parameter and program file into the PLC.</li> </ul>	RUN: Off ERR.: Flicker	QCPU
	<p><b>[CAN'T EXE. PRG.]</b></p> <p>After the index modification of the PLC parameter is changed, only the parameter is written to the PLC.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/ STOP → RUN</li> </ul>	When the index modification of the PLC parameter is changed, batch-write the parameter and program file into the PLC.	CPU Status: Stop	QnU

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2501	<p><b>[CAN'T EXE. PRG.]</b> There are multiple program files although "none" has been set at the PLC parameter program settings.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<p>Edit the PLC parameter program setting to "yes". Alternatively, delete unneeded programs.</p>		<p>Qn(H) QnPH QnPRH QnU</p>
	<p><b>[CAN'T EXE. PRG.]</b></p> <ul style="list-style-type: none"> <li>• There are three or more program files.</li> <li>• The program name differs from the program contents.</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Delete unnecessary program files.</li> <li>• Match the program name with the program contents.</li> </ul>		<p>Q00J/Q00/Q01</p>
2502	<p><b>[CAN'T EXE. PRG.]</b> The program file is incorrect. Alternatively, the file contents are not those of a sequence program.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<p>Check whether the program version is * * * .QPG, and check the file contents to be sure they are for a sequence program.</p>	<p>RUN: Off ERR.: Flicker CPU Status: Stop</p>	<p>QCPU</p>
	<p><b>[CAN'T EXE. PRG.]</b> The program file is not the one for the redundant CPU.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<p>Create a program using GX Developer or PX Developer for which the PLC type has been set to the redundant CPU (Q12PRH/Q25PRH), and write it to the CPU module.</p>		<p>QnPRH</p>
2503	<p><b>[CAN'T EXE. PRG.]</b> There are no program files at all.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>			<p>QCPU</p>
2504	<p><b>[CAN'T EXE. PRG.]</b> Two or more SFC normal programs or control programs have been designated.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Check program configuration.</li> <li>• Check parameters and program configuration.</li> </ul>		<p>Qn(H) QnPH QnPRH QnU</p>
	<p><b>[CAN'T EXE. PRG.]</b> There are two or more SFC programs.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:--</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset/ STOP → RUN</li> </ul>	<p>Reduce the SFC programs to one.</p>		<p>Q00J/Q00/Q01*<sup>3</sup></p>

\*3 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
2700	<p><b>[REMOTE PASS.FAIL]</b> The count of remote password mismatches reached the upper limit.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<p>Check for illegal accesses. If any illegal access is identified, take actions such as disabling communication of the connection.</p> <p>If no illegal access is identified, clear the error and perform the following. (Clearing the error also clears the count of remote password mismatches.)</p> <ul style="list-style-type: none"> <li>• Check if the remote password sent is correct.</li> <li>• Check if the remote password has been locked.</li> <li>• Check if concurrent access was made from multiple devices to one connection by UDP.</li> <li>• Check if the upper limit of the remote password mismatch count is too low.</li> </ul>	<p>RUN: ON ERR.: ON</p> <p>CPU Status: Continue</p>	QnU <sup>*8</sup>
2710	<p><b>[SNTP OPE.ERROR]</b> Time setting failed when the programmable controller was powered ON or reset.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When time setting function is executed</li> </ul>	<ul style="list-style-type: none"> <li>• Check if the time setting function is set up correctly.</li> <li>• Check if the specified SNTP server is operating normally, or if any failure has occurred on the network connected to the specified SNTP server computer.</li> </ul>	<p>RUN: Off/ON ERR.: Flicker/ON</p> <p>CPU Status: Stop/Continue</p>	

\*8 This applies to the Built-in Ethernet port QCPU.

## 12.1.5 Error code list (3000 to 3999)

The following shows the error messages from the error code 3000 to 3999, the contents and causes of the errors, and the corrective actions for the errors.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3000	<p><b>[PARAMETER ERROR]</b> In a multiple CPU system, the intelligent function module under control of another CPU is specified in the interrupt pointer setting of the PLC parameter.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Specify the head I/O number of the intelligent function module under control of the host CPU.</li> <li>• Delete the interrupt pointer setting of the parameter.</li> </ul>		Qn(H) <sup>*1</sup> QnPH QnU <sup>*10</sup>
	<p><b>[PARAMETER ERROR]</b> The PLC parameter settings for timer time limit setting, the RUN-PAUSE contact, the common pointer number, general data processing, number of empty slots, system interrupt settings, baud rate setting, and service processing setting are outside the range that can be used by the CPU module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>			QCPU
	<p><b>[PARAMETER ERROR]</b> In a program memory check, the check capacity has not been set within the range applicable for the CPU module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check the parameter item corresponding to the numerical value (parameter No.), and correct it.</li> <li>• Rewrite corrected parameters to the CPU module, reload the CPU power supply and/or reset the module.</li> </ul>	RUN: Off ERR.: Flicker CPU Status: Stop	QnPH QnPRH <sup>*5</sup>
	<p><b>[PARAMETER ERROR]</b> The parameter settings in the error individual information (special register SD16) are illegal.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• If the same error occurs, it is thought to be a hardware error. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU
	<p><b>[PARAMETER ERROR]</b> The ATA card is set to the memory card slot when the specified drive for the file register is set to "memory card (ROM)" and [Use the following file] or [Use the same file name as the program] (either one is allowed) is set in the PLC file setting.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>			QnU <sup>*11</sup>

\*1 The function version is B or later.

\*5 The module whose first 5 digits of serial No. is "07032" or later.

\*10 The Universal model QCPU except the Q00UJCPU.

\*11 The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3001	<p><b>[PARAMETER ERROR]</b> The parameter settings are corrupted.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check the parameter item corresponding to the numerical value (parameter No.), and correct it.</li> <li>• Rewrite corrected parameters to the CPU module, reload the CPU power supply and/or reset the module.</li> <li>• If the same error occurs, it is thought to be a hardware error. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU
3002	<p><b>[PARAMETER ERROR]</b> When "Use the following file" is selected for the file register in the PLC file setting of the PLC parameter dialog box, the specified file does not exist although the file register capacity has been set.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check the parameter item corresponding to the numerical value (parameter No.), and correct it.</li> <li>• Rewrite corrected parameters to the CPU module, reload the CPU power supply and/or reset the module.</li> <li>• If the same error occurs, it is thought to be a hardware error. (Contact your local Mitsubishi representative.)</li> </ul>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	Qn(H) QnPH QnPRH
	<p><b>[PARAMETER ERROR]</b> When [Use the following file] is set for the file register in the PLC file setting of the PLC parameter dialog box and the capacity of file register is not set, the file register file does not exist in the specified target memory.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>			QnU*10
	<p><b>[PARAMETER ERROR]</b> When [Use the following file.] is set for the device data storage file in [PLC file] of [PLC parameter], and [Capacity] is not set, the device data storage file does not exist in the target memory.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN/ At writing to progurammable controller</li> </ul>			QnU

\*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3003	<p><b>[PARAMETER ERROR]</b> The automatic refresh range of the multiple CPU system exceeded the file register capacity.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Change the file register file for the one refresh-enabled in the whole range.		Qn(H) <sup>*1</sup> QnPH QnU <sup>*10</sup>
	<p><b>[PARAMETER ERROR]</b> The number of devices set at the PLC parameter device settings exceeds the possible CPU module range.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to programmable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check the parameter item corresponding to the numerical value (parameter No.), and correct it.</li> <li>• If the error is still generated following the correction of the parameter settings, the possible cause is the memory error of the CPU module's program memory or the memory card. (Contact your local Mitsubishi representative.)</li> </ul>		
3004	<p><b>[PARAMETER ERROR]</b> The parameter file is incorrect. Alternatively, the contents of the file are not parameters.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to programmable controller</li> </ul>	Check whether the parameter file version is * * *.QPA, and check the file contents to be sure they are parameters.	RUN: Off ERR.: Flicker	QCPU
3005	<p><b>[PARAMETER ERROR]</b> The contents of the parameter are broken.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using the peripheral device, check the parameter item corresponding to the numerical value (parameter No.), and correct it.</li> <li>• Write the modified parameter items to the CPU module again, and power-on the Programmable Controller or reset the CPU module.</li> <li>• When the same error occurs again, the hardware is faulty. Contact your local Mitsubishi representative, explaining a detailed description of the problem.</li> </ul>	CPU Status: Stop	
3006	<p><b>[PARAMETER ERROR]</b></p> <ul style="list-style-type: none"> <li>• The high speed interrupt is set in a Q02CPU.</li> <li>• The high speed interrupt is set in a multiple CPU system.</li> <li>• The high speed interrupt is set when aQA1S6□B or QA6□B is used.</li> <li>• No module is installed at the I/O address designated by the high speed interrupt.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to programmable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Delete the setting of the Q02CPU' s high speed interrupt. To use high speed interrupts, change the CPU module to one of the Q02H/Q06H/ Q12H/Q25HCPU.</li> <li>• To use a multiple CPU system, delete the setting of the high-speed interrupt. To use high speed interrupts, change the system to a single CPU system.</li> <li>• To use either the QA1S6□B or QA6□B, delete the setting of the high speed interrupt. To use high speed interrupts, do not use the QA1S6□B/ QA6□B.</li> <li>• Re-examine the I/O address designated by the high speed interrupt setting.</li> </ul>		Qn(H) <sup>*4</sup>

\*1 The function version is B or later.

\*4 The module whose first 5 digits of serial No. is "04012" or later.

\*7 The module whose first 5 digits of serial No. is "09012" or later.

\*9 The module whose first 5 digits of serial No. is "10042" or later.

\*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3007	<p><b>[PARAMETER ERROR]</b> The parameter file in the drive specified as valid parameter drive by the DIP switches is inapplicable for the CPU module.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Create parameters using GX Developer, and write them to the drive specified as valid parameter drive by the DIP switches.		QnPRH
3009	<p><b>[PARAMETER ERROR]</b> In a multiple CPU system, the modules for AnS, A, Q2AS and QnA have been set to multiple control CPUs.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Re-set the parameter I/O assignment to control them under one CPU module. (Change the parameters of all CPUs in the multiple CPU system.)	RUN: Off ERR.: Flicker	Qn(H) <sup>*1</sup>
3010	<p><b>[PARAMETER ERROR]</b> The parameter-set number of CPU modules differs from the actual number in a multiple CPU system.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Match the number of (CPU modules in multiple CPU setting) - (CPUs set as empty in I/O assignment) with that of actually mounted CPU modules.	CPU Status: Stop	Qn(H) <sup>*1</sup> QnPH
3012	<p><b>[PARAMETER ERROR]</b> Multiple CPU setting or control CPU setting differs from that of the reference CPU settings in a multiple CPU system.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Match the multiple CPU setting or control CPU setting in the PLC parameter with that of the reference CPU (CPU No.1) settings.		Q00/Q01 <sup>*1</sup> Qn(H) <sup>*1</sup> QnU

\*1 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3013	<p><b>[PARAMETER ERROR]</b> Multiple CPU auto refresh setting is any of the followings in a multiple CPU system.</p> <ul style="list-style-type: none"> <li>When a bit device is specified as a refresh device, a number other than a multiple of 16 is specified for the refresh-starting device.</li> <li>The device specified is other than the one that may be specified.</li> <li>The number of send points is an odd number.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	<p>Check the following in the multiple CPU auto refresh setting and make correction.</p> <ul style="list-style-type: none"> <li>When specifying the bit device, specify a multiple of 16 for the refresh starting device.</li> <li>Specify the device that may be specified for the refresh device.</li> <li>Set the number of send points to an even number.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	Qn(H) <sup>*1</sup> QnPH
	<p><b>[PARAMETER ERROR]</b> In a multiple CPU system, the multiple CPU auto refresh setting is any of the following.</p> <ul style="list-style-type: none"> <li>The total number of transmission points is greater than the maximum number of refresh points.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	<p>Check the following in the multiple CPU auto refresh setting and make correction.</p> <ul style="list-style-type: none"> <li>The total number of transmission points is within the maximum number of refresh points.</li> </ul>		Q00/Q01 <sup>*1</sup>
	<p><b>[PARAMETER ERROR]</b> In a multiple CPU system, the multiple CPU auto refresh setting is any of the following.</p> <ul style="list-style-type: none"> <li>The device specified is other than the one that may be specified.</li> <li>The number of send points is an odd number.</li> <li>The total number of send points is greater than the maximum number of refresh points.</li> <li>The setting of the refresh range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W).</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	<p>Check the following in the multiple CPU auto refresh setting and make correction.</p> <ul style="list-style-type: none"> <li>Specify the device that may be specified for the refresh device.</li> <li>Set the number of send points to an even number.</li> <li>Set the total number of send points within the range of the maximum number of refresh points.</li> <li>Set the refresh range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).</li> </ul>		QnU <sup>*10</sup>
3014	<p><b>[PARAMETER ERROR]</b></p> <ul style="list-style-type: none"> <li>In a multiple CPU system, the online module change parameter (multiple CPU system parameter) settings differ from those of the reference CPU.</li> <li>In a multiple CPU system, the online module change setting is enabled although the CPU module mounted does not support online module chang parameter.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>Match the online module change parameter with that of the reference CPU.</li> <li>If the CPU module that does not support online module change is mounted, replace it with the CPU module that supports online module change.</li> </ul>	Qn(H) QnPH QnU <sup>*8</sup>	

\*1 The function version is B or later.

\*8 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*10 The Universal model QCPU except the Q00UJCPU.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3015	<p><b>[PARAMETER ERROR]</b> In a multiple CPU system configuration, the CPU verified is different from the one set in the parameter setting.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number/CPU No.</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Read the individual information of the error using the peripheral device, check the parameter item corresponding to the numerical value (parameter No./CPU No.) and parameter of target CPU, and correct them.		
3016	<p><b>[PARAMETER ERROR]</b> The CPU module incompatible with multiple CPU synchronized boot-up is set as the target for the synchronized boot-up in the [Multiple CPU synchronous startup setting].</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number/ CPU No.</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/ At reset/ At writing to progurammable controller</li> </ul>	Delete the CPU module incompatible with multiple CPU synchronized boot-up from the setting.		QnU <sup>*8</sup>
3040	<p><b>[PARAMETER ERROR]</b> The parameter file is damaged.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	With GX Developer, write [PLC parameter/Network parameter/Remote password] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be hardware error. (Contact your local Mitsubishi representative.)	RUN: Off ERR.: Flicker	
3041	<p><b>[PARAMETER ERROR]</b> Parameter file of intelligent function module is damaged.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	With GX Developer, write [Intelligent function module parameter] to a valid drive to write the parameters then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be a hardware error. (Contact your local Mitsubishi representative.)	CPU Status: Stop	
3042	<p><b>[PARAMETER ERROR]</b> The system file that have stored the remote password setting information is damaged.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• With GX Developer, write [PLC parameter/ Network parameter/Remote password] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be a hardware error. (Contact your local Mitsubishi representative.)</li> <li>• When a valid drive for parameter is set to other than [program memory], set the parameter file (PARAM) at the boot file setting to be able to transmit to the program memory.</li> </ul> <p>With GX Developer, write [PLC parameter/ Network parameter/Remote password] to a valid drive then reload the power supply for system and/or reset the CPU module. If the same error occurs, it is thought to be hardware error. (Contact your local Mitsubishi representative.)</p>		Qn(H) <sup>*5</sup> QnPH <sup>*5</sup> QnPRH <sup>*5</sup>

\*5 The module whose first 5 digits of serial No. is "07032" or later.

\*8 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3100	<p><b>[LINK PARA. ERROR]</b> In a multiple CPU system, the CC-Link IE controller network module controlled by another CPU is specified as the head I/O number of the CC-Link IE controller network module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Delete the network parameter of the CC-Link IE controller network module controlled by another CPU.</li> <li>• Change the setting to the head I/O number of the CC-Link IE controller network module controlled by host CPU.</li> </ul>		<p>Qn(H)<sup>*7</sup> QnPH<sup>*9</sup> QnU</p>
	<p><b>[LINK PARA. ERROR]</b> The network parameter of the CC-Link IE controller network operating as the normal station is overwritten to the control station. Or, the network parameter of the CC-Link IE controller network operating as the control station is overwritten to the normal station. (The network parameter is updated on the module by resetting.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ STOP → RUN</li> </ul>	<p>Reset the CPU module.</p>		
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The number of modules actually mounted is different from that is set in Network parameter for MELSECNET/H.</li> <li>• The head I/O number of the actually mounted module is different from the one set in the network parameter of the CC-Link IE controller network.</li> <li>• Data cannot be handled in the parameter exists.</li> <li>• The network type of CC-Link IE controller network is overwritten during power-on. (When changing the network type, switch RESET to RUN.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Check the network parameter and actual mounting status, and if they differ, make them matched. When network parameters are modified, write them to the CPU module.</li> <li>• Check the setting of extension base unit stage number.</li> <li>• Check the connection status of extension base unit and extension cable. When the GOT is bus-connected to the main base unit or extension base unit, also check its connection status.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>Qn(H)<sup>*7</sup> QnPH<sup>*9</sup> QnPRH<sup>*9</sup> QnU</p>
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The CC-Link IE controller network module is specified for the head I/O number of network parameter in the MELSECNET/H.</li> <li>• The MELSECNET/H module is specified for the head I/O number of network parameter in the CC-Link IE controller network.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ STOP → RUN</li> </ul>	<p>If an error occurs even after performing the above checks, the hardware may be faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p>		

\*7 The module whose first 5 digits of serial No. is "09012" or later.

\*9 The module whose first 5 digits of serial No. is "10042" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3100	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>Although the CC-Link IE controller network module is mounted, network parameter for the CC-Link IE controller network module is not set.</li> <li>Although the CC-Link IE controller network and MELSECNET/H modules are mounted, network parameter for the MELSECNET/H module is not set.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power-ON/ At reset/ STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Check the network parameter and actual mounting status, and if they differ, make them matched. When network parameters are modified, write them to the CPU module.</li> <li>Check the setting of extension base unit stage number.</li> <li>Check the connection status of extension base unit and extension cable. When the GOT is bus-connected to the main base unit or extension base unit, also check its connection status. If an error occurs even after performing the above checks, the hardware may be faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		<p>Qn(H)<sup>*7</sup> QnPH<sup>*9</sup> QnPRH<sup>*9</sup> QnU</p>
	<p><b>[LINK PARA. ERROR]</b></p> <p>In a multiple CPU system, the MELSECNET/H under control of another CPU is specified as the head I/O number in the network setting parameter of the MELSECNET/H.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Delete the MELSECNET/H network parameter of the MELSECNET/H under control of another CPU.</li> <li>Change the setting to the head I/O number of the MELSECNET/H under control of the host CPU.</li> </ul>		<p>Q00/Q01<sup>*1</sup> Qn(H)<sup>*1</sup> QnPH QnU<sup>*10</sup></p>
	<p><b>[LINK PARA. ERROR]</b></p> <p>The network parameter of the MELSECNET/H operating as the normal station is overwritten to the control station.</p> <p>Or, the network parameter of the MELSECNET/H operating as the control station is overwritten to the normal station. (The network parameter is updated on the module by resetting.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<p>Reset the CPU module.</p>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>Qn(H)<sup>*1</sup> QnPH QnPRH QnU</p>
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>The number of modules actually mounted is different from that is set in Network parameter for MELSECNET/H.</li> <li>The head I/O number of actually installed modules is different from that designated in the network parameter of MELSECNET/H.</li> <li>Some data in the parameters cannot be handled.</li> <li>The network type of MELSECNET/H is overwritten during power-on. (When changing the network type, switch RESET to RUN.)</li> <li>The mode switch of MELSECNET/H module<sup>*5</sup> is outside the range.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Check the network parameters and actual mounting status, and if they differ, make them matched. If any network parameter has been corrected, write it to the CPU module.</li> <li>Check the extension base unit stage No. setting.</li> <li>Check the connection status of the extension base units and extension cables. When the GOT is bus-connected to the main base unit and extension base units, also check the connection status.</li> </ul> <p>If the error occurs after the above checks, the possible cause is a hardware fault. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p> <ul style="list-style-type: none"> <li>Set the mode switch of MELSECNET/H module<sup>*5</sup> within the range.</li> </ul>		<p>QCPU</p>

\*1 The function version is B or later.  
 \*5 The module whose first 5 digits of serial No. is "07032" or later.  
 \*7 The module whose first 5 digits of serial No. is "09012" or later.  
 \*9 The module whose first 5 digits of serial No. is "10042" or later.  
 \*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3101	<p><b>[LINK PARA. ERROR]</b> The link refresh range exceeded the file register capacity.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Change the file register file for the one that enables entire range refresh.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*1</sup> QnPH QnPRH QnU <sup>*10</sup>
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• When the station number of the MELSECNET/H module is 0, the PLC-to-PLC network parameter has been set.</li> <li>• When the station number of the MELSECNET/H module is other than 0, the remote master parameter setting has been made.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Correct the type or station number of the MELSECNET/H module in the network parameter to meet the used system.		Qn(H) <sup>*1</sup> QnPH QnPRH
	<p><b>[LINK PARA. ERROR]</b> The refresh parameter for the CC-Link IE controller network is outside the range.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Check the network parameters and mounting status, and if they differ, match the network parameters and mounting status.</li> </ul> If any network parameter has been corrected, write it to the CPU module.		Qn(H) <sup>*7</sup> QnPH <sup>*9</sup> QnPRH <sup>*9</sup> QnU
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The network No. specified by a network parameter is different from that of the actually mounted network.</li> <li>• The head I/O No. specified by a network parameter is different from that of the actually mounted I/O unit.</li> <li>• The network class specified by a network parameter is different from that of the actually mounted network.</li> <li>• The network refresh parameter of the MELSECNET/H, MELSECNET/10 is out of the specified area.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Confirm the setting of the number of extension stages of the extension base units.</li> <li>• Check the connection status of the extension base units and extension cables.</li> </ul> When the GOT is bus-connected to the main base unit and extension base units, also check their connection status.  If the error occurs after the above checks, the cause is a hardware fault. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)		QCPU
	<p><b>[LINK PARA. ERROR]</b> A multi-remote I/O network was configured using a module that does not support the MELSECNET/H multi-remote I/O network.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Use a module that supports the MELSECNET/H multi-remote I/O network.		QnPH

\*1 The function version is B or later.

\*7 The module whose first 5 digits of serial No. is "09012" or later.

\*9 The module whose first 5 digits of serial No. is "10042" or later.

\*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU												
3101	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>The system A of the MELSECNET/H remote master station has been set to other than Station No. 0.</li> <li>The system B of the MELSECNET/H remote master station has been set to Station No. 0.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Set the system A of the MELSECNET/H remote master station to Station No. 0.</li> <li>Set the system B of the MELSECNET/H remote master station to any of Station No. 1 to 64.</li> </ul>		QnPRH												
	<p><b>[LINK PARA. ERROR]</b></p> <p>Since the number of points of the B/W device set in [Device] of the PLC parameter is lower than the number of B/W refresh device points shown in the following table when parameters of the MELSECNET/H are not set, the refresh between the CPU module and the MELSECNET/H cannot be performed..</p> <table border="1" data-bbox="244 734 655 994"> <thead> <tr> <th>Refresh device</th> <th>No. of refresh device points of B device</th> <th>No. of refresh device points of W device</th> </tr> </thead> <tbody> <tr> <td rowspan="4">No. of mountable network modules</td> <td>1 8192 points (8192 points×1 module)</td> <td>8192 points (8192 points×1 module)</td> </tr> <tr> <td>2 8192 points (4096 points×2 modules)</td> <td>8192 points (4096 points×2 modules)</td> </tr> <tr> <td>3 6144 points (2048 points×3 modules)</td> <td>6144 points (2048 points×3 modules)</td> </tr> <tr> <td>4 8192 points (2048 points×4 modules)</td> <td>8192 points (2048 points×4 modules)</td> </tr> </tbody> </table> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	Refresh device	No. of refresh device points of B device	No. of refresh device points of W device	No. of mountable network modules	1 8192 points (8192 points×1 module)	8192 points (8192 points×1 module)	2 8192 points (4096 points×2 modules)	8192 points (4096 points×2 modules)	3 6144 points (2048 points×3 modules)	6144 points (2048 points×3 modules)	4 8192 points (2048 points×4 modules)	8192 points (2048 points×4 modules)	<p>Set the refresh parameter of the MELSECNET/H in accordance with the number of points of B/W devices set in [Device] of the PLC parameter.</p>	<p>RUN: Off ERR.: Flicker CPU Status: Stop</p>	<p>Qn(H)<sup>*7</sup> QnPH<sup>*7</sup> QnPRH<sup>*7</sup> QnU</p>
	Refresh device	No. of refresh device points of B device	No. of refresh device points of W device													
No. of mountable network modules	1 8192 points (8192 points×1 module)	8192 points (8192 points×1 module)														
	2 8192 points (4096 points×2 modules)	8192 points (4096 points×2 modules)														
	3 6144 points (2048 points×3 modules)	6144 points (2048 points×3 modules)														
	4 8192 points (2048 points×4 modules)	8192 points (2048 points×4 modules)														
<p><b>[LINK PARA. ERROR]</b></p> <p>The setting of the network refresh range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W).</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<p>Set the network refresh range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).</p>		QnU													

\*7 The module whose first 5 digits of serial No. is "09012" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3102	<p><b>[LINK PARA. ERROR]</b> A CC-Link IE controller network parameter error was detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Correct and write the network parameters.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*7</sup> QnPH <sup>*9</sup> QnPRH <sup>*9</sup> QnU
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The network module detected a network parameter error.</li> <li>• A MELSECNET/H network parameter error was detected.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• If the error occurs after correction, it suggests a hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU
	<p><b>[LINK PARA. ERROR]</b> The station No. specified in pairing setting are not correct.</p> <ul style="list-style-type: none"> <li>• The stations are not numbered consecutively.</li> <li>• Pairing setting has not been made for the CPU module at the normal station.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Refer to the troubleshooting of the network module, and if the error is due to incorrect pairing setting, reexamine the pairing setting of the network parameter.		QnPRH
	<p><b>[LINK PARA. ERROR]</b> The CC-Link IE controller network module whose first 5 digits of serial No. is "09041" or earlier is mounted.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Mount the CC-Link IE controller network module whose first 5 digits of serial No. is "09042" or later.		QnU
	<p><b>[LINK PARA. ERROR]</b> Group cyclic function in CC-Link IE controller network that does not correspond to group cyclic function is set.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Set group cyclic function in function version D or later of CC-Link IE controller network.		QnU <sup>*9</sup>
	<p><b>[LINK PARA. ERROR]</b> Pairing setting in CC-Link IE controller network modules installed in CPUs except for redundant CPUs was performed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Examine the pairing setting for the network parameter in the control station.		Q00J/Q00/Q01 Qn(H) <sup>*9</sup> QnPH <sup>*9</sup> QnU <sup>*9</sup>

\*7 The module whose first 5 digits of serial No. is "09012" or later.

\*9 The module whose first 5 digits of serial No. is "10042" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3102	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>LB/LW own station send range at LB/LW4000 or later was set.</li> <li>LB/LW setting (2) was performed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	Examine the network range assignments for the network parameter in the control station.		Q00J/Q00/Q01
3103	<p><b>[LINK PARA. ERROR]</b></p> <p>In a multiple CPU system, Ethernet interface module under control of another station is specified to the start I/O number of the Ethernet network parameter.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Delete the Ethernet network parameter of Ethernet interface module under control of another station.</li> <li>Change the setting to the start I/O number of Ethernet interface module under control of the host station.</li> </ul>		Q00/Q01*1 Qn(H)*1 QnPH QnU*10
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>Although the number of modules has been set to one or greater number in the Ethernet module count parameter setting, the number of actually mounted module is zero.</li> <li>The start I/O No. of the Ethernet network parameter differs from the I/O No. of the actually mounted module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Correct and write the network parameters.</li> <li>If the error occurs after correction, it suggests a hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	QCPU
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>Ethernet module whose network type is set to "Ethernet (main base)" is mounted on the extension base unit in the redundant system.</li> <li>Ethernet module whose network type is set to "Ethernet (extension base)" is mounted on the main base unit in the redundant system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>			QnPRH*7
3104	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>The Ethernet, MELSECNET/H and MELSECNET/10 use the same network number.</li> <li>The network number, station number or group number set in the network parameter is out of range.</li> <li>The specified I/O number is outside the range of the used CPU module.</li> <li>The Ethernet-specific parameter setting is not normal.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name/ Drive name</li> <li>Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>Correct and write the network parameters.</li> <li>If the error occurs after correction, it suggests a hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>		QCPU

\*1 The function version is B or later.

\*7 The module whose first 5 digits of serial No. is "09012" or later.

\*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3105	<p><b>[LINK PARA. ERROR]</b> In a multiple CPU system, the CC-Link module under control of another station is specified as the head I/O number of the CC-Link network parameter.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Delete the CC-Link network parameter of the CC-Link module under control of another station.</li> <li>• Change the setting to the start I/O number of the CC-Link module under control of the host station.</li> </ul>		<p>Q00/Q01<sup>*1</sup></p> <p>Qn(H)<sup>*1</sup></p> <p>QnPH</p> <p>QnU<sup>*10</sup></p>
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• Though the number of CC-Link modules set in the network parameters is one or more, the number of actually mounted modules is zero.</li> <li>• The start I/O number in the common parameters is different from that of the actually mounted module.</li> <li>• The station type of the CC-Link module count setting parameters is different from that of the actually mounted station.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Correct and write the network parameters.</li> <li>• If the error occurs after correction, it suggests a hardware fault. (Contact your local Mitsubishi representative.)</li> </ul>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>QCPU</p>
	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• CC-Link module whose station type is set to "master station (compatible with redundant function)" is mounted on the extension base unit in the redundant system.</li> <li>• CC-Link module whose station type is set to "master station (extension base)" is mounted on the main base unit in the redundant system.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>			<p>QnPRH<sup>*7</sup></p>

\*1 The function version is B or later.

\*7 The module whose first 5 digits of serial No. is "09012" or later.

\*10 The Universal model QCPU except the Q00UJCPU.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3106	<p><b>[LINK PARA. ERROR]</b> The CC-Link link refresh range exceeded the file register capacity.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Change the file register file for the one refresh-enabled in the whole range.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*1</sup> QnPH QnPRH QnU
	<p><b>[LINK PARA. ERROR]</b> The network refresh parameter for CC-Link is out of range.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Check the parameter setting.		QCPU
	<p><b>[LINK PARA. ERROR]</b> The setting of the network refresh range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W).</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Set the network refresh range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).		QnU
3107	<p><b>[LINK PARA. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The CC-Link parameter setting is incorrect.</li> <li>• The set mode is not allowed for the version of the mounted CC-Link module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Check the parameter setting.		QCPU
3200	<p><b>[SFC PARA. ERROR]</b> The parameter setting is illegal.</p> <ul style="list-style-type: none"> <li>• Though Block 0 was set to "Automatic start" in the SFC setting of the PLC parameter dialog box, Block 0 does not exist.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.		Q00J/Q00/Q01 <sup>*1</sup> QnPH QnPRH QnU
3201	<p><b>[SFC PARA. ERROR]</b> The block parameter setting is illegal.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			Qn(H) QnPH QnPRH

\*1 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3202	<p><b>[SFC PARA. ERROR]</b> The number of step relays specified in the device setting of the PLC parameter dialog box is less than that used in the program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.		Qn(H) QnPH QnPRH
3203	<p><b>[SFC PARA. ERROR]</b> The execution type of the SFC program specified in the program setting of the PLC parameter dialog box is other than scan execution.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/ At reset/ STOP → RUN*3</li> </ul>			Qn(H) QnPH QnPRH QnU
3300	<p><b>[SP. PARA ERROR]</b> The start I/O number in the intelligent function module parameter set on GX Configurator differs from the actual I/O number.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number*2</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Check the parameter setting.	RUN: Off ERR.: Flicker  CPU Status: Stop	QCPU
3301	<p><b>[SP. PARA ERROR]</b> • The refresh setting of the intelligent function module exceeded the file register capacity. • The intelligent function module set in GX Configurator differs from the actually mounted module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number*2</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Change the file register file for the one which allows refresh in the whole range.</li> <li>• Check the parameter setting.</li> </ul>		Q00J/Q00/Q01 Qn(H)*1 QnPH QnPRH QnU
	<p><b>[SP. PARA ERROR]</b> The intelligent function module's refresh parameter setting is outside the available range.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number*2</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Check the parameter setting.		QCPU
	<p><b>[SP. PARA ERROR]</b> The setting of the refresh parameter range crosses over the boundary between the internal user device and the extended data register (D) or extended link register (W).</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number*2</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Set the refresh parameter range so that it does not cross over the boundary between the internal user device and the extended data register (D) or extended link register (W).		QnU

\*1 The function version is B or later.

\*2 Parameter No. is the value gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H.

\*3 The diagnostic timing of CPU modules except for Universal QCPU can be performed only when switching the CPU modules to run.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3302	<p><b>[SP. PARA ERROR]</b> The intelligent function module's refresh parameter are abnormal.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name</li> <li>• Individual Information:Parameter number*<sup>2</sup></li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	Check the parameter setting.		QCPU
3303	<p><b>[SP. PARA ERROR]</b> In a multiple CPU system, the automatic refresh setting or other parameter setting was made to the intelligent function module under control of another station.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/ Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/ At reset/ STOP → RUN/ At writing to progurammable controller</li> </ul>	<ul style="list-style-type: none"> <li>• Delete the automatic refresh setting or other parameter setting of the intelligent function module under control of another CPU.</li> <li>• Change the setting to the automatic refresh setting or other parameter setting of the intelligent function module under control of the host CPU.</li> </ul>		Q00/Q01* <sup>1</sup> Qn(H)* <sup>1</sup> QnPH QnU* <sup>10</sup>
3400	<p><b>[REMOTE PASS. ERR.]</b> The head I/O number of the target module of the remote password is set to other than 0<sub>H</sub> to 0FF0<sub>H</sub>.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Change the head I/O number of the target module to be within the 0 <sub>H</sub> to 0FF0 <sub>H</sub> range.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H)* <sup>1</sup> QnPH QnPRH QnU* <sup>7</sup>
	<p><b>[REMOTE PASS. ERR.]</b> The head I/O number of the target module of the remote password is set to other than 0<sub>H</sub> to 07E0<sub>H</sub>.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Change the head I/O number of the target module to be within the 0 <sub>H</sub> to 07E0 <sub>H</sub> range.		Q02U
	<p><b>[REMOTE PASS. ERR.]</b> The head I/O number of the target module of the remote password is outside the following range.</p> <ul style="list-style-type: none"> <li>• Q00JCPU: 0<sub>H</sub> to 1E0<sub>H</sub></li> <li>• Q00CPU/Q01CPU: 0<sub>H</sub> to 3E0<sub>H</sub></li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Change the head I/O number of the target module of the remote password for the number within the following range.		Q00J/Q00/Q01* <sup>1</sup>

\*1 The function version is B or later.

\*2 Parameter No. is the value gained by dividing the head I/O number of parameter in the intelligent function module set by GX Configurator by 10H.

\*7 The module whose first 5 digits of serial No. is "09012" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
3401	<p><b>[REMOTE PASS. ERR.]</b> Position specified as the head I/O number of the remote password file is incorrect due to one of the following reasons:</p> <ul style="list-style-type: none"> <li>• Module is not loaded.</li> <li>• Other than a the intelligent function module (I/O module)</li> <li>• Intelligent function module other than serial communication module, modem interface module or Ethernet module</li> <li>• Serial communication module or Ethernet module of function version A</li> </ul> <p>The intelligent function module where remote password is available is not mounted.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Mount serial communication module, modem interface module or Ethernet module of function version B or later in the position specified in the head I/O No. of the remote password file.		Qn(H) <sup>*1</sup> QnPH QnPRH QnU
	<p><b>[REMOTE PASS. ERR.]</b> Any of the following modules is not mounted on the slot specified for the head I/O number of the remote password.</p> <ul style="list-style-type: none"> <li>• Serial communication module of function version B or later</li> <li>• Ethernet module of function version B or later</li> <li>• Modem interface module of function version B or later</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	Mount any of the following modules in the position specified for the head I/O number of the remote password.	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	Q00J/Q00/Q01 <sup>*1</sup>
	<p><b>[REMOTE PASS. ERR.]</b> Serial communication module, modem interface module or Ethernet module of function version B or later controlled by another CPU was specified in a multiple CPU system.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Change it for the Ethernet module of function version B or later connected by the host CPU.</li> <li>• Delete the remote password setting.</li> </ul>		Qn(H) <sup>*1</sup> QnPH QnU <sup>*10</sup>

\*1 The function version is B or later.

\*10 The Universal model QCPU except the Q00UJCPU.

## 12.1.6 Error code list (4000 to 4999)

The following shows the error messages from the error code 4000 to 4999, the contents and causes of the errors, and the corrective actions for the errors.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4000	<p><b>[INSTRCT. CODE ERR]</b></p> <ul style="list-style-type: none"> <li>The program contains an instruction code that cannot be decoded.</li> <li>An unusable instruction is included in the program.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN When instruction executed</li> </ul>			QCPU
4001	<p><b>[INSTRCT. CODE ERR]</b></p> <p>The program contains a dedicated instruction for SFC although it is not an SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN When instruction executed</li> </ul>			Q00J/Q00/Q01 <sup>*2</sup> Qn(H) QnPH QnPRH QnU
4002	<p><b>[INSTRCT. CODE ERR]</b></p> <ul style="list-style-type: none"> <li>The name of dedicated instruction specified by the program is incorrect.</li> <li>The dedicated instruction specified by the program cannot be executed by the specified module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN When instruction executed</li> </ul>	Read the common information of the error using a peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	
4003	<p><b>[INSTRCT. CODE ERR]</b></p> <p>The number of devices for the dedicated instruction specified by the program is incorrect.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN When instruction executed</li> </ul>			QCPU
4004	<p><b>[INSTRCT. CODE ERR]</b></p> <p>The device which cannot be used by the dedicated instruction specified by the program is specified.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/STOP → RUN When instruction executed</li> </ul>			

12

12.1 Error Code List  
12.1.6 Error code list (4000 to 4999)

\*2 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4010	<p><b>[MISSING END INS.]</b> There is no END (FEND) instruction in the program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>	<p>Read the common information of the error using a peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.</p>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QCPU
4020	<p><b>[CAN'T SET(P)]</b> The total number of internal file pointers used by the program exceeds the number of internal file pointers set in the parameters.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>			Qn(H) QnPH QnPRH QnU
4021	<p><b>[CAN'T SET(P)]</b></p> <ul style="list-style-type: none"> <li>• The common pointer Nos. assigned to files overlap.</li> <li>• The local pointer Nos. assigned to files overlap.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>			QCPU
4030	<p><b>[CAN'T SET(I)]</b> The allocation pointer Nos. assigned by files overlap.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/STOP → RUN</li> </ul>			QCPU
4100	<p><b>[OPERATION ERROR]</b> The instruction cannot process the contained data.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			QCPU
	<p><b>[OPERATION ERROR]</b> Access error of ATA card occurs by SP.FREAD/SP.FWRITE instructions.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Take measurements against noise.</li> <li>• Reset and restart the CPU module.</li> </ul> <p>When the same error is displayed again, the ATA card has hardware failure. (Please consult your local Mitsubishi service center or representative, explaining a detailed description of the problem.)</p>	<p>RUN: Off/On ERR.: Flicker/On</p> <p>CPU Status: Stop/ Continue*1</p>	Qn(H) QnPH QnPRH QnU*11
	<p><b>[OPERATION ERROR]</b> The file being accessed by other functions with SP.FWRITE instruction was accessed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Stop the file accessed with other functions to execute SP.FWRITE instruction.</li> <li>• Stop the access with other functions and the SP.FWRITE instruction to execute at same time.</li> </ul>	QnU*11	

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*11 The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4101	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>The number of setting data dealt with the instruction exceeds the applicable range.</li> <li>The storage data and constant of the device specified by the instruction exceeds the applicable range.</li> <li>When writing to the host CPU shared memory, the write prohibited area is specified for the write destination address.</li> <li>The range of storage data of the device specified by the instruction is duplicated.</li> <li>The device specified by the instruction exceeds the range of the number of device points.</li> <li>The interrupt pointer No. specified by the instruction exceeds the applicable range.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.	RUN: Off/On ERR.: Flicker/On	QCPU
	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>The storage data of file register specified by the instruction exceeds the applicable range. Or, file register is not set.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>		CPU Status: Stop/ Continue*1	QnU*10
	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>The block data that crosses over the boundary between the internal user device and the extended data register (D) or extended link register is specified (including 32-bit binary, real number (single precision, double precision), indirect address, and control data)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>		QnU	

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)  
\*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4102	<p><b>[OPERATION ERROR]</b> In a multiple CPU system, the link direct device (J□□) was specified for the network module under control of another station.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Delete from the program the link direct device which specifies the network module under control of another CPU.</li> <li>• Using the link direct device, specify the network module under control of the host CPU.</li> </ul>	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue*1	Q00/Q01*2 Qn(H)*2 QnPH QnU*10
	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>• The network No. or station No. specified for the dedicated instruction is wrong.</li> <li>• The link direct device (J□□) setting is incorrect.</li> <li>• The module No./ network No./number of character strings exceeds the range that can be specified.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			QCPU
	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>• The specification of character string (" ") specified by dedicated instruction cannot be used for the character string.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.		QnU
4103	<p><b>[OPERATION ERROR]</b> The configuration of the PID dedicated instruction is incorrect.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			Q00J/Q00/ Q01*2 Qn(H) QnPRH QnU
4105	<p><b>[OPERATION ERROR]</b> PLOADP/PUNLOADP/PSWAPP instructins were executed while setting program memory check.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Delete the program memory check setting.</li> <li>• When using the program memory check, delete PLOADP/PUNLOADP/PSWAPP instructions.</li> </ul>		QnPH*5
4107	<p><b>[OPERATION ERROR]</b> 33 or more multiple CPU dedicated instructions were executed from one CPU module.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Using the multiple CPU dedicated instruction completion bit, provide interlocks to prevent one CPU module from executing 33 or more multiple CPU dedicated instructions.		Q00/Q01*2 Qn(H)*2 QnPH Q00U/Q01U Q02U

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*2 The function version is B or later.

\*5 The module whose first 5 digits of serial No. is "07032" or later.

\*10 The Universal model QCPU except the Q00UJCPU.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4109	<p><b>[OPERATION ERROR]</b> With high speed interrupt setting PR, PRC, UDCNT1, UDCNT2, PLSY or PWM instruction is executed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Delete the high-speed interrupt setting. When using high-speed interrupt, delete the PR, PRC, UDCNT1, UDCNT2, PLSY and PWM instructions.		Qn(H) <sup>*3</sup>
4111	<p><b>[OPERATION ERROR]</b> An attempt was made to perform write/read to/from the CPU shared memory write/read disable area of the host station CPU module with the instruction.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using GX Developer, and check and correct the error step corresponding to that value (program error location).	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue <sup>*1</sup>	Q00/Q01 <sup>*2</sup> QnU
4112	<p><b>[OPERATION ERROR]</b> The CPU module that cannot be specified with the multiple CPU dedicated instruction was specified.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			Q00/Q01 <sup>*2</sup> QnU <sup>*10</sup>
4113	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>• When the SP.DEVST instruction is executed, the number of writing to the standard ROM of the day exceeds the value specified by SD695.</li> <li>• The value outside the specified range is set to SD695.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Check that the number of execution of the SP.DEVST instruction is proper.</li> <li>• Execute the SP.DEVST instruction again the following day or later day. Or, arrange the value of SD695.</li> <li>• Correct the value of SD695 so that it does not exceed the range.</li> </ul>	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/Continue	QnU
4120	<p><b>[OPERATION ERROR]</b> Since the manual system switching enable flag (special register SM1592) is OFF, manual system switching cannot be executed by the control system switching instruction (SP. CONTSW).</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	To execute control system switching by the SP. CONTSW instruction, turn ON the manual system switching enable flag (special register SM1592).	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue <sup>*1</sup>	QnPRH
4121	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>• In the separate mode, the control system switching instruction (SP. CONTSW) was executed in the standby system CPU module.</li> <li>• In the debug mode, the control system switching instruction (SP. CONTSW) was executed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>• Reexamine the interlock signal for the SP. CONTSW instruction, and make sure that the SP. CONTSW instruction is executed in the control system only. (Since the SP. CONTSW instruction cannot be executed in the standby system, it is recommended to provide an interlock using the operation mode signal or like.)</li> <li>• As the SP. CONTSW instruction cannot be executed in the debug mode, reexamine the interlock signal related to the operation mode.</li> </ul>	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue <sup>*1</sup>	QnPRH

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*2 The function version is B or later.

\*3 The module whose first 5 digits of serial No. is "04012" or later.

\*10 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4122	<p><b>[OPERATION ERROR]</b></p> <ul style="list-style-type: none"> <li>The dedicated instruction was executed to the module mounted on the extension base unit in the redundant system.</li> <li>The instruction for accessing the intelligent function module mounted on the extension base unit from the standby system at separate mode was executed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>Delete the dedicated instruction for the module mounted on the extension base unit.</li> <li>Delete the instruction for accessing the intelligent function module mounted on the extension base unit from the standby system.</li> </ul>	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/Continue	QnPRH <sup>*6</sup>
4130	<p><b>[OPERATION ERROR]</b></p> <p>Instructions to read SFC step comment (S(P).SFCSCOMR) and SFC transition condition comment (S(P).SFCTCOMR) are executed for the comment file in ATA card</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When END/other instruction executed</li> </ul>	Target comment file is to be other than the comment file in ATA card.	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue <sup>*1</sup>	Qn(H) <sup>*4</sup> QnPH <sup>*5</sup> QnPRH
4131	<p><b>[OPERATION ERROR]</b></p> <p>The SFC program is started up by the instruction while the other SFC program has not yet been completed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Check the SFC program specified by the instruction. Or, check the executing status of the SFC program.	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/Continue	
4140	<p><b>[OPERATION ERROR]</b></p> <p>Operation where the input data is special value ("0", unnormalized number, nonnumeric, ±∞) is performed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check the error step corresponding to the numerical value (program error part), and correct it.	RUN: Off/On ERR.: Flicker/On	QnU
4141	<p><b>[OPERATION ERROR]</b></p> <p>Overflow occurs at operation.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check the error step corresponding to the numerical value (program error part), and correct it.	CPU Status: Stop/ Continue <sup>*1</sup>	
4200	<p><b>[FOR NEXT ERROR]</b></p> <p>No NEXT instruction was executed following the execution of a FOR instruction.</p> <p>Alternatively, there are fewer NEXT instructions than FOR instructions.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.	RUN: Off ERR.: Flicker  CPU Status: Stop	QCPU

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*4 The module whose first 5 digits of serial No. is "07012" or later.

\*5 The module whose first 5 digits of serial No. is "07032" or later.

\*6 The module whose first 5 digits of serial No. is "09012" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4201	<p><b>[FOR NEXT ERROR]</b> A NEXT instruction was executed although no FOR instruction has been executed. Alternatively, there are more NEXT instructions than FOR instructions.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.	RUN: Off ERR.: Flicker  CPU Status: Stop	QCPU
4202	<p><b>[FOR NEXT ERROR]</b> More than 16 nesting levels are programmed.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Keep nesting levels at 16 or under.		
4203	<p><b>[FOR NEXT ERROR]</b> A BREAK instruction was executed although no FOR instruction has been executed prior to that.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.		
4210	<p><b>[CAN'T EXECUTE(P)]</b> The CALL instruction is executed, but there is no subroutine at the specified pointer.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			
4211	<p><b>[CAN'T EXECUTE(P)]</b> There was no RET instruction in the executed subroutine program.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.		
4212	<p><b>[CAN'T EXECUTE(P)]</b> The RET instruction exists before the FEND instruction of the main routine program.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			
4213	<p><b>[CAN'T EXECUTE(P)]</b> More than 16 nesting levels are programmed.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Keep nesting levels at 16 or under.		

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU		
4220	<p><b>[CAN'T EXECUTE(I)]</b> Though an interrupt input occurred, the corresponding interrupt pointer does not exist.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<p>Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.</p>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QCPU		
4221	<p><b>[CAN'T EXECUTE(I)]</b> An IRET instruction does not exist in the executed interrupt program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>					
4223	<p><b>[CAN'T EXECUTE(I)]</b> The IRET instruction exists before the FEND instruction of the main routine program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>					
	<p><b>[CAN'T EXECUTE(I)]</b></p> <ul style="list-style-type: none"> <li>• The IRET instruction was executed in the fixed scan execution type program.</li> <li>• The STOP instruction was executed in the fixed scan execution type program.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			QnU		
4225	<p><b>[CAN'T EXECUTE(I)]</b> The interrupt pointer for the module mounted on the extension base unit is set in the redundant system.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-ON/At reset</li> </ul>			<p>Delete the setting of interrupt pointer for the module mounted on the extension base unit, since it cannot be used.</p>		QnPRH*6
4230	<p><b>[INST. FORMAT ERR.]</b> The number of CHK and CHKEND instructions is not equal.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			<p>Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.</p>		Qn(H) QnPH
4231	<p><b>[INST. FORMAT ERR.]</b> The number of IX and IXEND instructions is not equal.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>		QCPU			

\*6 The module whose first 5 digits of serial No. is "09012" or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4235	<p><b>[INST. FORMAT ERR.]</b> The configuration of the check conditions for the CHK instruction is incorrect. Alternatively, a CHK instruction has been used in a low speed execution type program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			Qn(H) QnPH
4350	<p><b>[MULTI-COM.ERROR]</b></p> <ul style="list-style-type: none"> <li>• The multiple CPU high-speed transmission dedicated instruction used in the program specifies the wrong CPU module. Or, the setting in the CPU module is incompatible with the multiple CPU high-speed transmission dedicated instruction.</li> <li>• The reserved CPU is specified.</li> <li>• The uninstalled CPU is specified.</li> <li>• The head I/O number of the target CPU/16 (n1) is outside the range of 3E<sub>H</sub> to 3E3<sub>H</sub>.</li> <li>• The CPU module where the instruction cannot be executed is specified.</li> <li>• The instruction is executed in a single CPU system.</li> <li>• The host CPU is specified.</li> <li>• The instruction is executed without setting the "Use multiple CPU high speed communication".</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnU <sup>*7</sup>
4351	<p><b>[MULTI-COM.ERROR]</b></p> <ul style="list-style-type: none"> <li>• The multiple CPU high-speed transmission dedicated instruction specified by the program cannot be executed to the specified target CPU module.</li> <li>• The instruction name is wrong.</li> <li>• The instruction unsupported by the target CPU module is specified.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			
4352	<p><b>[MULTI-COM.ERROR]</b> The number of devices for the multiple CPU high-speed transmission dedicated instruction specified by the program is wrong.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			

\*7 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4353	<p><b>[MULTI-COM.ERROR]</b> The device which cannot be used for the multiple CPU high-speed transmission dedicated instruction specified by the program is specified.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.		QnU*7
4354	<p><b>[MULTI-COM.ERROR]</b> The character string which cannot be handled by the multiple CPU high-speed transmission dedicated instruction is specified.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			QnU*7
4355	<p><b>[MULTI-COM.ERROR]</b> The number of read/write data (number of request/receive data) for the multiple CPU high-speed transmission dedicated instruction specified by the program is not valid.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			
4400	<p><b>[SFCP. CODE ERROR]</b> No SFCP or SFCPEND instruction in SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Write the program to the CPU module again using GX Developer.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) QnPH QnPRH
4410	<p><b>[CAN'T SET(BL)]</b> The block number designated by the SFC program exceeds the range.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			
4411	<p><b>[CAN'T SET(BL)]</b> Block number designations overlap in SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>		Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU	
4420	<p><b>[CAN'T SET(S)]</b> A step number designated in an SFC program exceeds the range.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			

\*2 The function version is B or later.

\*7 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4421	<p><b>[CAN'T SET(S)]</b> Total number of steps in all SFC programs exceed the maximum.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Write the program to the CPU module again using GX Developer.		Q00J/Q00/Q01* <sup>2</sup> Qn(H) QnPH QnPRH QnU
4422	<p><b>[CAN'T SET(S)]</b> Step number designations overlap in SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			
4423	<p><b>[CAN'T SET(S)]</b> The total number of (maximum step No.+1) of each block exceeds the total number of step relays.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Correct the total number of step relays so that it does not exceed the total number of (maximum step No.+1) of each block.		
4430	<p><b>[SFC EXE. ERROR]</b> The SFC program cannot be executed.</p> <ul style="list-style-type: none"> <li>• The data of the block data setting is illegal.</li> <li>• The SFC data device of the block data setting is beyond the device setting range set in the PLC parameter.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/Drive name</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Write the program to the CPU module again using GX Developer.</li> <li>• After correcting the setting of the SFC data device, write it to the CPU module.</li> <li>• After correcting the device setting range set in the PLC parameter, write it to the CPU module.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	Q00J/Q00/Q01* <sup>2</sup> QnU
4431	<p><b>[SFC EXE. ERROR]</b> The SFC program cannot be executed.</p> <ul style="list-style-type: none"> <li>• The block parameter setting is abnormal.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/Drive name</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Write the program to the CPU module again using GX Developer.		
4432	<p><b>[SFC EXE. ERROR]</b> The SFC program cannot be executed.</p> <ul style="list-style-type: none"> <li>• The structure of the SFC program is illegal.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/Drive name</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			

\*2 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4500	<p><b>[SFCP. FORMAT ERR.]</b> The numbers of BLOCK and BEND instructions in an SFC program are not equal.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Write the program to the CPU module again using the peripheral device.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) QnPH QnPRH
4501	<p><b>[SFCP. FORMAT ERR.]</b> The configuration of the STEP* to TRAN* to TSET to SEND instructions in the SFC program is incorrect.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			
4502	<p><b>[SFCP. FORMAT ERR.]</b> The structure of the SFC program is illegal.</p> <ul style="list-style-type: none"> <li>• STEP1* instruction does not exist in the block of the SFC program.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			
4503	<p><b>[SFCP. FORMAT ERR.]</b> The structure of the SFC program is illegal.</p> <ul style="list-style-type: none"> <li>• The step specified in the TSET instruction does not exist.</li> <li>• In jump transition, the host step number was specified as the destination step number.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	<ul style="list-style-type: none"> <li>• Write the program to the CPU module again using GX Developer.</li> <li>• Read the common information of the error using GX Developer, and check and correct the error step corresponding to that value (program error location).</li> </ul>	Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU	
4504	<p><b>[SFCP. FORMAT ERR.]</b> The structure of the SFC program is illegal.</p> <ul style="list-style-type: none"> <li>• The step specified in the TAND instruction does not exist.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	Write the program to the CPU module again using GX Developer.		

\*2 The function version is B or later.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4505	<p><b>[SFCP. FORMAT ERR.]</b> The structure of the SFC program is illegal.</p> <ul style="list-style-type: none"> <li>In the operation output of a step, the SET Sn/BLmSn or RST Sn/BLmSn instruction was specified for the host step.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>STOP → RUN</li> </ul>	Read the common information of the error using GX Developer, and check and correct the error step corresponding to that value (program error location).	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00J/Q00/Q01* <sup>2</sup> QnU
4506	<p><b>[SFCP. FORMAT ERR.]</b> The structure of the SFC program is illegal.</p> <ul style="list-style-type: none"> <li>In a reset step, the host step number was specified as the destination step.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>STOP → RUN</li> </ul>			
4600	<p><b>[SFCP. OPE. ERROR]</b> The SFC program contains data that cannot be processed.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	Read common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.	RUN: Off/On ERR.: Flicker/On  CPU Status: Stop/ Continue* <sup>1</sup>	Qn(H) QnPH QnPRH
4601	<p><b>[SFCP. OPE. ERROR]</b> Exceeds device range that can be designated by the SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>			
4602	<p><b>[SFCP. OPE. ERROR]</b> The START instruction in an SFC program is preceded by an END instruction.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>			

\*1 CPU operation can be set in the parameters at error occurrence. (LED indication varies.)

\*2 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4610	<p><b>[SFCP. EXE. ERROR]</b> The active step information at presumptive start of the SFC program is incorrect.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>	<p>Read common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.</p> <p>The program is automatically subjected to an initial start.</p>	<p>RUN: On ERR.: On</p> <p>CPU Status: Continue</p>	<p>Qn(H) QnPH QnPRH</p>
4611	<p><b>[SFCP. EXE. ERROR]</b> Key-switch was reset during RUN when presumptive start was designated for SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• STOP → RUN</li> </ul>			
4620	<p><b>[BLOCK EXE. ERROR]</b> Startup was executed at a block in the SFC program that was already started up.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	<p>Read common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.</p> <ul style="list-style-type: none"> <li>• Read the common information of the error using GX Developer, and check and correct the error step corresponding to that value (program error location).</li> <li>• Turn ON if the special relay SM321 is OFF.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU</p>
4621	<p><b>[BLOCK EXE. ERROR]</b> Startup was attempted at a block that does not exist in the SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			
4630	<p><b>[STEP EXE. ERROR]</b> Startup was executed at a block in the SFC program that was already started up.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>			

\*2 The function version is B or later.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
4631	<p><b>[STEP EXE. ERROR]</b></p> <ul style="list-style-type: none"> <li>Startup was attempted at the step that does not exist in the SFC program. Or, the step that does not exist in the SFC program was specified for end.</li> <li>Forced transition was executed based on the transition condition that does not exist in the SFC program. Or, the transition condition for forced transition that does not exist in the SFC program was canceled.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	<ul style="list-style-type: none"> <li>Read the common information of the error using the peripheral device, and check and correct the error step corresponding to that value (program error location).</li> <li>Turn ON if the special relay SM321 is OFF.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>Q00J/Q00/Q01*2 Qn(H) QnPH QnPRH QnU</p>
4632	<p><b>[STEP EXE. ERROR]</b></p> <p>There were too many simultaneous active steps in blocks that can be designated by the SFC program.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>	<p>Read common information of the error using the peripheral device, check error step corresponding to its numerical value (program error location), and correct the problem.</p>	<p>CPU Status: Stop</p>	<p>Qn(H) QnPH QnPRH QnU</p>
4633	<p><b>[STEP EXE. ERROR]</b></p> <p>There were too many simultaneous active steps in all blocks that can be designated.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Program error location</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>When instruction executed</li> </ul>			

\*2 The function version is B or later.

## 12.1.7 Error code list (5000 to 5999)

The following shows the error messages from the error code 5000 to 5999, the contents and causes of the errors, and the corrective actions for the errors.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
5000	<p><b>[WDT ERROR]</b></p> <ul style="list-style-type: none"> <li>The scan time of the initial execution type program exceeded the initial execution monitoring time specified in the PLC RAS setting of the PLC parameter.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Time (value set)</li> <li>Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Read the individual information of the error from the peripheral device, check its value (time), and shorten the scan time.</li> <li>Change the initial execution monitoring time or the WDT value in the PLC RAS setting of the PLC parameter.</li> <li>Resolve the endless loop caused by jump transition.</li> </ul>	RUN: Off ERR.: Flicker	Qn(H) QnPH QnPRH QnU
	<p><b>[WDT ERROR]</b></p> <ul style="list-style-type: none"> <li>The power supply of the standby system is turned OFF.</li> <li>The tracking cable is disconnected or connected without turning off or resetting the standby system.</li> <li>The tracking cable is not secured by the connector fixing screws.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Time (value set)</li> <li>Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Since power-off of the standby system increases the control system scan time, reset the WDT value, taking the increase of the control system scan time into consideration.</li> <li>When the tracking cable is disconnected during operation, securely connect it and restart the CPU module. If the same error is displayed again, the tracking cable or CPU module has a hardware fault. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		QnPRH
5001	<p><b>[WDT ERROR]</b></p> <ul style="list-style-type: none"> <li>The scan time of the program exceeded the WDT value specified in the PLC RAS setting of the PLC parameter.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Time (value set)</li> <li>Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Read the individual information of the error using the peripheral device, check its value (time), and shorten the scan time.</li> <li>Change the initial execution monitoring time or the WDT value in the PLC RAS setting of the PLC parameter.</li> <li>Resolve the endless loop caused by jump transition.</li> </ul>	CPU Status: Stop	QCPU
	<p><b>[WDT ERROR]</b></p> <ul style="list-style-type: none"> <li>The power supply of the standby system is turned OFF.</li> <li>The tracking cable is disconnected or connected without turning off or resetting the standby system.</li> <li>The tracking cable is not secured by the connector fixing screws.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Time (value set)</li> <li>Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Since power-off of the standby system increases the control system scan time, reset the WDT value, taking the increase of the control system scan time into consideration.</li> <li>When the tracking cable is disconnected during operation, securely connect it and restart the CPU module. If the same error is displayed again, the tracking cable or CPU module has a hardware fault. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>		QnPRH

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
5010	<p><b>[PRG. TIME OVER]</b> The program scan time exceeded the constant scan setting time specified in the PLC RAS setting of the PLC parameter.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Time (value set)</li> <li>• Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Review the constant scan setting time.</li> <li>• Review the constant scan setting time and low speed program execution time in the PLC parameter so that the excess time of constant scan can be fully secured.</li> </ul>	<p>RUN: On</p> <p>ERR.: On</p> <p>CPU Status: Continue</p>	<p>Qn(H) QnPH QnPRH QnU</p>
	<p><b>[PRG. TIME OVER]</b> The low speed program execution time specified in the PLC RAS setting of the PLC parameter exceeded the excess time of the constant scan.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Time (value set)</li> <li>• Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>			<p>Qn(H) QnPH QnPRH</p>
	<p><b>[PRG. TIME OVER]</b> The program scan time exceeded the constant scan setting time specified in the PLC RAS setting of the PLC parameter.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Time (value set)</li> <li>• Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<p>Review the constant scan setting time in the PLC parameter so that the excess time of constant scan can be fully secured.</p>		<p>Q00J/Q00/Q01</p>
5011	<p><b>[PRG. TIME OVER]</b> The scan time of the low speed execution type program exceeded the low speed execution watch time specified in the PLC RAS setting of the PLC parameter dialog box.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Time (value set)</li> <li>• Individual Information:Time (value actually measured)</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<p>Read the individual information of the error using the peripheral device, check the numerical value (time) there, and shorten scan time if necessary. Change the low speed execution watch time in the PLC RAS setting of the PLC parameter dialog box.</p>		<p>Qn(H) QnPH</p>

## 12.1.8 Error code list (6000 to 6999)

The following shows the error messages from the error code 6000 to 6999, the contents and causes of the errors, and the corrective actions for the errors.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6000	<p><b>[FILE DIFF.]</b> In a redundant system, the control system and standby system do not have the same programs and parameters.</p> <p>The file type detected as different between the two systems can be checked by the file name of the error common information.</p> <ul style="list-style-type: none"> <li>The program is different. (File name = *****.QPG)</li> <li>The PLC parameters/network parameters/redundant parameters are different. (File name = PARAM.QPA)</li> <li>The remote password is different. (File name = PARAM.QPA)</li> <li>The intelligent function module parameters are different. (File name = IPARAM.QPA)</li> <li>The device initial values are different. (File name = *****.QDI)</li> <li>The capacity of each write destination within the CPU for online pchange of multiple program blocks is different. (File name = MBOC.QMB) (This can be detected from the standby system of the redundant system.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:File name</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/ At tracking cable connection/At changing to backup mode/At completion of write during RUN/ At system switching/At switching both systems into RUN</li> </ul>	<ul style="list-style-type: none"> <li>Match the programs and parameters of the control system and standby system.</li> <li>Verify the systems by either of the following procedures 1), 2) to clarify the differences between the files of the two systems, then correct a wrong file, and execute "Write to PLC" again.</li> </ul> <ol style="list-style-type: none"> <li>After reading the programs/parameters of System A using GX Developer or PX Developer, verify them with those of System B.</li> <li>Verify the programs/parameters of GX Developer or PX Developer saved in the offline environment with those written to the CPU modules of both systems.</li> </ol> <ul style="list-style-type: none"> <li>When the capacity of each write destination within the CPU for online change of multiple program blocks is different between the two systems, take corrective action 1) or 2).</li> </ul> <ol style="list-style-type: none"> <li>Using the memory copy from control system to standby system, copy the program memory from the control system to the standby system.</li> <li>Format the CPU module program memories of both systems. (For the capacity of each write destination within the CPU for online change of multiple program blocks, set the same value to both systems.)</li> </ol>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnPRH
6001	<p><b>[FILE DIFF.]</b> In a redundant system, the valid parameter drive settings (SW2, SW3) made by the DIP switches are not the same.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset/At tracking cable connection/At operation mode change</li> </ul>	Match the valid parameter drive settings (SW2, SW3) by the DIP switches of the control system and standby system.		

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6010	<p><b>[OPE. MODE DIFF.]</b> The operational status of the control system and standby system in the redundant system is not the same. (This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Synchronise the operation statuses of the control system and standby system.	RUN: On ERR.: On  CPU Status: Continue	QnPRH
6020	<p><b>[OPE. MODE DIFF.]</b> At power ON/reset, the RUN/STOP switch settings of the control system and standby system are not the same in a redundant system. (This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Set the RUN/STOP switches of the control system and standby system to the same setting.		
6030	<p><b>[UNIT LAY. DIFF.]</b></p> <ul style="list-style-type: none"> <li>• In a redundant system, the module configuration differs between the control system and standby system.</li> <li>• The network module mode setting differs between the two systems.</li> </ul> <p>(This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/At tracking cable connection/At operation mode change</li> </ul>	<ul style="list-style-type: none"> <li>• Match the module configurations of the control system and standby system.</li> <li>• In the redundant setting of the network parameter dialog box, match the mode setting of System B to that of System A.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	
6035	<p><b>[UNIT LAY. DIFF.]</b> In a redundant system, the CPU module model name differs between the control system and standby system. (This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/At tracking cable connection/At operation mode change</li> </ul>	Match the model names of the control system and standby system.		

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6036	<p><b>[UNIT LAY. DIFF.]</b> A difference in the remote I/O configuration of the MELSECNET/H multiplexed remote I/O network between the control system and standby system of a redundant system was detected. (This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information • Common Information:Module No. • Individual Information:–</p> <p>■Diagnostic Timing • Always</p>	Check the network cables of the MELSECNET/H multiplexed remote I/O network for disconnection.		
6040	<p><b>[CARD TYPE DIFF.]</b> In a redundant system, the memory card installation status (installed/not installed) differs between the control system and standby system.</p> <p>■Collateral information • Common Information:– • Individual Information:–</p> <p>■Diagnostic Timing • At power ON/At reset</p>	Match the memory card installation statuses (set/not set) of the control system and standby system.	RUN: Off ERR.: Flicker CPU Status: Stop	QnPRH
6041	<p><b>[CARD TYPE DIFF.]</b> In a redundant system, the memory card type differs between the control system and standby system.</p> <p>■Collateral information • Common Information:– • Individual Information:–</p> <p>■Diagnostic Timing • At power ON/At reset</p>	Match the memory card types of the control system and standby system.		
6050	<p><b>[CAN'T EXE. MODE]</b> The function inexecutable in the debug mode or operation mode (backup/separate mode) was executed. (This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information • Common Information:– • Individual Information:–</p> <p>■Diagnostic Timing • Always</p>	Execute the function executable in the debug mode or operation mode (backup/separate mode).	RUN: On ERR.: On CPU Status: Continue	
6060	<p><b>[CPU MODE DIFF.]</b> In a redundant system, the operation mode (backup/separate) differs between the control system and standby system. (This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information • Common Information:– • Individual Information:–</p> <p>■Diagnostic Timing • At power ON/At reset/At tracking cable connection</p>	Match the operation modes of the control system and standby system.	RUN: Off ERR.: Flicker CPU Status: Stop	



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6061	<p><b>[CPU MODE DIFF.]</b>                      In a redundant system, the operation mode (backup/separate) differs between the control system and standby system.                      (This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Match the operation modes of the control system and standby system.	RUN: Off ERR.: Flicker	QnPRH
6062	<p><b>[CPU MODE DIFF.]</b>                      Both System A and B are in the same system status (control system).                      (This can be detected from the system B of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset/At tracking cable connection</li> </ul>	Power the CPU module (System B) which resulted in a stop error, OFF and then ON.	CPU Status: Stop	
6100	<p><b>[TRK. TRANS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• An error (e.g. retry limit exceeded) occurred in tracking data transmission.                      (This error may be caused by tracking cable removal or other system power-off (including reset).)</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Tracking transmission data classification</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> <li>• Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>	RUN: On ERR.: On	
6101	<p><b>[TRK. TRANS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• A timeout error occurred in tracking (data transmission).                      (This error may be caused by tracking cable removal or other system power-off (including reset).)</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed.                      (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Tracking transmission data classification</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>		CPU Status: Continue	

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6102	<p><b>[TRK. TRANS. ERR.]</b> A data sum value error occurred in tracking (data reception). (This can be detected from the control system or standby system of the redundant system.)</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• Always</li> </ul>			
6103	<p><b>[TRK. TRANS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• A data error (other than sum value error) occurred in tracking (data reception). (This error may be caused by tracking cable removal or other system power-off (including reset).)</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> <li>• Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>		
6105	<p><b>[TRK. TRANS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• An error (e.g. retry limit exceeded) occurred in tracking (data transmission). (This error may be caused by tracking cable removal or other system power-off (including reset).)</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Tracking transmission data classification</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• Always</li> </ul>		RUN: On ERR.: On  CPU Status: Continue	QnPRH
6106	<p><b>[TRK. TRANS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• A timeout error occurred in tracking (data transmission). (This error may be caused by tracking cable removal or other system power-off (including reset).)</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Tracking transmission data classification</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> <li>• Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>		

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6107	<p><b>[TRK. TRANS. ERR.]</b> A data sum value error occurred in tracking (data reception). (This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>			
6108	<p><b>[TRK. TRANS. ERR.]</b></p> <ul style="list-style-type: none"> <li>• A data error (other than sum value error) occurred in tracking (data reception). (This error may be caused by tracking cable removal or other system power-off (including reset).)</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>(This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Check the CPU module or tracking cable. If the error still occurs, this indicates the CPU module or tracking cable is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> <li>• Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>		
6110	<p><b>[TRK. SIZE ERROR]</b> The tracking capacity exceeded the allowed range. (This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Tracking capacity excess error factor</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Reexamine the tracking capacity.	RUN: On ERR.: On  CPU Status: Continue	QnPRH
6111	<p><b>[TRK. SIZE ERROR]</b> The control system does not have enough file register capacity for the file registers specified in the tracking settings. (This can be detected from the control system or standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Switch to the file registers of which capacity is greater than the file registers specified in the tracking settings.		
6112	<p><b>[TRK. SIZE ERROR]</b> File registers greater than those of the standby system were tracked and transmitted from the control system. (This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When an END instruction executed</li> </ul>	Switch to the file registers of which capacity is greater than the file registers specified in the tracking settings.		

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6120	<p><b>[TRK. CABLE ERR.]</b></p> <ul style="list-style-type: none"> <li>• A start was made without the tracking cable being connected.</li> <li>• A start was made with the tracking cable faulty.</li> <li>• As the tracking hardware on the CPU module side was faulty, communication with the other system could not be made via the tracking cable. (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<p>Make a start after connecting the tracking cable. If the same error still occurs, this indicates the tracking cable or CPU module side tracking transmission hardware is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</p>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	
6130	<p><b>[TRK. DISCONNECT]</b></p> <ul style="list-style-type: none"> <li>• The tracking cable was removed.</li> <li>• The tracking cable became faulty while the CPU module is running.</li> <li>• The CPU module side tracking hardware became faulty. (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• If the tracking cable was removed, connect the tracking cable to the connectors of the CPU modules of the two systems.</li> <li>• When the error is not resolved after connecting the tracking cable to the connectors of the CPU modules of the two systems and resetting the error, the tracking cable or CPU module side tracking hardware is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>	<p>RUN: On ERR.: On</p> <p>CPU Status: Continue</p>	
6140	<p><b>[TRK.INIT. ERROR]</b></p> <ul style="list-style-type: none"> <li>• The other system did not respond during initial communication at power ON/reset.</li> <li>• The error occurred at a startup since the redundant system startup procedure was not followed. (This can be detected from the control system or standby system of the redundant system.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Power the corresponding CPU module OFF and then ON again, or reset it and then unreset. If the same error still occurs, this indicates the CPU module is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> <li>• Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnPRH
6200	<p><b>[CONTROL EXE.]</b></p> <p>The standby system has been switched to the control system in a redundant system. (Detected by the CPU that was switched from the standby system to the control system)</p> <p>Since this error code does not indicate the error information of the CPU module but indicates its status, the error code and error information are not stored into SD0 to 26, but are stored into the error log every system switching. (Check the error information by reading the error log using GX Developer.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Reason(s) for system switching</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	-	<p>RUN: On ERR.: Off</p> <p>CPU Status: No error</p>	

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6210	<p><b>[STANDBY]</b>                      The control system has been switched to the standby system in a redundant system. (Detected by the CPU that was switched from the control system to the standby system)                      Since this error code does not indicate the error information of the CPU module but indicates its status, the error code and error information are not stored into SD0 to 26, but are stored into the error log every system switching.                      (Check the error information by reading the error log using GX Developer.)</p> <p>■Collateral information                      • Common Information:Reason(s) for system switching                      • Individual Information:–</p> <p>■Diagnostic Timing                      • Always</p>	-	RUN: On ERR.: Off  CPU Status: No error	
6220	<p><b>[CAN'T SWITCH]</b>                      System switching cannot be executed due to standby system error/ tracking cable error/ online module change in execution at separate mode.                      Causes for switching system at control system are as follows:</p> <ul style="list-style-type: none"> <li>• System switching by SP. CONTSW instruction</li> <li>• System switching request from network module</li> </ul> <p>■Collateral information                      • Common Information:Reason(s) for system switching                      • Individual Information:Reason(s) for system switching failure</p> <p>■Diagnostic Timing                      • At switching execution</p>	<ul style="list-style-type: none"> <li>• Check the status of the standby system and resolve the error.</li> <li>• Complete the online module change.</li> </ul>	RUN: On ERR.: On  CPU Status: No error	QnPRH
6300	<p><b>[STANDBY SYS. DOWN]</b>                      Any of the following errors was detected in the backup mode.</p> <ul style="list-style-type: none"> <li>• The standby system has not started up in the redundant system.</li> <li>• The standby system has developed a stop error in the redundant system.</li> <li>• The CPU module in the debug mode was connected to the operating control system.                      (This can be detected from the control system of the redundant system.)</li> </ul> <p>■Collateral information                      • Common Information:–                      • Individual Information:–</p> <p>■Diagnostic Timing                      • Always</p>	<ul style="list-style-type: none"> <li>• Check whether the standby system is on or not, and if it is not on, power it on.</li> <li>• Check whether the standby system has been reset or not, and if it has been reset, unreset it.</li> <li>• Check whether the standby system has developed a stop error or not, and if it has developed the error, remove the error factor and restart it.</li> <li>• When the CPU module in the debug mode was connected to the control system operating in the backup mode, make connection so that the control system and standby system are combined correctly.</li> </ul>	RUN: On ERR.: On  CPU Status: Continue	

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6310	<p><b>[CONTROL SYS. DOWN]</b> Any of the following errors was detected in the backup mode.</p> <ul style="list-style-type: none"> <li>The control system has not started up in the redundant system.</li> <li>The control system has developed a stop error in the redundant system.</li> <li>The CPU module in the debug mode was connected to the operating standby system.</li> <li>The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>(This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>The standby system exists but the control system does not exist.</li> <li>Check whether the system other than the standby system is on or not, and if it is not on, power it on.</li> <li>Check whether the system other than the standby system has been reset or not, and if it is has been reset, unreset it.</li> <li>Check whether the system other than the standby system has developed a stop error or not, and if has developed the error, remove the error factor, set the control system and standby system to the same operating status, and restart.</li> <li>When the CPU module in the debug mode was connected to the control system operating in the backup mode, make connection so that the control system and control system are combined correctly.</li> <li>Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnPRH
6311	<p><b>[CONTROL SYS. DOWN]</b></p> <ul style="list-style-type: none"> <li>As consistency check data has not transmitted from the control system in a redundant system, the other system cannot start as a standby system.</li> <li>The error occurred at a startup since the redundant system startup procedure was not followed.</li> </ul> <p>(This can be detected from the standby system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Replace the tracking cable. If the same error still occurs, this indicates the CPU module is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> <li>Confirm the redundant system startup procedure, and execute a startup again.</li> </ul>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	
6312	<p><b>[CONTROL SYS. DOWN]</b> The control system detected the error of the system configuration and informed it to the standby system (host system) in the redundant system.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<p>Restart the system after checking that the connection between base unit and the system configuration (type/number/parameter of module) are correct.</p>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnPRH*1
6313	<p><b>[PRG. MEM. CLEAR]</b> The memory copy from control system to standby system was executed, and the program memory was cleared.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:–</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At execution of the memory copy from control system to standby system</li> </ul>	<p>After the memory copy from control system to standby system is completed, switch power OFF and then ON, or make a reset.</p>	<p>RUN: Off ERR.: Flicker</p> <p>CPU Status: Stop</p>	QnPRH
6400				

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
6410	<p><b>[MEM.COPY EXE]</b> The memory copy from control system to standby system was executed. (This can be detected from the control system of the redundant system.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At execution of the function of copying memory from control system to standby system</li> </ul>	–	RUN: On ERR.: On  CPU Status: Continue	QnPRH
6500	<p><b>[TRK. PARA. ERROR]</b> The file register file specified in the tracking setting of the PLC parameter dialog box does not exist.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Read the individual information of the error using GX Developer, and check and correct the drive name and file name. Create the specified file.	RUN: Off ERR.: Flicker	
6501	<p><b>[TRK. PARA. ERROR]</b> The file register range specified in the device detail setting of the tracking setting of the PLC parameter dialog box exceeded the specified file register file capacity.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:File name/Drive name</li> <li>• Individual Information:Parameter number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Read the individual information of the error using GX Developer, and increase the file register capacity.	CPU Status: Stop	

## 12.1.9 Error code list (7000 to 10000)

The following shows the error messages from the error code 7000 to 10000, the contents and causes of the errors, and the corrective actions for the errors.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
7000	<p><b>[MULTI CPU DOWN]</b></p> <ul style="list-style-type: none"> <li>In the operating mode of a multiple CPU system, a CPU error occurred at the CPU where "All station stop by stop error of CPU " was selected.</li> <li>In a multiple CPU system, a CPU module incompatible with the multiple CPU system was mounted.</li> <li>CPU modules other than CPU No.1 were removed from the base unit in operation, or reset.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	<ul style="list-style-type: none"> <li>Read the individual information of the error using GX Developer, identify the error of the CPU module, and remove the error.</li> <li>Remove the CPU module incompatible with the multiple CPU system from the main base unit.</li> <li>Check the mounting status of CPU modules other than CPU No.1 and whether the CPU modules were reset.</li> </ul>		<p>Q00/Q01*1</p> <p>Qn(H)*1</p> <p>QnPH</p> <p>QnU*6</p>
	<p><b>[MULTI CPU DOWN]</b></p> <p>In a multiple CPU system, CPU other than CPU No.1 cannot be started up due to stop error of the CPU No.1 at power-on, which occurs to CPU No.2 to No.4.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<p>Read the individual information of the error using GX Developer, identify the error of the CPU module, and remove the error.</p>		<p>Q00/Q01*1</p> <p>Qn(H)*1</p> <p>QnPH</p> <p>QnU*6</p>
7002	<p><b>[MULTI CPU DOWN]</b></p> <ul style="list-style-type: none"> <li>There is no response from the target CPU module in a multiple CPU system during initial communication.</li> <li>In a multiple CPU system, a CPU module incompatible with the multiple CPU system was mounted.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reset the CPU module and RUN it again. If the same error is displayed again, this suggests the hardware fault of any of the CPU modules. (Contact your local Mitsubishi representative.)</li> <li>Remove the CPU module incompatible with the multiple CPU system from the main base unit. Or, replace the CPU module incompatible with the multiple CPU system with the compatible one.</li> </ul>	<p>RUN: Off</p> <p>ERR.: Flicker</p> <p>CPU Status: Stop</p>	<p>Q00/Q01*1</p> <p>Qn(H)*1</p> <p>QnPH</p>
	<p><b>[MULTI CPU DOWN]</b></p> <ul style="list-style-type: none"> <li>There is no response from the target CPU module in a multiple CPU system during initial communication.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Reset the CPU module and RUN it again. If the same error is displayed again, this suggests the hardware fault of any of the CPU modules. (Contact your local Mitsubishi representative.)</li> </ul>		<p>QnU*6</p>
7003	<p><b>[MULTI CPU DOWN]</b></p> <p>There is no response from the target CPU module in a multiple CPU system at initial communication stage.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<p>Reset the CPU module and RUN it again. If the same error is displayed again, this suggests the hardware fault of any of the CPU modules. (Contact your local Mitsubishi representative.)</p>		<p>Q00/Q01*1</p> <p>Qn(H)*1</p> <p>QnPH</p>

\*1 The function version is B or later.

\*6 The Universal model QCPU except the Q00UJCPU.



Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
7004	<p><b>[MULTI CPU DOWN]</b> In a multiple CPU system, a data error occurred in communication between the CPU modules.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	<ul style="list-style-type: none"> <li>• Check the system configuration to see if modules are mounted in excess of the number of I/O points.</li> <li>• When there are no problems in the system configuration, this indicates the CPU module hardware is faulty. (Contact your local Mitsubishi representative, explaining a detailed description of the problem.)</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00/Q01*1 QnU*6
7010	<p><b>[MULTI EXE. ERROR]</b></p> <ul style="list-style-type: none"> <li>• In a multiple CPU system, a faulty CPU module was mounted.</li> <li>• In a multiple CPU system, a CPU module incompatible with the multiple CPU system was mounted. (The CPU module compatible with the multiple CPU system was used to detect an error.)</li> <li>• In a multiple CPU system, any of the CPU No. 2 to 4 was reset with power ON. (The CPU whose reset state was cancelled was used to detect an error.)</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Read the individual information of the error using GX Developer, and replace the faulty CPU module.</li> <li>• Replace the CPU module with the one compatible with the multiple CPU system.</li> <li>• Do not reset any of the No. 2 to 4 CPU modules.</li> <li>• Reset CPU No. 1 and restart the multiple CPU system.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00/Q01*1 Qn(H)*1 QnPH QnU*6
	<p><b>[MULTI EXE. ERROR]</b> The PC CPU module-compatible software package (PPC-DRV-01)*5 whose version is 1.06 or earlier is used in a multiple CPU system.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Change the version of the PC CPU module-compatible software package (PPC-DRV-01)*5 to 1.07 or later.	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00/Q01*1
	<p><b>[MULTI EXE. ERROR]</b> The Q172(H)CPU(N) or Q173(H)CPU(N) is mounted on the multiple CPU high-speed main base unit (Q3□DB). (This may result in a module failure.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Replace the Q172(H)CPU(N) and Q173(H)CPU(N) with the Motion CPU compatible with the multiple CPU high-speed main base unit.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H)*4 QnPH*4
	<p><b>[MULTI EXE. ERROR]</b> The Universal model QCPU (except Q02UCPU) and Q172(H)CPU(N) are mounted on the same base unit. (This may result in a module failure.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Check the QCPU and Motion CPU that can be used in a multiple CPU system, and change the system configuration.	RUN: Off ERR.: Flicker  CPU Status: Stop	

\*1 The function version is B or later.

\*4 The module whose first 5 digits of serial No. is "09082" or later.

\*6 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
7011	<p><b>[MULTI EXE. ERROR]</b> Either of the following settings was made in a multiple CPU system.</p> <ul style="list-style-type: none"> <li>Multiple CPU automatic refresh setting was made for the inapplicable CPU module.</li> <li>"I/O sharing when using multiple CPUs" setting was made for the inapplicable CPU module.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Correct the multiple CPU automatic refresh setting.</li> <li>Correct the "I/O sharing when using multiple CPUs" setting.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00/Q01* <sup>1</sup> QnU* <sup>6</sup>
	<p><b>[MULTI EXE. ERROR]</b> The system configuration for using the Multiple CPU high speed transmission function is not met.</p> <ul style="list-style-type: none"> <li>The QnUCPU is not used for the CPU No.1.</li> <li>The Multiple CPU high speed main base unit (Q3□DB) is not used.</li> <li>Points other than 0 is set to the send range for the CPU module incompatible with the multiple CPU high speed transmission function.</li> <li>Points other than 0 is set to the send range for the CPU module incompatible with the multiple CPU.</li> </ul> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Change the system configuration to meet the conditions for using the Multiple CPU high speed transmission function.</li> <li>Set the send range of CPU, that does not correspond to multiple CPU compatible area, at 0 point, when performing automatic refreshing in multiple CPU compatible area.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	QnU* <sup>3</sup>
7013	<p><b>[MULTI EXE. ERROR]</b> The Q172(H)CPU(N) or Q173(H)CPU(N) is mounted to the CPU slot or slots 0 to 2. (The module may break down.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Check the QCPU and Motion CPU that can be used in a multiple CPU system, and change the system configuration.</li> <li>Remove the Motion CPU incompatible with the multiple CPU system.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	QnU
7020	<p><b>[MULTI CPU ERROR]</b> In the operating mode of a multiple CPU system, an error occurred in the CPU where "system stop" was not selected. (The CPU module where no error occurred was used to detect an error.)</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>Always</li> </ul>	Read the individual information of the error using the peripheral device, check the error of the CPU module resulting in CPU module fault, and remove the error.	RUN: On ERR.: On  CPU Status: Continue	Q00/Q01* <sup>1</sup> Qn(H)* <sup>1</sup> QnPH QnU* <sup>6</sup>
7030	<p><b>[CPU LAY. ERROR]</b> An assignment error occurred in the CPU-mountable slot (CPU slot, I/O slot 0, 1) in excess of the number of CPU modules specified in the multiple CPU setting of the PLC parameter dialog box.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>Common Information:Module No.(CPU No.)</li> <li>Individual Information:--</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>Set the same value to the number of CPU modules specified in the multiple CPU setting of the PLC parameter dialog box and the number of mounted CPU modules (including CPU (empty)).</li> <li>Make the type specified in the I/O assignment setting of the PLC parameter dialog box consistent with the CPU module configuration.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00J/Q01/Q01* <sup>1</sup> QnU

\*1 The function version is B or later.

\*3 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*6 The Universal model QCPU except the Q00UJCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
7031	<p><b>[CPU LAY. ERROR]</b> An assignment error occurred within the range of the number of CPUs specified in the multiple CPU setting of the PLC parameter dialog box.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Set the same value to the number of CPU modules specified in the multiple CPU setting of the PLC parameter dialog box and the number of mounted CPU modules (including CPU (empty)).</li> <li>• Make the type specified in the I/O assignment setting of the PLC parameter dialog box consistent with the CPU module configuration.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00J/Q01/Q01* <sup>1</sup> QnU
7032	<p><b>[CPU LAY. ERROR]</b> • The number of CPU modules mounted in a multiple CPU system is wrong.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Configure a system so that the number of mountable modules of each CPU module does not exceed the maximum number of mountable modules specified in the specification.	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00/Q01* <sup>1</sup> QnU <sup>6</sup>
7035	<p><b>[CPU LAY. ERROR]</b> The CPU module has been mounted on the inapplicable slot.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Mount the CPU module on the applicable slot.	RUN: Off ERR.: Flicker  CPU Status: Stop	Q00J/Q00/Q01* <sup>1</sup> QnPRH QnU
7036	<p><b>[CPU RAY ERROR]</b> The host CPU No. set by the multiple CPU setting and the host CPU No. determined by the mounting position of the CPU module are not the same.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Module No.(CPU No.)</li> <li>• Individual Information:–</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	<ul style="list-style-type: none"> <li>• Mount the mounting slot of the CPU module correctly.</li> <li>• Correct the host CPU No. set by the multiple CPU setting to the CPU No. determined by the mounting position of the CPU module.</li> </ul>	RUN: Off ERR.: Flicker  CPU Status: Stop	QnU <sup>3</sup>
8031	<p><b>[INCORRECT FILE]</b> The error of stored file (enabled parameter file) is detected.</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:File diagnostic information</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• At power-On/</li> <li>• At reset/</li> <li>• STOP → RUN/At writing to programmable controller</li> </ul>	Write the file shown as SD17 to SD22 of individual information to the drive shown as SD16(L) of individual information, and turn ON from OFF the power supply of the CPU module or cancel the reset.  If the same error is displayed again, the CPU module has hardware failure. Contact your local Mitsubishi representative, explaining a detailed description of the problem.	RUN: Off ERR.: Flicker  CPU Status: Stop	QnU
9000	<p><b>[F****]</b> Annunciator (F) was set ON</p> <p>■Collateral information</p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:Annunciator number</li> </ul> <p>■Diagnostic Timing</p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the individual information of the error using the peripheral device, and check the program corresponding to the numerical value (annunciator number).	RUN: On ERR.: On/Off * <sup>2</sup>  CPU Status: Continue  RUN: ERR.: USER LED On  CPU Status: Continue	QCPU

\*1 The function version is B or later.

\*3 The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

Error Code (SD0)	Error Contents and Cause	Corrective Action	LED Status CPU Status	Corresponding CPU
9010	<p><b>[&lt;CHK&gt;ERR ***.***]</b> Error detected by the CHK instruction.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:Program error location</li> <li>• Individual Information:Failure No.</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• When instruction executed</li> </ul>	Read the individual information of the error using the peripheral device, and check the program corresponding to the numerical value (error number) there.	RUN: On ERR.: Off USER LED On  CPU Status: Continue	Qn(H) QnPH QnPRH
9020	<p><b>[BOOT OK]</b> Storage of data onto ROM was completed normally in automatic write to standard ROM. (BOOT LED also flickers.)</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• At power ON/At reset</li> </ul>	Use the DIP switches to set the valid parameter drive to the standard ROM. Then, switch power on again, and perform boot operation from the standard ROM.	RUN: Off ERR.: Flicker  CPU Status: Stop	Qn(H) <sup>*1</sup> QnPH QnPRH
10000	<p><b>[CONT. UNIT ERROR]</b> In the multiple CPU system, an error occurred in the CPU module other than the Process CPU/High Performance model QCPU.</p> <p>■<b>Collateral information</b></p> <ul style="list-style-type: none"> <li>• Common Information:–</li> <li>• Individual Information:–</li> </ul> <p>■<b>Diagnostic Timing</b></p> <ul style="list-style-type: none"> <li>• Always</li> </ul>	Check the details of the generated error by connecting to the corresponding CPU module using GX Developer.	RUN: Off ERR.: Flicker  CPU Status: Continue	Qn(H) <sup>*1</sup> QnPH

\*1 The function version is B or later.

## 12.2 Canceling of Errors

---

Q series CPU module can perform the cancel operation for errors only when the errors allow the CPU module to continue its operation.

To cancel the errors, follow the steps shown below.

- 1) Eliminate the cause of the error.
- 2) Store the error code to be canceled in the special register SD50.
- 3) Energize the special relay SM50 (OFF → ON).
- 4) The error to be canceled is canceled.

After the CPU module is reset by the canceling of the error, the special relays, special registers, and LEDs associated with the error are returned to the status under which the error occurred.

If the same error occurs again after the cancellation of the error, it will be registered again in the error history.

When multiple enunciators(F) detected are canceled, the first one with No. F only is canceled.

Refer to the following manual for details of error canceling.

→ QCPU User's Manual (Function Explanation, Program Fundamentals)

---

### POINT

- (1) When the error is canceled with the error code to be canceled stored in the SD50, the lower one digit of the code is neglected.  
(Example)  
If error codes 2100 and 2101 occur, and error code 2100 to cancel error code 2101.  
If error codes 2100 and 2111 occur, error code 2111 is not canceled even if error code 2100 is canceled.
  - (2) Errors developed due to trouble in other than the CPU module are not canceled even if the special relay (SM50) and special register (SD50) are used to cancel the error.  
(Example)  
Since "SP. UNIT DOWN" is the error that occurred in the base unit (including the extension cable), intelligent function module, etc. the error cause cannot be removed even if the error is canceled by the special relay (SM50) and special register (SD50).  
Refer to the error code list and remove the error cause.
-





# APPENDICES

A

# Appendix 1 OPERATION PROCESSING TIME

---

## *Appendix 1.1* Definition

---

- (1) Processing time taken by the QCPU is the total of the following processing times.
  - Total of each instruction processing time
  - END processing time (including I/O refresh time)
  - Processing time for the function that increases the scan time
- (2) Instruction processing time  
This is the total of processing time of each instruction shown in Appendix 1.2, 1.3 and 1.4.
- (3) END processing time, I/O refresh time, and processing time for the function that increases the scan time  
Refer to the following manual(s) for the END processing time, I/O refresh time, and processing time for the function that increases the scan time.
  - (a) For QCPUs
    - QnUCPU User's Manual (Functions Explanation, Program Fundamentals)
    - Qn(H)/QnPH/QnPRHCPU User's Manual (Functions Explanation, Program Fundamentals)



# Appendix 1.2 Operation Processing Time of Basic Model QCPU

The processing time for the individual instructions are shown in the table on the following pages. Operation processing times can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

## (1) Sequence instructions

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
LD LDI AND ANI OR ORI	X0	0.20	0.16	0.10
	D0.0	0.30	0.24	0.15
LDP LDF ANDP ANDF ORP ORF	X0	0.30	0.24	0.15
	D0.0			
ANB ORB MPS MRD MPP	—	0.20	0.16	0.10
	When not executed	0.20	0.16	0.10
When executed				
MEP MEF	When not executed	0.30	0.24	0.15
	When executed			
EGP	When not executed (OFF→OFF) (ON→ON)	0.20	0.16	0.10
	When executed (OFF→ON) (ON→OFF)			
EGF	When not executed (OFF→OFF) (ON→ON)	17	9.5	9.4
	When executed (OFF→ON) (ON→OFF)	18	14	14

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.2 Operation Processing Time of Basic Model QCPU

Instruction		Condition (Device)		Processing Time (μs)			
				Q00JCPU	Q00CPU	Q01CPU	
OUT	Y	When not changed	(OFF→OFF) (ON→ON)	0.20	0.16	0.10	
		When changed	(OFF→ON) (ON→OFF)	0.20	0.16	0.10	
	D0.0	When not changed	(OFF→OFF) (ON→ON)	0.40	0.32	0.20	
		When changed	(OFF→ON) (ON→OFF)	0.40	0.32	0.20	
	F	When OFF		24	20	19	
		When ON	When displayed	260	210	200	
			Display completed	205	165	155	
	T	When not executed		1.1	0.88	0.55	
		When executed	After time up		1.1	0.88	0.55
			When added	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
	C	When not executed		1.1	0.88	0.55	
		When executed	After time up		1.1	0.88	0.55
			When added	K	1.1	0.88	0.55
D				1.2	0.96	0.60	
OUTH	T	When not executed		1.1	0.88	0.55	
		When executed	After time up		1.1	0.88	0.55
			When added	K	1.1	0.88	0.55
				D	1.2	0.96	0.60
SET	Y	When not executed		0.20	0.16	0.10	
		When executed	When not changed (ON→ON)	0.20	0.16	0.10	
			When changed (OFF→ON)	0.20	0.16	0.10	
	D0.0	When not executed		0.40	0.32	0.20	
		When executed	When not changed (ON→ON)	0.40	0.32	0.20	
			When changed (OFF→ON)	0.40	0.32	0.20	
F	When not executed		0.50	0.44	0.25		
	When executed	When displayed	255	205	195		
Display completed		195	160	150			
RST	Y	When not executed		0.20	0.16	0.10	
		When executed	When not changed (OFF→OFF)	0.20	0.16	0.10	
			When changed (ON→OFF)	0.20	0.16	0.10	
	D0.0	When not executed		0.40	0.32	0.20	
		When executed	When not changed (ON→ON)	0.40	0.32	0.20	
			When changed (OFF→ON)	0.40	0.32	0.20	
	SM	When not executed		0.20	0.16	0.10	
		When executed		0.20	0.16	0.10	
	F	When not executed		0.48	0.44	0.25	
		When executed	When displayed	75	69	65	
			Display completed	43	35	33	
	T, C	When not executed		0.80	0.64	0.40	
		When executed		1.0	0.80	0.50	
	D	When not executed		0.40	0.32	0.20	
		When executed		0.60	0.48	0.30	
	Z	When not executed		0.50	0.40	0.25	
When executed		9.4	7.9	7.4			
R	When not executed		—	0.32	0.20		
	When executed		—	0.48	0.30		

Instruction		Condition (Device)	Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
PLS			12	9.5	9.2
PLF			11	9.5	8.9
FF	Y	When not executed	0.68	0.40	0.25
		When executed	7.5	6.2	5.7
DELTA	DY0	When not executed	0.50	0.40	0.25
		When executed	26	21	21
DELTAP	DY0	When not executed	0.48	0.40	0.25
		When executed	58	45	43
SFT		When not executed	0.50	0.34	0.25
SFTP		When executed	12	8.7	8.3
MC		M0	0.40	0.32	0.20
		D0.0	3.3	2.9	2.8
MCR		—	0.20	0.16	0.10
FEND END		Error check performed	660	600	520
		No error check performed (• Battery check) (• Fuse blown check) (• I/O module verification)	660	600	520
NOP		—	0.20	0.16	0.10
NOPLF PAGE		—	0.20	0.16	0.10

A

(2) Basic instructions

The processing time when the instruction is not executed is calculated as follows:

Q00JCPU ..... 0.20 × (No. of steps for each instruction + 1) μs

Q00CPU ..... 0.16 × (No. of steps for each instruction + 1) μs

Q01CPU ..... 0.10 × (No. of steps for each instruction + 1) μs

Instruction		Condition (Device)	Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
LD =	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND =	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR =	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD <>	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND <>	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.2 Operation Processing Time of Basic Model Q0CPU

Instruction	Condition (Device)		Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
OR < >	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD >	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND >	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR >	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD < =	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND < =	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR < =	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD <	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND <	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR <	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LD > =	In conductive status		0.80	0.64	0.40
	In non-conductive status		0.80	0.64	0.40
AND > =	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
OR > =	When not executed		0.70	0.56	0.35
	When executed	In conductive status	0.80	0.64	0.40
		In non-conductive status	0.80	0.64	0.40
LDD =	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD =	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD =	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD < >	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD < >	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50

Instruction	Condition (Device)		Processing Time ( $\mu$ s)		
			Q00JCPU	Q00CPU	Q01CPU
ORD <>	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD >	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD >	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD >	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD <=	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD <=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD <=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD <	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD <	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD <	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
LDD >=	In conductive status		1.0	0.80	0.50
	In non-conductive status		1.0	0.80	0.50
ANDD >=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
ORD >=	When not executed		0.80	0.64	0.40
	When executed	In conductive status	1.0	0.80	0.50
		In non-conductive status	1.0	0.80	0.50
BKCMP = S1 S2 D n	n = 1		130	105	97
BKCMP = P S1 S2 D n	n = 96		205	175	165
BKCMP <> S1 S2 D n	n = 1		130	105	98
BKCMP <> P S1 S2 D n	n = 96		210	180	165
BKCMP > S1 S2 D n	n = 1		130	105	97
BKCMP > P S1 S2 D n	n = 96		210	180	165
BKCMP >= S1 S2 D n	n = 1		130	105	98
BKCMP >= P S1 S2 D n	n = 96		205	175	165

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
BKCMPL S1 S2 D n	n = 1	130	105	98
BKCMPLP S1 S2 D n	n = 96	210	180	165
BKCMPL= S1 S2 D n	n = 1	130	105	97
BKCMPL=P S1 S2 D n	n = 96	205	175	165
+ S D +P S D	When executed	1.0	0.80	0.50
+ S1 S2 D +P S1 S2 D	When executed	1.2	0.96	0.60
- S D - P S D	When executed	1.0	0.80	0.50
- S1 S2 D - P S1 S2 D	When executed	1.2	0.96	0.60
D+ S D D+P S D	When executed	1.3	1.04	0.65
D+ S1 S2 D D+P S1 S2 D	When executed	1.5	1.2	0.75
D- S D D- P S D	When executed	1.3	1.04	0.65
D- S1 S2 D D- P S1 S2 D	When executed	1.5	1.2	0.75
* S1 S2 D * P S1 S2 D	When executed	1.1	0.88	0.55
/ S1 S2 D /P S1 S2 D	—	19	16	15
D* S1 S2 D D* P S1 S2 D	—	41	34	31
D/ S1 S2 D D/P S1 S2 D	—	28	23	21
B+ S D B+P S D	—	34	28	26
B+ S1 S2 D B+P S1 S2 D	—	47	39	37
B- S D B- P S D	—	34	28	26
B- S1 S2 D B- P S1 S2 D	—	48	40	38
DB+ S D DB+P S D	—	58	48	44
DB+ S1 S2 D DB+P S1 S2 D	—	60	49	46
DB- S D DB- P S D	—	59	48	45
DB- S1 S2 D DB- P S1 S2 D	—	60	51	45

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
B * S1 S2 D B * P S1 S2 D	—	42	35	33
B/ S1 S2 D B/P S1 S2 D	—	48	40	37
DB * S1 S2 D DB * P S1 S2 D	—	140	120	110
DB/ S1 S2 D DB/P S1 S2 D	—	83	69	65
BK + S1 S2 D n	n = 1	105	86	80
BK + P S1 S2 D n	n = 96	185	155	140
BK - S1 S2 D n	n = 1	105	86	80
BK - P S1 S2 D n	n = 96	185	155	140
INC INCP	—	0.70	0.56	0.35
DINC DINCP	—	0.90	0.72	0.45
DEC DECP	—	0.70	0.56	0.35
DDEC DDECP	—	0.90	0.72	0.45
BCD BCDP	—	20	16	15
DBCD DBCDP	—	26	21	20
BIN BINP	—	19	16	15
DBIN DBINP	—	22	18	17
DBL DBLP	—	19	16	15
WORD WORDP	—	23	19	17
GRY GRYP	—	19	16	15
DGRY DGRYP	—	23	19	17
GBIN GBINP	—	52	42	40
DGBIN DGBINP	—	110	88	84
NEG NEGP	—	16	13	12
DNEG DNEGP	—	19	17	15

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.2 Operation Processing Time of Basic Model QCPU

Instruction		Condition (Device)	Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
BKBCD	Ⓢ Ⓣ n	n = 1	78	63	57
BKBCDP	Ⓢ Ⓣ n	n = 96	315	275	250
BKBIN	Ⓢ Ⓣ n	n = 1	74	61	57
BKBINP	Ⓢ Ⓣ n	n = 96	285	255	230
MOV		Ⓢ = D0, Ⓣ = D1	0.70	0.56	0.35
MOV P		Ⓢ = D0, Ⓣ = J1 \ W1	155	130	120
DMOV		Ⓢ = D0, Ⓣ = D1	0.90	0.72	0.45
DMOV P		Ⓢ = D0, Ⓣ = J1 \ W1	165	135	120
\$MOV		0 characters	46	38	35
\$MOV P		32 characters	98	80	73
CML		—	0.70	0.56	0.35
CML P		—	0.90	0.72	0.45
DCML		—	0.90	0.72	0.45
DCML P		—	0.90	0.72	0.45
BMOV	Ⓢ Ⓣ n	n = 1	27	21	20
BMOV P	Ⓢ Ⓣ n	n = 96	72	62	53
FMOV	Ⓢ Ⓣ n	n = 1	23	19	17
FMOV P	Ⓢ Ⓣ n	n = 96	48	41	36
XCH		—	7.6	6.3	5.7
XCH P		—	7.6	6.3	5.7
DXCH		—	9.5	8.0	7.1
DXCH P		—	9.5	8.0	7.1
BXCH	Ⓣ1 Ⓣ2 n	n = 1	62	51	48
BXCH P	Ⓣ1 Ⓣ2 n	n = 96	165	140	125
SWAP		—	17	14	13
SWAP P		—	17	14	13
CJ		—	10	8.5	8.1
SCJ		—	10	8.5	8.1
JMP		—	11	8.5	8.1
GOEND		—	3.3	2.9	2.8
DI		—	13	12	11
EI		—	14	11	11
IMASK		—	41	34	35
IRET		—	205	170	155
RFS RFSP	X	n = 1	55	46	43
		n = 96	79	64	59
	Y	n = 1	54	45	41
		n = 96	73	61	56



(3) Application instructions

The processing time when the instruction is not executed is calculated as follows:

- Q00JCPU ..... 0.20 × (No. of steps for each instruction + 1) μs  
 Q00CPU ..... 0.16 × (No. of steps for each instruction + 1) μs  
 Q01CPU ..... 0.10 × (No. of steps for each instruction + 1) μs

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
WAND (S) (D) WANDP (S) (D)	When executed	1.0	0.80	0.50
WAND (S1) (S2) (D) WANDP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DAND (S) (D) DANDP (S) (D)	When executed	1.3	1.04	0.65
DAND (S1) (S2) (D) DANDP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKAND (S1) (S2) (D) n BKANDP (S1) (S2) (D) n	n = 1	110	87	79
	n = 96	185	155	140
WOR (S) (D) WORP (S) (D)	When executed	1.0	0.80	0.50
WOR (S1) (S2) (D) WORP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DOR (S) (D) DORP (S) (D)	When executed	1.3	1.04	0.65
DOR (S1) (S2) (D) DORP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKOR (S1) (S2) (D) n BKORP (S1) (S2) (D) n	n = 1	110	87	81
	n = 96	185	155	140
WXOR (S) (D) WXORP (S) (D)	When executed	1.0	0.80	0.50
WXOR (S1) (S2) (D) WXORP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DXOR (S) (D) DXORP (S) (D)	When executed	1.3	1.04	0.65
DXOR (S1) (S2) (D) DXORP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKXOR (S1) (S2) (D) n BKXORP (S1) (S2) (D) n	n = 1	110	87	81
	n = 96	185	155	140

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.2 Operation Processing Time of Basic Model QCPU

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
WXNR (S) (D) WXNRP (S) (D)	When executed	1.0	0.80	0.50
WXNR (S1) (S2) (D) WXNRP (S1) (S2) (D)	When executed	1.2	0.96	0.60
DXNR (S) (D) DXNRP (S) (D)	When executed	1.3	1.04	0.65
DXNR (S1) (S2) (D) DXNRP (S1) (S2) (D)	When executed	1.5	1.2	0.75
BKXNR (S1) (S2) (D) n BKXNRP (S1) (S2) (D) n	n = 1 n = 96	110 185	87 155	82 140
ROR (D) n RORP (D) n	n = 1 n = 15	13 13	11 11	9.7 9.7
RCR (D) n RCRP (D) n	n = 1 n = 15	15 15	12 13	12 12
ROL (D) n ROLP (D) n	n = 1 n = 15	13 13	11 11	10 10
RCL (D) n RCLP (D) n	n = 1 n = 15	15 16	13 13	12 12
DROR (D) n DRORP (D) n	n = 1 n = 31	15 15	12 13	12 12
DRCR (D) n DRCRP (D) n	n = 1 n = 31	17 18	14 16	14 15
DROL (D) n DROLP (D) n	n = 1 n = 31	14 14	13 13	12 12
DRCL (D) n DRCLP (D) n	n = 1 n = 31	18 20	15 17	14 16
SFR (D) n SFRP (D) n	n = 1 n = 15	13 13	10 11	9.7 9.5
SFL (D) n SFLP (D) n	n = 1 n = 15	12 12	10 9.8	9.5 9.5
BSFLR (D) n BSFLRP (D) n	n = 1 n = 96	42 69	35 58	33 54
BSFL (D) n BSFLP (D) n	n = 1 n = 96	41 63	34 53	32 50
DSFR (D) n DSFRP (D) n	n = 1 n = 96	19 71	16 61	15 53
DSFL (D) n DSFLP (D) n	n = 1 n = 96	19 70	16 60	15 52

Instruction	Condition (Device)		Processing Time ( $\mu$ s)		
			Q00JCPU	Q00CPU	Q01CPU
BSET $\textcircled{D}$ n	n = 1		27	22	20
BSETP $\textcircled{D}$ n	n = 15		27	22	20
BRST $\textcircled{D}$ n	n = 1		27	22	21
BRSTP $\textcircled{D}$ n	n = 15		27	22	21
TEST $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		35	30	27
TESTP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$					
DTEST $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		37	31	28
DTESTP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$					
BKRST $\textcircled{D}$ n	n = 1		49	41	38
BKRSTP $\textcircled{D}$ n	n = 96		64	54	50
SER $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	All match	56	54	42
		None match	56	54	42
SERP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 96	All match	280	240	220
		None match	280	240	220
DSER $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	All match	71	67	53
		None match	71	67	54
DSERP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 96	All match	495	415	375
		None match	500	415	375
SUM SUMP	$\textcircled{S} = 0$		32	26	25
	$\textcircled{S} = \text{FFFFH}$		27	22	21
DSUM DSUMP	$\textcircled{S} = 0$		54	44	42
	$\textcircled{S} = \text{FFFFFFFFH}$		54	44	42
DECO $\textcircled{S}$ $\textcircled{D}$ n	n = 2		60	50	46
DECOP $\textcircled{S}$ $\textcircled{D}$ n	n = 8		80	65	61
ENCO $\textcircled{S}$ $\textcircled{D}$ n	n = 2	M1 = ON	66	55	51
		M4 = ON	66	54	51
ENCOP $\textcircled{S}$ $\textcircled{D}$ n	n = 8	M1 = ON	90	76	71
		M256 = ON	76	74	71
SEG SEGP	—		8.0	6.8	6.1
DIS $\textcircled{S}$ $\textcircled{D}$ n	n = 1		47	39	36
DISP $\textcircled{S}$ $\textcircled{D}$ n	n = 4		53	43	40
UNI $\textcircled{S}$ $\textcircled{D}$ n	n = 1		54	44	41
UNIP $\textcircled{S}$ $\textcircled{D}$ n	n = 4		60	49	46
NDIS $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$	—		92	76	38
NDISP $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$					
NUNI $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$	—		47	39	36
NUNIP $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$					

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
WTOB (S) (D) n	n = 1	56	46	42
WTOBP (S) (D) n	n = 96	190	155	145
BTOW (S) (D) n	n = 1	56	46	42
BTOWP (S) (D) n	n = 96	190	155	145
MAX (S) (D) n	n = 1	48	40	36
MAXP (S) (D) n	n = 96	300	240	235
MIN (S) (D) n	n = 1	48	40	36
MINP (S) (D) n	n = 96	300	240	235
DMAX (S) (D) n	n = 1	52	43	39
DMAXP (S) (D) n	n = 96	600	490	460
DMIN (S) (D) n	n = 1	52	43	39
DMINP (S) (D) n	n = 96	585	475	445
SORT (S1) n (S2) (D1) (D2)	n = 1	66	55	50
	n = 96	105	86	80
DSORT (S1) n (S2) (D1) (D2)	n = 1	98	57	52
	n = 96	115	96	88
WSUM (S) (D) n	n = 1	52	43	40
WSUMP (S) (D) n	n = 96	175	140	135
DWSUM (S) (D) n	n = 1	61	51	46
DWSUMP (S) (D) n	n = 96	515	420	395
FOR n	n = 0	11	8.9	8.1
NEXT	—	8.8	7.3	6.8
BREAK	—	37	30	28
BREAKP				
CALL Pn CALLP Pn	—	17	14	13
CALL Pn (S1) to (S5)	—	245	200	190
CALLP Pn (S1) to (S5)				
RET	Return to original program	16	13	12
FCALL Pn FCALLP Pn	—	29	24	22
FCALL Pn (S1) to (S5) FCALLP Pn (S1) to (S5)	—	250	205	190

Instruction	Condition (Device)	Processing Time ( $\mu$ s)		
		Q00JCPU	Q00CPU	Q01CPU
COM	—	110	77	72
IX	—	65	54	51
IXEND	—	30	26	25
IXDEV + IXSET	Number of contacts 1	145	120	110
	Number of contacts 14	770	630	585
FIFW FIFWP	Number of data points 0	36	32	28
	Number of data points 96	36	32	28
FIFR FIFRP	Number of data points 1	45	41	36
	Number of data points 96	93	82	70
FPOP FPOPP	Number of data points 1	40	37	32
	Number of data points 96	40	37	32
FINS FINSP	Number of data points 0	53	44	38
	Number of data points 96	100	89	76
FDEL FDELP	Number of data points 1	60	50	43
	Number of data points 96	110	95	82
FROM n1 n2 (D) n3	n3 = 1	125	105	93
FROMP n1 n2 (D) n3 *1	n3 = 1000	740	695	685
DFRO n1 n2 (D) n3	n3 = 1	130	110	100
DFROP n1 n2 (D) n3 *1	n3 = 500	745	695	675
TO n1 n2 (S) n3	n3 = 1	120	105	92
TOP n1 n2 (S) n3 *1	n3 = 1000	735	680	645
DTO n1 n2 (S) n3	n3 = 1	130	110	99
DTOP n1 n2 (S) n3 *1	n3 = 500	740	680	640

\*1 : The FROM/TO instruction differs in processing time according to the number of slots and the loaded modules. (The CPU also differs in processing time according to the extension base type.)

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
LIMIT LIMITP	—	34	28	26
DLIMIT DLIMITP	—	41	34	30
BAND BANDP	—	33	28	25
DBAND DBANDP	—	40	34	30
ZONE ZONEP	—	31	25	24
DZONE DZONEP	—	37	29	28
RSET RSETP	—	—	18	16
DATERD DATERDP	—	30	25	23
DATEWR DATEWRP	—	69	57	54
DATE+ DATE+P	No digit increase	47	39	36
	Digit increase	50	42	38
DATE - DATE - P	No digit increase	47	40	36
	Digit increase	50	42	38
SECOND SECONDP	—	28	24	22
HOUR HOURP	—	38	32	29
WDT WDTP	—	18	15	14
DUTY	—	41	36	32
ZRRDB ZRRDBP	—	—	24	22
ZRWRB ZRWRBP	—	—	27	24
ADRSET ADRSETP	—	23	19	18
ZPUSH ZPUSHP	—	38	33	30
ZPOP ZPOPP	—	37	31	29
ZCOM	—	105	82	80

(4) Processing time for QCPU instructions (QCPU instructions only)

Instruction	Condition (Device)	Processing Time (μs)		
		Q00JCPU	Q00CPU	Q01CPU
UNIRD	n = 1	96	80	74
UNIRDP	n = 16	440	370	340

(5) Instructions executable by the product with the first 5 digits of the serial No. "04122" or higher

Instruction	Condition (Device)		Processing Time (μs)			
			Q00JCPU	Q00CPU	Q01CPU	
LDE =	Single precision	In conductive status		43.0	35.5	33.0
		In non-conductive status		46.0	38.0	35.5
ANDE =	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	35.5	29.5	26.5
			In non-conductive status	42.0	35.0	32.5
ORE =	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	42.0	35.0	32.5
			In non-conductive status	37.0	31.0	28.5
LDE < >	Single precision	In conductive status		46.0	38.0	35.5
		In non-conductive status		43.5	36.0	33.0
ANDE < >	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	31.5	29.0
			In non-conductive status	39.5	33.0	30.5
ORE < >	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	35.0
			In non-conductive status	34.5	29.0	26.5
LDE >	Single precision	In conductive status		46.0	37.5	35.5
		In non-conductive status		46.0	38.5	35.0
ANDE >	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	32.0	29.0
			In non-conductive status	42.0	35.0	32.5
ORE >	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	34.5
			In non-conductive status	37.0	31.0	29.0
LDE < =	Single precision	In conductive status		45.5	37.5	35.0
		In non-conductive status		46.5	38.5	35.5
ANDE < =	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	31.5	29.0
			In non-conductive status	42.5	35.5	32.5
ORE < =	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	34.5
			In non-conductive status	37.5	31.5	28.5
LDE <	Single precision	In conductive status		45.5	37.5	35.0
		In non-conductive status		46.5	38.5	35.5
ANDE <	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.0	31.5	29.0
			In non-conductive status	42.5	35.5	32.5
ORE <	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	37.5	34.5
			In non-conductive status	37.5	31.5	29.0
LDE > =	Single precision	In conductive status		45.5	38.0	35.5
		In non-conductive status		46.5	38.0	35.0
ANDE > =	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	38.5	32.0	29.0
			In non-conductive status	42.5	35.5	32.5

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.2 Operation Processing Time of Basic Model QCPU

Instruction	Condition (Device)		Processing Time (μs)			
			Q00JCPU	Q00CPU	Q01CPU	
ORE > =	Single precision	When not executed		1.5	1.2	1.0
		When executed	In conductive status	45.0	38.5	34.5
			In non-conductive status	37.5	31.0	28.5
E+ (S) (D) E+P (S) (D)	Single precision	(S) = 0, (D) = 0		29.5	25.0	23.0
		(S) = 2 <sup>127</sup> , (D) = 2 <sup>127</sup>		65.5	60.5	49.5
E+ (S1) (S2) (D) E+P (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0		31.0	27.0	24.0
		(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>		66.5	56.0	51.0
E - (S) (D) E -P (S) (D)	Single precision	(S) = 0, (D) = 0		29.5	25.0	23.0
		(S) = 2 <sup>127</sup> , (D) = 2 <sup>127</sup>		48.5	41.0	37.5
E - (S1) (S2) (D) E -P (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0		31.0	27.0	24.0
		(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>		50.5	42.5	38.5
E* (S1) (S2) (D) E*P (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0		30.0	25.5	23.0
		(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>		65.5	55.0	49.5
E/ (S1) (S2) (D) E/P (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 1		30.0	26.0	23.0
		(S1) = 2 <sup>127</sup> , (S2) = - 2 <sup>126</sup>		69.5	57.5	53.0
INT INTP	Single precision	(S) = 0		21.5	18.5	16.0
		(S) = 32766.5		38.0	32.0	29.5
DINT DINTP	Single precision	(S) = 0		23.0	19.5	17.5
		(S) = 1234567890.3		42.0	35.5	32.0
FLT FLTP	Single precision	(S) = 0		22.5	19.5	17.0
		(S) = 7FFFH		26.5	23.0	20.0
DFLT DFLTP	Single precision	(S) = 0		23.0	20.0	17.5
		(S) = 7FFFFFFFH		26.0	23.5	19.5
ENEG ENEGP	(S) = 0		20.5	17.0	15.5	
	(S) = E - 1.0		31.5	26.0	24.0	
EMOV EMOVP	—		1.5	1.2	1.0	
ESTR ESTRP	—		604.0	686.0	831.0	
EVAL EVALP	Decimal point format all 2-digit specification		138.0	148.0	196.0	
	Exponent format all 6-digit specification		164.0	177.0	214.0	



Instruction	Condition (Device)		Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
SIN SINP	Single precision		204.0	173.0	157.0
COS COSP	Single precision		187.0	158.0	144.0
TAN TANP	Single precision		224.0	190.0	173.0
RAD RADP	Single precision		51.0	43.0	39.0
DEG DEGP	Single precision		51.0	43.0	39.0
SQR SQRP	Single precision		60.0	51.0	46.5
EXP EXPP	Single precision	Ⓢ = - 10	306.0	259.0	235.0
		Ⓢ = 1	306.0	259.0	235.0
LOG LOGP	Single precision	Ⓢ = 1	73.0	61.5	56.0
		Ⓢ = 10	301.0	255.0	232.0
RND RNDP	—		12.5	11.0	10.0
SRND SRNDP	—		13.5	12.0	11.0

A

Instruction Name	Condition/Number of Points Processed		Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
COM *2	With auto refresh of CPU shared memory	Refresh range: 2k words (0.5k words assigned equally to all CPUs)	—	920	880
	Without auto refresh of CPU shared memory	—	—	150	135
FROM	Read from CPU shared memory of host CPU	n3 = 1	—	100	90
		n3 = 320	—	440	420
	Read from CPU shared memory of another CPU	n3 = 1	—	110	105
		n3 = 320	—	305	290
TO	Write to CPU shared memory of host CPU	n3 = 1	—	100	95
		n3 = 320	—	440	425
S.TO	Write to CPU shared memory of host CPU	n4 = 1	—	205	195
		n4 = 320	—	545	525

\*2: If the processing overlaps those of the other CPUs in a multiple CPU system, the processing time increases by a maximum of the following time.

For a system having only the main base unit

$$(\text{Instruction processing time increase}) = 4 \times 0.54 \times (\text{number of points processed}) \times (\text{number of other CPUs}) (\mu\text{s})$$

For a system including extension base units

$$(\text{Instruction processing time increase}) = 4 \times 1.30 \times (\text{number of points processed}) \times (\text{number of other CPUs}) (\mu\text{s})$$

(6) Table of the time to be added when file register, module access device or link direct device is used

Instruction Name	data	Device Specification Location	Processing Time (μs)		
			Q00JCPU	Q00CPU	Q01CPU
File register (ZR)	Bit	Source	—	34	32
		Destination	—	23	22
	Word	Source	—	13	12
		Destination	—	9	8
	Double word	Source	—	14	13
		Destination	—	10	9
Module access device (Un\G□ , U3En\G0 to G511)	Bit	Source	99	82	77
		Destination	167	137	129
	Word	Source	74	61	58
		Destination	72	60	56
	Double word	Source	76	63	59
		Destination	92	75	71
Link direct device (Jn\□ )	Bit	Source	178	147	137
		Destination	303	248	233
	Word	Source	154	126	118
		Destination	153	125	117
	Double word	Source	155	127	119
		Destination	163	133	125

# Appendix 1.3 Operation Processing Time of High Performance Model QCPU/Process CPU/Redundant CPU

The processing time for the individual instructions are shown in the table on the following pages. Operation processing time can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing times rather than as being strictly accurate.

## (1) Sequence instructions

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
LD LDI AND ANI OR ORI	—	0.079	0.034	0.034	0.034
LDP LDF ANDP ANDF ORP ORF	—	0.158	0.068	0.068	0.068
ANB ORB MPS MRD MPP	—	0.079	0.034	0.034	0.034
INV	When not executed	0.079	0.034	0.034	0.034
	When executed				
MEP MEF	When not executed	0.173	0.073	0.073	0.073
	When executed				
EGP EGF	When not executed (OFF→OFF) (ON→ON)	0.158	0.068	0.068	0.068
	When executed (OFF→ON) (ON→OFF)				

A

Instruction	Condition (Device)		Processing Time (μs)					
			Qn	QnH	QnPH	QnPRH		
OUT	When not changed (OFF→OFF) (ON→ON)		0.158	0.068	0.068	0.068		
	When changed (OFF→ON) (ON→OFF)		0.158	0.068	0.068	0.068		
	F	When OFF		2.8	1.2	1.2	1.2	
		When ON	When displayed	162	69.7	69.7	69.7	
			Display completed	126	54	54	54	
	T	When not executed		0.63	0.27	0.27	0.27	
		When executed	After time up	0.63	0.27	0.27	0.27	
			When added	K	0.63	0.27	0.27	0.27
				D	0.63	0.27	0.27	0.27
	C	When not executed		0.63	0.27	0.27	0.27	
		When executed	After time up	0.63	0.27	0.27	0.27	
			When added	K	0.63	0.27	0.27	0.27
D				0.63	0.27	0.27	0.27	
OUTH	When not executed		0.63	0.27	0.27	0.27		
	When executed	After time up	0.63	0.27	0.27	0.27		
		When added	K	0.63	0.27	0.27	0.27	
			D	0.63	0.27	0.27	0.27	
SET	When not executed		0.158	0.068	0.068	0.068		
	When executed	When not changed (ON→ON)	0.158	0.068	0.068	0.068		
		When changed (OFF→ON)	0.158	0.068	0.068	0.068		
	F	When not executed		0.47	0.20	0.20	0.20	
When executed		When displayed	161	69	69	69		
		Display completed	0.47	0.20	0.20	0.20		
RST	When not executed		0.158	0.068	0.068	0.068		
	When executed	When not changed (OFF→OFF)	0.158	0.068	0.068	0.068		
		When changed (ON→OFF)	0.158	0.068	0.068	0.068		
	SM	When not executed		0.158	0.068	0.068	0.068	
		When executed		0.158	0.068	0.068	0.068	
	F	When not executed		0.47	0.20	0.20	0.20	
		When executed	When displayed	90	38	38	38	
			Display completed	0.47	0.20	0.20	0.20	
	T, C	When not executed		0.63	0.27	0.27	0.27	
		When executed		0.63	0.27	0.27	0.27	
	D	When not executed		0.24	0.10	0.10	0.10	
		When executed		0.24	0.10	0.10	0.10	
	Z	When not executed		0.47	0.20	0.20	0.20	
		When executed		4.3	1.9	1.9	1.9	
R	When not executed		0.40	0.17	0.17	0.17		
	When executed		0.40	0.17	0.17	0.17		
PLS PLF	—		1.0	0.44	0.44	0.44		
FF	Y	When not executed	0.47	0.20	0.20	0.20		
		When executed	0.47	0.20	0.20	0.20		
DELTA DELTAP	DY0	When not executed	0.47	0.20	0.20	0.20		
		When executed	5.9	2.6	2.6	2.6		

Instruction	Condition (Device)	Processing Time ( $\mu$ s)			
		Qn	QnH	QnPH	QnPRH
SFT	When not executed	0.47	0.20	0.20	0.20
SFTP	When executed	1.66	0.71	0.71	0.71
MC	—	0.24	0.10	0.10	0.10
MCR	—	0.079	0.034	0.034	0.034
FEND END	Error check performed	380	150	150	500
	No error check performed (• Battery check) (• Fuse blown check) (• I/O module verification)	380	150	150	500
NOP	—	0.079	0.034	0.034	0.034
NOPLF PAGE	—	0.079	0.034	0.034	0.034

A

(2) Basic instructions

The processing time when the instruction is not executed is calculated as follows:

Q02CPU ..... 0.079 × (No. of steps for each instruction + 1) μs

Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU,

Q25PHCPU, Q12PRHCPU, Q25PRHCPU 0.034 × (No. of steps for each instruction + 1) μs

Instruction	Condition (Device)		Processing Time (μs)			
			Qn	QnH	QnPH	QnPRH
LD =	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND =	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR =	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD < >	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND < >	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR < >	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD >	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND >	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR >	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD < =	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND < =	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10

Instruction	Condition (Device)		Processing Time ( $\mu$ s)			
			Qn	QnH	QnPH	QnPRH
OR < =	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD <	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND <	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR <	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LD > =	In conductive status		0.24	0.10	0.10	0.10
	In non-conductive status		0.24	0.10	0.10	0.10
AND > =	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
OR > =	When not executed		0.24	0.10	0.10	0.10
	When executed	In conductive status	0.24	0.10	0.10	0.10
		In non-conductive status	0.24	0.10	0.10	0.10
LDD =	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.39	0.17	0.17	0.17
ANDD =	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.39	0.17	0.17	0.17
ORD =	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD < >	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD < >	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD < >	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD >	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD >	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD >	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
LDD < =	In conductive status		0.55	0.24	0.24	0.24
	In non-conductive status		0.55	0.24	0.24	0.24
ANDD < =	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24
ORD < =	When not executed		0.39	0.17	0.17	0.17
	When executed	In conductive status	0.55	0.24	0.24	0.24
		In non-conductive status	0.55	0.24	0.24	0.24

Instruction	Condition (Device)		Processing Time (μs)				
			Qn	QnH	QnPH	QnPRH	
LDD <	In conductive status		0.55	0.24	0.24	0.24	
	In non-conductive status		0.55	0.24	0.24	0.24	
ANDD <	When not executed		0.39	0.17	0.17	0.17	
	When executed	In conductive status	0.55	0.24	0.24	0.24	
		In non-conductive status	0.55	0.24	0.24	0.24	
ORD <	When not executed		0.39	0.17	0.17	0.17	
	When executed	In conductive status	0.55	0.24	0.24	0.24	
		In non-conductive status	0.55	0.24	0.24	0.24	
LDD > =	In conductive status		0.55	0.24	0.24	0.24	
	In non-conductive status		0.55	0.24	0.24	0.24	
ANDD > =	When not executed		0.39	0.17	0.17	0.17	
	When executed	In conductive status	0.55	0.24	0.24	0.24	
		In non-conductive status	0.55	0.24	0.24	0.24	
ORD > =	When not executed		0.39	0.17	0.17	0.17	
	When executed	In conductive status	0.55	0.24	0.24	0.24	
		In non-conductive status	0.55	0.24	0.24	0.24	
LDE = *1	Single precision	In conductive status		93	40	6.4	6.4
				14.9	6.4		
		In non-conductive status		92	40	6.4	6.4
				14.9	6.4		
	Double precision	In conductive status		93	40	—	—
				14.9	6.4	—	—
		In non-conductive status		92	40	—	—
				14.9	6.4	—	—
ANDE = *1	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
		In non-conductive status	92	40	6.4	6.4	
	14.9		6.4				
	Double precision	When not executed		—	—	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4	—	—
In non-conductive status		92	40	—	—		
	14.9	6.4	—	—			
ORE = *1	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	93	40	6.4	6.4
				14.9	6.4		
		In non-conductive status	92	40	6.4	6.4	
	14.9		6.4				
	Double precision	When not executed		0.55	0.24	—	—
		When executed	In conductive status	93	40	—	—
				14.9	6.4	—	—
In non-conductive status		92	40	—	—		
	14.9	6.4	—	—			

\*1 : The Qn/QnH changes in processing time depending on the serial No. of the CPU module.

Top : The first 5 digits of the serial No. are "05031" or lower

Bottom : The first 5 digits of the serial No. are "05032" or higher

For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.



Instruction	Condition (Device)		Processing Time ( $\mu$ s)					
			Qn	QnH	QnPH	QnPRH		
LDE<> *1	Single precision	In conductive status	92	40	6.4	6.4		
			14.9	6.4				
		In non-conductive status	92	40	6.4	6.4		
			14.9	6.4				
	Double precision	In conductive status	92	40	—	—		
			14.9	6.4				
		In non-conductive status	92	40	—	—		
			14.9	6.4				
ANDE<> *1	Single precision	When not executed		0.55	0.24	0.24	0.24	
		When executed	In conductive status	92	40	6.4	6.4	
				14.9	6.4			
			In non-conductive status	93	40	6.4	6.4	
				14.9	6.4			
		Double precision	When not executed		0.55	0.24	—	—
	When executed		In conductive status	92	40	—	—	
				14.9	6.4			
			In non-conductive status	92	40	—	—	
				14.9	6.4			
	ORE<> *1		Single precision	When not executed		0.55	0.24	0.24
		When executed		In conductive status	93	40	6.4	6.4
14.9					6.4			
In non-conductive status				92	40	6.4	6.4	
				14.9	6.4			
Double precision		When not executed		0.55	0.24	—	—	
		When executed	In conductive status	93	40	—	—	
				14.9	6.4			
			In non-conductive status	92	40	—	—	
				14.9	6.4			
		LDE> *1	Single precision	When not executed		92	40	6.4
In conductive status				14.9	6.4			
In conductive status				92	40	6.4	6.4	
In conductive status				14.9	6.4			
Double precision	In non-conductive status		92	40	—	—		
	In non-conductive status		14.9	6.4				
	In non-conductive status		92	40	—	—		
	In non-conductive status		14.9	6.4				
ANDE> *1	Single precision	When not executed		0.55	0.24	0.24	0.24	
		When executed	In conductive status	92	40	6.4	6.4	
				14.9	6.4			
			In non-conductive status	93	40	6.4	6.4	
				14.9	6.4			
		Double precision	When not executed		0.55	0.24	—	—
	When executed		In conductive status	92	40	—	—	
				14.9	6.4			
			In non-conductive status	92	40	—	—	
				14.9	6.4			

\*1 : The Qn/QnH changes in processing time depending on the serial No. of the CPU module.

Top : The first 5 digits of the serial No. are "05031" or lower

Bottom : The first 5 digits of the serial No. are "05032" or higher

For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Instruction	Condition (Device)		Processing Time (μs)					
			Qn	QnH	QnPH	QnPRH		
ORE> *1	Single precision	When not executed		0.55	0.24	0.24	0.24	
		When executed	In conductive status	93	40	6.4	6.4	
			In non-conductive status	14.9	6.4			
		Double precision	When not executed		0.55	0.24	—	—
	When executed		In conductive status	93	40	—	—	
			In non-conductive status	14.9	6.4			
	Single precision		In conductive status		93	40	6.4	6.4
		In non-conductive status		14.9	6.4			
Double precision		In conductive status		93	40	—	—	
		In non-conductive status		14.9	6.4			
	ANDE<= *1	Single precision	When not executed		0.55	0.24	0.24	0.24
			When executed	In conductive status	92	40	6.4	6.4
In non-conductive status				14.9	6.4			
Double precision			When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—	
			In non-conductive status	14.9	6.4			
		ORE<= *1	Single precision	When not executed		0.55	0.24	0.24
When executed				In conductive status	92	40	6.4	6.4
	In non-conductive status			14.9	6.4			
Double precision	When not executed			0.55	0.24	—	—	
	When executed		In conductive status	92	40	—	—	
			In non-conductive status	14.9	6.4			
	LDE< *1		Single precision	In conductive status		92	40	6.4
In non-conductive status				14.9	6.4			
Double precision		In conductive status		92	40	—	—	
		In non-conductive status		14.9	6.4			

\*1 : The Qn/QnH changes in processing time depending on the serial No. of the CPU module.

Top : The first 5 digits of the serial No. are "05031" or lower

Bottom : The first 5 digits of the serial No. are "05032" or higher

For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Instruction	Condition (Device)		Processing Time ( $\mu$ s)						
			Qn	QnH	QnPH	QnPRH			
ANDE< *1	Single precision	When not executed		0.55	0.24	0.24	0.24		
		When executed	In conductive status	92	40	6.4	6.4		
			In non-conductive status	14.9	6.4				
		Double precision	When not executed		0.55	0.24	—	—	
	When executed		In conductive status	92	40	—	—		
			In non-conductive status	14.9	6.4				
	ORE< *1		Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed		In conductive status	93	40	6.4	6.4	
In non-conductive status				14.9	6.4				
Double precision		When not executed		0.55	0.24	—	—		
		When executed	In conductive status	93	40	—	—		
			In non-conductive status	14.9	6.4				
		LDE>= *1	Single precision	In conductive status		93	40	6.4	6.4
In non-conductive status				14.9	6.4				
Double precision	In conductive status			92	40	—	—		
	In non-conductive status			14.9	6.4				
	ANDE>= *1		Single precision	When not executed		0.55	0.24	0.24	0.24
				When executed	In conductive status	92	40	6.4	6.4
In non-conductive status					14.9	6.4			
Double precision				When not executed		0.55	0.24	—	—
		When executed	In conductive status	92	40	—	—		
			In non-conductive status	14.9	6.4				

\*1 : The Qn/QnH changes in processing time depending on the serial No. of the CPU module.

Top : The first 5 digits of the serial No. are "05031" or lower

Bottom : The first 5 digits of the serial No. are "05032" or higher

For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Instruction	Condition (Device)		Processing Time (μs)				
			Qn	QnH	QnPH	QnPRH	
ORE>= *1	Single precision	When not executed		0.55	0.24	0.24	0.24
		When executed	In conductive status	92	40	6.4	6.4
			In non-conductive status	14.9	6.4		
		Double precision	When not executed		0.55	0.24	—
	When executed		In conductive status	92	40	—	—
			In non-conductive status	14.9	6.4		
	When executed		In conductive status	92	40	—	—
		In non-conductive status	14.9	6.4			
LD\$ =	In conductive status		38	16	16	16	
	In non-conductive status		34	15	15	15	
AND\$ =	When not executed		0.56	0.23	0.23	0.23	
	When executed	In conductive status	39	17	17	17	
		In non-conductive status	32	14	14	14	
OR\$ =	When not executed		0.56	0.24	0.24	0.24	
	When executed	In conductive status	40	17	17	17	
		In non-conductive status	33	14	14	14	
LD\$ < >	In conductive status		32	14	14	14	
	In non-conductive status		40	17	17	17	
AND\$ < >	When not executed		0.56	0.23	0.23	0.23	
	When executed	In conductive status	33	14	14	14	
		In non-conductive status	39	17	17	17	
OR\$ < >	When not executed		0.56	0.24	0.24	0.24	
	When executed	In conductive status	32	14	14	14	
		In non-conductive status	39	17	17	17	
LD\$ >	In conductive status		32	14	14	14	
	In non-conductive status		40	17	17	17	

\*1 : The Qn/QnH changes in processing time depending on the serial No. of the CPU module.

Top : The first 5 digits of the serial No. are "05031" or lower

Bottom : The first 5 digits of the serial No. are "05032" or higher

For the condition to be satisfied when the instruction is not executed, there is no differentiation between the top and bottom.

Instruction	Condition (Device)		Processing Time (µs)			
			Qn	QnH	QnPH	QnPRH
AND\$ >	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	33	14	14	14
		In non-conductive status	39	17	17	17
OR\$ >	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	32	14	14	14
		In non-conductive status	39	17	17	17
LD\$ < =	In conductive status		40	17	17	17
	In non-conductive status		32	14	14	14
AND\$ < =	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	39	17	17	17
		In non-conductive status	32	14	14	14
OR\$ < =	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	40	17	17	17
		In non-conductive status	33	14	14	14
LD\$ <	In conductive status		32	14	14	14
	In non-conductive status		40	17	17	17
AND\$ <	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	32	14	14	14
		In non-conductive status	39	16	16	16
OR\$ <	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	32	14	14	14
		In non-conductive status	39	16	16	16
LD\$ > =	In conductive status		40	17	17	17
	In non-conductive status		32	14	14	14
AND\$ > =	When not executed		0.56	0.23	0.23	0.23
	When executed	In conductive status	39	16	16	16
		In non-conductive status	32	14	14	14
OR\$ > =	When not executed		0.56	0.24	0.24	0.24
	When executed	In conductive status	39	17	17	17
		In non-conductive status	32	14	14	14
BKCMP = S1 S2 D n	n = 1		48	21	21	21
BKCMP = P S1 S2 D n	n = 96		142	61	61	61
BKCMP <> S1 S2 D n	n = 1		48	21	21	21
BKCMP <>P S1 S2 D n	n = 96		150	65	65	65
BKCMP > S1 S2 D n	n = 1		48	21	21	21
BKCMP >P S1 S2 D n	n = 96		142	61	61	61
BKCMP >= S1 S2 D n	n = 1		48	21	21	21
BKCMP >=P S1 S2 D n	n = 96		150	65	65	65
BKCMP < S1 S2 D n	n = 1		48	21	21	21
BKCMP <P S1 S2 D n	n = 96		158	68	68	68
BKCMP <= S1 S2 D n	n = 1		48	21	21	21
BKCMP <=P S1 S2 D n	n = 96		150	65	65	65

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
+ (S) (D) +P (S) (D)	When executed	0.39	0.17	0.17	0.17
+ (S1) (S2) (D) +P (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
- (S) (D) - P (S) (D)	When executed	0.39	0.17	0.17	0.17
- (S1) (S2) (D) - P (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
D+ (S) (D) D+P (S) (D)	When executed	0.71	0.31	0.31	0.31
D+ (S1) (S2) (D) D+P (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
D - (S) (D) D - P (S) (D)	When executed	0.71	0.30	0.30	0.30
D - (S1) (S2) (D) D - P (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
* (S1) (S2) (D) * P (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
/ (S1) (S2) (D) /P (S1) (S2) (D)	—	2.7	1.2	1.2	1.2
D * (S1) (S2) (D) D * P (S1) (S2) (D)	—	7.9	3.4	3.4	3.4
D/ (S1) (S2) (D) D/P (S1) (S2) (D)	—	14	6.1	6.1	6.1
B+ (S) (D) B+P (S) (D)	—	2.2	1.0	1.0	1.0
B+ (S1) (S2) (D) B+P (S1) (S2) (D)	—	5.0	2.2	2.2	2.2
B - (S) (D) B - P (S) (D)	—	2.0	0.9	0.9	0.9
B - (S1) (S2) (D) B - P (S1) (S2) (D)	—	4.9	2.1	2.1	2.1
DB+ (S) (D) DB+P (S) (D)	—	12	5.0	5.0	5.0
DB+ (S1) (S2) (D) DB+P (S1) (S2) (D)	—	12	5.3	5.3	5.3
DB - (S) (D) DB - P (S) (D)	—	11	4.8	4.8	4.8
DB - (S1) (S2) (D) DB - P (S1) (S2) (D)	—	12	5.2	5.2	5.2
B * (S1) (S2) (D) B * P (S1) (S2) (D)	—	3.7	1.6	1.6	1.6

Instruction	Condition (Device)		Processing Time (μs)			
			Qn	QnH	QnPH	QnPRH
B/ $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ B/P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		3.8	1.6	1.6	1.6
DB * $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ DB * P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		24	10	10	10
DB/ $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ DB/P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		27	12	12	12
E+ $\textcircled{S}$ $\textcircled{D}$ E+P $\textcircled{S}$ $\textcircled{D}$	Single precision	$\textcircled{S} = 0, \textcircled{D} = 0$	1.8	0.78	0.78	0.78
		$\textcircled{S} = 2^{127}, \textcircled{D} = 2^{127}$	1.8	0.78	0.78	0.78
	Double precision	$\textcircled{S} = 0, \textcircled{D} = 0$	203	87	—	—
		$\textcircled{S} = 2^{127}, \textcircled{D} = 2^{127}$	203	87	—	—
E+ $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ E+P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	Single precision	$\textcircled{S1} = 0, \textcircled{S2} = 0$	2.4	1.1	1.1	1.1
		$\textcircled{S1} = 2^{127}, \textcircled{S2} = 2^{127}$	2.4	1.1	1.1	1.1
	Double precision	$\textcircled{S1} = 0, \textcircled{S2} = 0$	209	90	—	—
		$\textcircled{S1} = 2^{127}, \textcircled{S2} = 2^{127}$	209	90	—	—
E- $\textcircled{S}$ $\textcircled{D}$ E-P $\textcircled{S}$ $\textcircled{D}$	Single precision	$\textcircled{S} = 0, \textcircled{D} = 0$	1.8	0.78	0.78	0.78
		$\textcircled{S} = 2^{127}, \textcircled{D} = 2^{127}$	1.8	0.78	0.78	0.78
	Double precision	$\textcircled{S} = 0, \textcircled{D} = 0$	202	87	—	—
		$\textcircled{S} = 2^{127}, \textcircled{D} = 2^{127}$	202	87	—	—
E- $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ E-P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	Single precision	$\textcircled{S1} = 0, \textcircled{S2} = 0$	2.4	1.1	1.1	1.1
		$\textcircled{S1} = 2^{127}, \textcircled{S2} = 2^{127}$	2.4	1.1	1.1	1.1
	Double precision	$\textcircled{S1} = 0, \textcircled{S2} = 0$	210	90	—	—
		$\textcircled{S1} = 2^{127}, \textcircled{S2} = 2^{127}$	210	90	—	—
E* $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ E*P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	Single precision	$\textcircled{S1} = 0, \textcircled{S2} = 0$	2.4	1.1	1.1	1.1
		$\textcircled{S1} = 2^{126}, \textcircled{S2} = 2^{127}$	2.4	1.1	1.1	1.1
	Double precision	$\textcircled{S1} = 0, \textcircled{S2} = 0$	222	96	—	—
		$\textcircled{S1} = 2^{126}, \textcircled{S2} = 2^{127}$	222	96	—	—
E/ $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ E/P $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	Single precision	$\textcircled{S1} = 0, \textcircled{S2} = 1$	12	5.2	5.2	5.2
		$\textcircled{S1} = 2^{127}, \textcircled{S2} = -2^{126}$	12	5.2	5.2	5.2
	Double precision	$\textcircled{S1} = 0, \textcircled{S2} = 1$	369	159	—	—
		$\textcircled{S1} = 2^{127}, \textcircled{S2} = -2^{126}$	369	159	—	—

Instruction	Condition (Device)		Processing Time (μs)			
			Qn	QnH	QnPH	QnPRH
\$+ (S) (D) \$+P (S) (D)	—		68	29	29	29
\$+ (S1) (S2) (D) \$+P (S1) (S2) (D)	—		81	35	35	35
INC INCP	—		0.32	0.14	0.14	0.14
DINC DINCP	—		0.47	0.20	0.20	0.20
DEC DECP	—		0.32	0.14	0.14	0.14
DDEC DDECP	—		0.47	0.20	0.20	0.20
BCD BCDP	—		1.1	0.48	0.48	0.48
DBCD DBCDP	—		3.2	1.4	1.4	1.4
BIN BINP	—		1.0	0.44	0.44	0.44
DBIN DBINP	—		1.9	0.82	0.82	0.82
INT INTP	Single precision	(S) = 0	3.2	1.4	1.4	1.4
		(S) = 32766.5	3.2	1.4	1.4	1.4
	Double precision	(S) = 0	22	9.3	—	—
		(S) = 32766.5	22	9.3	—	—
DINT DINTP	Single precision	(S) = 0	2.5	1.1	1.1	1.1
		(S) = 1234567890.3	2.5	1.1	1.1	1.1
	Double precision	(S) = 0	24	10	—	—
		(S) = 1234567890.3	24	10	—	—
FLT FLTP	Single precision	(S) = 0	2.1	0.92	0.92	0.92
		(S) = 7FFFH	2.1	0.92	0.92	0.92
	Double precision	(S) = 0	22	9.6	—	—
		(S) = 7FFFH	22	9.6	—	—
DFLT DFLTP	Single precision	(S) = 0	2.1	0.88	0.88	0.88
		(S) = 7FFFFFFFH	2.1	0.88	0.88	0.88
	Double precision	(S) = 0	26	11	—	—
		(S) = 7FFFFFFFH	26	11	—	—



Instruction	Condition (Device)	Processing Time ( $\mu\text{s}$ )			
		Qn	QnH	QnPH	QnPRH
DBL DBLP	—	4.5	1.9	1.9	1.9
WORD WORDP	—	4.7	2.0	2.0	2.0
GRY GRYP	—	4.7	2.0	2.0	2.0
DGRY DGRYP	—	5.3	2.3	2.3	2.3
GBIN GBINP	—	18	7.7	7.7	7.7
DGBIN DGBINP	—	32	14	14	14
NEG NEGP	—	3.6	1.6	1.6	1.6
DNEG DNEGP	—	4.3	1.8	1.8	1.8
ENEG ENEGP	—	3.9	1.7	1.7	1.7
BKBCD (S) (D) n	n = 1	38	17	17	17
BKBCDP (S) (D) n	n = 96	99	43	43	43
BKBIN (S) (D) n	n = 1	38	17	17	17
BKBINP (S) (D) n	n = 96	99	43	43	43
MOV MOV P	(S) = D0, (D) = D1	0.24	0.10	0.10	0.10
	(S) = D0, (D) = J1 \ W1	—	—	—	—
		140* <sup>1</sup>	60* <sup>1</sup>	60* <sup>1</sup>	60* <sup>1</sup>
DMOV DMOV P	(S) = D0, (D) = D1	0.47	0.20	0.20	0.20
	(S) = D0, (D) = J1 \ W1	—	—	—	—
		147* <sup>1</sup>	64* <sup>1</sup>	64* <sup>1</sup>	64* <sup>1</sup>
EMOV EMOV P	—	0.63	0.27	0.27	0.27
\$MOV \$MOV P	—	40	17	17	17
CML CMLP	—	0.40	0.17	0.17	0.17
DCML DCMLP	—	0.55	0.24	0.24	0.24

\*1 : The upper row indicates the processing time when A38B/A1S38B and the extension base are used.  
The center row indicates the processing time when A38HB/A1S38HB is used.  
The lower row indicates the processing time when Q312B is used.

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
BMOV (S) (D) n	n = 1	17	7.1	7.1	7.1
BMOVP (S) (D) n	n = 96	32	14	14	14
FMOV (S) (D) n	n = 1	6.7	2.9	2.9	2.9
FMOVP (S) (D) n	n = 96	14	6.1	6.1	6.1
XCH XCHP DXCH DXCHP	—	1.3	0.54	0.54	0.54
BXCH (D1) (D2) n	n = 1	31	13	13	13
BXCHP (D1) (D2) n	n = 96	84	36	36	36
SWAP SWAPP	—	3.7	1.6	1.6	1.6
CJ	—	3.2	1.4	1.4	1.4
SCJ	—	3.2	1.4	1.4	1.4
JMP	—	3.2	1.4	1.4	1.4
GOEND	—	0.39	0.34	0.34	0.34
DI	—	0.95	0.41	0.41	0.41
EI	—	1.3	0.54	0.54	0.54
IMASK	—	11	4.6	4.6	4.6
IRET	—	1.6	0.68	0.68	0.68
RFS	n = 1	6.7	4.7	4.7	4.7
RFSP	n = 96	19	13	13	13
UDCNT1	—	15	6.5	6.5	—
UDCNT2	—	16	6.8	6.8	—
TTMR	—	10	4.4	4.4	—
STMR	—	20	7.1	7.1	—
ROTC	—	26	11	11	—
RAMP	—	18	7.7	7.7	—
SPD	—	19	8.3	8.3	—
PLSY	—	10	4.5	4.5	—
PWM	—	9.1	3.9	3.9	—
MTR	—	11	4.9	4.9	—

(3) Application instructions

The processing time when the instruction is not executed is calculated as follows:

Q02CPU .....  $0.079 \times (\text{No. of steps for each instruction} + 1) \mu\text{s}$   
 Q02HCPU, Q06HCPU, Q12HCPU, Q25HCPU, Q02PHCPU, Q06PHCPU, Q12PHCPU,  
 Q25PHCPU, Q12PRHCPU, Q25PRHCPU  $\cdot 0.034 \times (\text{No. of steps for each instruction} + 1) \mu\text{s}$

Instruction	Condition (Device)	Processing Time ( $\mu\text{s}$ )			
		Qn	QnH	QnPH	QnPRH
WAND (S) (D) WANDP (S) (D)	When executed	0.39	0.17	0.17	0.17
WAND (S1) (S2) (D) WANDP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DAND (S) (D) DANDP (S) (D)	When executed	0.71	0.31	0.31	0.31
DAND (S1) (S2) (D) DANDP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKAND (S1) (S2) (D) n	n = 1	36	16	16	16
BKANDP (S1) (S2) (D) n	n = 96	74	32	32	32
WOR (S) (D) WORP (S) (D)	When executed	0.40	0.17	0.17	0.17
WOR (S1) (S2) (D) WORP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DOR (S) (D) DORP (S) (D)	When executed	0.71	0.31	0.31	0.31
DOR (S1) (S2) (D) DORP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKOR (S1) (S2) (D) n	n = 1	36	16	16	16
BKORP (S1) (S2) (D) n	n = 96	74	32	32	32
WXOR (S) (D) WXORP (S) (D)	When executed	0.39	0.17	0.17	0.17
WXOR (S1) (S2) (D) WXORP (S1) (S2) (D)	When executed	0.47	0.20	0.20	0.20
DXOR (S) (D) DXORP (S) (D)	When executed	0.71	0.31	0.31	0.31
DXOR (S1) (S2) (D) DXORP (S1) (S2) (D)	When executed	0.79	0.34	0.34	0.34
BKXOR (S1) (S2) (D) n	n = 1	36	16	16	16
BKXORP (S1) (S2) (D) n	n = 96	74	32	32	32

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.3 Operation Processing Time of High Performance Model QCPU/Process CPU/Redundant CPU

Instruction	Condition (Device)	Processing Time ( $\mu$ s)			
		Qn	QnH	QnPH	QnPRH
WXNR $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ WXNRP $\text{\textcircled{S}}$ $\text{\textcircled{D}}$	When executed	0.40	0.17	0.17	0.17
WXNR $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ WXNRP $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$	When executed	0.47	0.20	0.20	0.20
DNXR $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ DNXRP $\text{\textcircled{S}}$ $\text{\textcircled{D}}$	When executed	0.71	0.31	0.31	0.31
DNXR $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ DNXRP $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$	When executed	0.79	0.34	0.34	0.34
BKNXOR $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n BKNXORP $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1 n = 96	36 74	16 32	16 32	16 32
ROR $\text{\textcircled{D}}$ n RORP $\text{\textcircled{D}}$ n	n = 1 n = 15	2.0 2.0	0.85 0.85	0.85 0.85	0.85 0.85
RCR $\text{\textcircled{D}}$ n RCRP $\text{\textcircled{D}}$ n	n = 1 n = 15	1.6 1.6	0.68 0.68	0.68 0.68	0.68 0.68
ROL $\text{\textcircled{D}}$ n ROLP $\text{\textcircled{D}}$ n	n = 1 n = 15	2.0 2.0	0.85 0.85	0.85 0.85	0.85 0.85
RCL $\text{\textcircled{D}}$ n RCLP $\text{\textcircled{D}}$ n	n = 1 n = 15	1.6 1.6	0.68 0.68	0.68 0.68	0.68 0.68
DROR $\text{\textcircled{D}}$ n DRORP $\text{\textcircled{D}}$ n	n = 1 n = 31	3.9 4.0	1.7 1.7	1.7 1.7	1.7 1.7
DRCR $\text{\textcircled{D}}$ n DRCRP $\text{\textcircled{D}}$ n	n = 1 n = 31	4.3 4.3	1.8 1.9	1.8 1.9	1.8 1.9
DROL $\text{\textcircled{D}}$ n DROLP $\text{\textcircled{D}}$ n	n = 1 n = 31	3.9 4.0	1.7 1.7	1.7 1.7	1.7 1.7
DRCL $\text{\textcircled{D}}$ n DRCLP $\text{\textcircled{D}}$ n	n = 1 n = 31	4.3 4.3	1.8 1.9	1.8 1.9	1.8 1.9
SFR $\text{\textcircled{D}}$ n SFRP $\text{\textcircled{D}}$ n	n = 1 n = 15	1.7 2.0	0.75 0.85	0.75 0.85	0.75 0.85
SFL $\text{\textcircled{D}}$ n SFLP $\text{\textcircled{D}}$ n	n = 1 n = 15	1.7 2.0	0.75 0.85	0.75 0.85	0.75 0.85
BSFLR $\text{\textcircled{D}}$ n BSFLRP $\text{\textcircled{D}}$ n	n = 1 n = 96	20 24	8.6 10	8.6 10	8.6 10
BSFL $\text{\textcircled{D}}$ n BSFLP $\text{\textcircled{D}}$ n	n = 1 n = 96	20 23	8.5 10	8.5 10	8.5 10
DSFR $\text{\textcircled{D}}$ n DSFRP $\text{\textcircled{D}}$ n	n = 1 n = 96	1.3 25	0.58 11	0.58 11	0.58 11
DSFL $\text{\textcircled{D}}$ n DSFLP $\text{\textcircled{D}}$ n	n = 1 n = 96	1.3 26	0.58 11	0.58 11	0.58 11

Instruction	Condition (Device)		Processing Time ( $\mu$ s)			
			Qn	QnH	QnPH	QnPRH
BSET $\textcircled{D}$ n	n = 1		7.6	3.3	3.3	3.3
BSETP $\textcircled{D}$ n	n = 15		7.6	3.3	3.3	3.3
BRST $\textcircled{D}$ n	n = 1		7.6	3.3	3.3	3.3
BRSTP $\textcircled{D}$ n	n = 15		7.6	3.3	3.3	3.3
TEST $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		8.2	3.5	3.5	3.5
TESTP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$						
DTEST $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$	—		9.2	3.9	3.9	3.9
DTESTP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$						
BKRST $\textcircled{S}$ n	n = 1		18	7.8	7.8	7.8
BKRSTP $\textcircled{S}$ n	n = 96		19	8.2	8.2	8.2
SER $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	All match	22	9.6	9.6	9.6
		None match	21	8.9	8.9	8.9
SERP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 96	All match	115	49	49	49
		None match	133	57	57	57
DSER $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	All match	23	9.9	9.9	9.9
		None match	23	9.7	9.7	9.7
DSERP $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 96	All match	142	61	61	61
		None match	132	57	57	57
SUM	$\textcircled{S} = 0$		3.9	1.7	1.7	1.7
SUMP	$\textcircled{S} = \text{FFFFH}$					
DSUM	$\textcircled{S} = 0$		4.7	2.0	2.0	2.0
DSUMP	$\textcircled{S} = \text{FFFFFFFFH}$		12	5.0	5.0	5.0
DECO $\textcircled{S}$ $\textcircled{D}$ n	n = 2		20	8.6	8.6	8.6
DECOP $\textcircled{S}$ $\textcircled{D}$ n	n = 8		27	12	12	12
ENCO $\textcircled{S}$ $\textcircled{D}$ n	n = 2	M1 = ON	21	9.1	9.1	9.1
		M4 = ON	21	9.1	9.1	9.1
ENCOP $\textcircled{S}$ $\textcircled{D}$ n	n = 8	M1 = ON	28	12	12	12
		M256 = ON	26	11	11	11
SEG	—		1.3	0.54	0.54	0.54
SEGP						
DIS $\textcircled{S}$ $\textcircled{D}$ n	n = 1		18	7.7	7.7	7.7
DISP $\textcircled{S}$ $\textcircled{D}$ n	n = 4		19	8.3	8.3	8.3
UNI $\textcircled{S}$ $\textcircled{D}$ n	n = 1		21	8.9	8.9	8.9
UNIP $\textcircled{S}$ $\textcircled{D}$ n	n = 4		23	9.7	9.7	9.7
NDIS $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$	—		41	18	18	18
NDISP $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$						
NUNI $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$	—		42	18	18	18
NUNIP $\textcircled{S1}$ $\textcircled{D}$ $\textcircled{S2}$						
WTOB $\textcircled{S}$ $\textcircled{D}$ n	n = 1		47	20	20	20
WTOBP $\textcircled{S}$ $\textcircled{D}$ n	n = 96		99	43	43	43
BTOW $\textcircled{S}$ $\textcircled{D}$ n	n = 1		45	19	19	19
BTOWP $\textcircled{S}$ $\textcircled{D}$ n	n = 96		89	38	38	38

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
MAX (S) (D) n	n = 1	17	7.1	7.1	7.1
MAXP (S) (D) n	n = 96	136	59	59	59
MIN (S) (D) n	n = 1	17	7.1	7.1	7.1
MINP (S) (D) n	n = 96	159	69	69	69
DMAX (S) (D) n	n = 1	27	12	12	12
DMAXP (S) (D) n	n = 96	181	78	78	78
DMIN (S) (D) n	n = 1	27	12	12	12
DMINP (S) (D) n	n = 96	112	48	48	48
SORT (S1) n (S2) (D1) (D2)	n = 1	16	7.1	7.1	7.1
	n = 96	14	6.2	6.2	6.2
DSORT (S1) n (S2) (D1) (D2)	n = 1	17	7.1	7.1	7.1
	n = 96	16	6.8	6.8	6.8
WSUM (S) (D) n	n = 1	16.4	7.1	7.1	7.1
WSUMP (S) (D) n	n = 96	68.4	29.5	29.5	29.5
DWSUM (S) (D) n	n = 1	18.9	8.2	8.2	8.2
DWSUMP (S) (D) n	n = 96	130.4	56.1	56.1	56.1
FOR n	n = 0	2.3	1.0	1.0	1.0
NEXT	—	3.3	1.4	1.4	1.4
BREAK	—	11	4.6	4.6	4.6
BREAKP					
CALL Pn	Internal file pointer	2.1	0.88	0.88	0.88
CALLP Pn	Common pointer	33	14	14	14
CALL Pn (S1) to (S5)	—	135	58	58	58
CALLP Pn (S1) to (S5)					
RET	Return to original program	2.9	1.3	1.3	1.3
	Return to other program	20	8.5	8.5	8.5
FCALL Pn	Internal file pointer	3.6	1.6	1.6	1.6
FCALLP Pn	Common pointer	20	8.7	8.7	8.7
FCALL Pn (S1) to (S5)	—	134	57	57	57
FCALLP Pn (S1) to (S5)					
ECALL * Pn	—	77	33	33	33
ECALLP * Pn *: Program name					
ECALL * Pn (S1) to (S5)	—	162	70	70	70
ECALLP * Pn (S1) to (S5) *: Program name					
EFCALL * Pn	—	78	34	34	34
EFCALLP * Pn *: Program name					
EFCALL * Pn (S1) to (S5)	—	200	86	86	86
EFCALLP * Pn (S1) to (S5) *: Program name					

\*1: Indicates extension of scan time to completion of instruction.

Instruction	Condition (Device)		Processing Time (μs)			
			Qn	QnH	QnPH	QnPRH
COM	—		55	16	16	16
IX	—		12	5.2	5.2	5.2
IXEND	—		4.7	2.0	2.0	2.0
IXDEV + IXSET	Number of contacts 1		48	21	21	21
	Number of contacts 14		93	40	40	40
FIFW	Number of data points 0		11	4.5	4.5	4.5
FIFWP	Number of data points 96		11	4.5	4.5	4.5
FIFR	Number of data points 1		13	5.6	5.6	5.6
FIFRP	Number of data points 96		32	14	14	14
FPOP	Number of data points 1		16	7.0	7.0	7.0
FPOPP	Number of data points 96		16	7.0	7.0	7.0
FINS	Number of data points 0		20	8.4	8.4	8.4
FINSP	Number of data points 96		36	15	15	15
FDEL	Number of data points 1		19	7.5	7.5	7.5
FDELP	Number of data points 96		39	15	15	15
FROM n1 n2 $\text{\textcircled{D}}$ n3	n3 = 1		—	—	—	—
			—	—	—	—
			47	22	22	22
FROMP n1 n2 $\text{\textcircled{D}}$ n3 *1	n3 = 1000		—	—	—	—
			—	—	—	—
			476	437	437	437
DFRO n1 n2 $\text{\textcircled{D}}$ n3	n3 = 1		—	—	—	—
			—	—	—	—
			51	24	24	24
DFROP n1 n2 $\text{\textcircled{D}}$ n3 *1	n3 = 500		—	—	—	—
			—	—	—	—
			478	437	437	437
TO n1 n2 $\text{\textcircled{D}}$ n3	n3 = 1		—	—	—	—
			—	—	—	—
			48	20	20	20
TOP n1 n2 $\text{\textcircled{D}}$ n3 *1	n3 = 1000		—	—	—	—
			—	—	—	—
			479	412	412	412
DTO n1 n2 $\text{\textcircled{D}}$ n3	n3 = 1		—	—	—	—
			—	—	—	—
			50	23	23	23
DTOP n1 n2 $\text{\textcircled{D}}$ n3 *1	n3 = 500		—	—	—	—
			—	—	—	—
			457	416	416	416
PR	SM701ON	Variable 1 character	33	11	11	—
		Variable 32 character	48	18	18	—
	SM701OFF		21	7.8	7.8	—
PRC	—		181	16	16	—
LED	When displayed		—	—	—	—
	Display completed		—	—	—	—

\*1 : The upper row indicates the processing time when A38B/A1S38B and the extension base are used.  
The center row indicates the processing time when A38HB/A1S38HB is used.  
The bottom row indicates the processing times taken when the Q312B is used to execute the instruction for the QJ71C24 in slot 0.  
The FROM/TO instruction differs in processing time according to the number of slots and the loaded modules.  
(The QnCPU/QnHCPU also differs in processing time according to the extension base type.)

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
LEDC	When displayed	—	—	—	—
	Display completed	—	—	—	—
LEDR	No display → no display	0.40	0.17	0.17	0.17
	LED instruction execution → no display	103	44	44	44
CHKST	—	5.8	2.5	2.5	2.5
CHK	1 contact no error	24	10	10	10
	150 contact no error	1676	721	721	721
	1 contact error	88	38	38	38
CHKCIR	10 steps	5.8	2.5	2.5	2.5
SLT	All internal devices	—	—	—	—
	File register 8k points	—	—	—	—
	SLT execution completion	—	—	—	—
SLTR	—	—	—	—	
STRA	Start	—	—	—	—
	STRA execution completion	—	—	—	—
STRAR	—	—	—	—	
PTRA	—	—	—	—	
PTRAR	—	—	—	—	
PTRAEXE PTRAEPEP	When operating	—	—	—	—
	Trace in progress	—	—	—	—
BINDA BINDAP	Ⓢ = 1	15	6.7	6.7	6.7
	Ⓢ = - 32768	24	10	10	10
DBINDA DBINDAP	Ⓢ = 1	43	18	18	18
	Ⓢ = - 2147483648	86	37	37	37
BINHA BINHAP	Ⓢ = 1	18	7.7	7.7	7.7
	Ⓢ = FFFFH	19	8.2	8.2	8.2
DBINHA DBINHAP	Ⓢ = 1	23	10	10	10
	Ⓢ = FFFFFFFFH	24	10	10	10
BCDDA BCDDAP	Ⓢ = 1	23	9.8	9.8	9.8
	Ⓢ = 9999	21	8.9	8.9	8.9
DBCDDA DBCDDAP	Ⓢ = 1	22	9.5	9.5	9.5
	Ⓢ = 99999999	29	13	13	13
DABIN DABINP	Ⓢ = 1	57	25	25	25
	Ⓢ = - 32768	58	25	25	25
DDABIN DDABINP	Ⓢ = 1	92	40	40	40
	Ⓢ = - 2147483648	106	46	46	46
HABIN HABINP	Ⓢ = 1	13	5.8	5.8	5.8
	Ⓢ = FFFFH	15	6.4	6.4	6.4
DHABIN DHABINP	Ⓢ = 1	22	9.5	9.5	9.5
	Ⓢ = FFFFFFFFH	25	11	11	11



Instruction	Condition (Device)	Processing Time (µs)				
		Qn	QnH	QnPH	QnPRH	
DABCD DABCDP	Ⓢ = 1	16	6.9	6.9	6.9	
	Ⓢ = 9999	17	7.2	7.2	7.2	
DDABCD DDABCDP	Ⓢ = 1	25	11	11	11	
	Ⓢ = 99999999	29	13	13	13	
COMRD COMRDP	—	40	17	17	17	
LEN LENP	1 character	18	8.0	8.0	8.0	
	96 characters	86	37	37	37	
STR STRP	—	53	23	23	23	
DSTR DSTRP	—	123	53	53	53	
VAL VALP	—	95	41	41	41	
DVAL DVALP	—	166	72	72	72	
ESTR ESTRP	—	564	243	243	243	
EVAL EVALP	Decimal point format all 2-digit specification	100	43	43	43	
	Exponent format all 6-digit specification	127	55	55	55	
ASC Ⓢ Ⓣ n	n = 1	64	28	28	28	
ASCP Ⓢ Ⓣ n	n = 96	289	125	125	125	
HEX Ⓢ Ⓣ n	n = 1	60	26	26	26	
HEXP Ⓢ Ⓣ n	n = 96	343	148	148	148	
RIGHT Ⓢ Ⓣ n	n = 1	49	21	21	21	
RIGHTP Ⓢ Ⓣ n	n = 96	131	56	56	56	
LEFT Ⓢ Ⓣ n	n = 1	50	21	21	21	
LEFTP Ⓢ Ⓣ n	n = 96	131	56	56	56	
MIDR MIDRP	—	53	23	23	23	
MIDW MIDWP	—	128	55	55	55	
INSTR INSTRP	No match		58	25	25	25
	Match	Head	55	24	24	24
		End	58	25	25	25

Instruction	Condition (Device)		Processing Time ( $\mu$ s)			
			Qn	QnH	QnPH	QnPRH
EMOD EMODP	—		527	227	227	227
EREXP EREXPP	—		1656	713	713	713
SIN SINP	Single precision		115	50	50	50
	Double precision		1945	837	—	—
COS COSP	Single precision		122	53	53	53
	Double precision		2618	1127	—	—
TAN TANP	Single precision		123	53	53	53
	Double precision		2618	1127	—	—
ASIN ASINP	Single precision		111	48	48	48
	Double precision		2491	1072	—	—
ACOS ACOSP	Single precision		115	49	49	49
	Double precision		2367	1019	—	—
ATAN ATANP	Single precision		157	68	68	68
	Double precision		3140	1352	—	—
RAD RADP	Single precision		17	7.2	7.2	7.2
	Double precision		24	10	—	—
DEG DEGP	Single precision		17	7.2	7.2	7.2
	Double precision		23	9.9	—	—
SQR SQRP	Single precision		28	12	12	12
	Double precision		1812	780	—	—
EXP EXPP	Single precision	$\textcircled{S} = -10$	129	56	56	56
		$\textcircled{S} = 1$				
	Double precision	$\textcircled{S} = -10$	2386	1026	—	—
		$\textcircled{S} = 1$				
LOG LOGP	Single precision	$\textcircled{S} = 1$	113	49	49	49
		$\textcircled{S} = 10$				
	Double precision	$\textcircled{S} = 1$	2146	924	—	—
		$\textcircled{S} = 10$				
RND RNDP	—		3.9	1.7	1.7	1.7
SRND SRNDP	—		3.5	1.5	1.5	1.5

Instruction	Condition (Device)	Processing Time ( $\mu$ s)			
		Qn	QnH	QnPH	QnPRH
BSQR BSQRP	$\textcircled{S} = 0$	6.2	2.7	2.7	2.7
	$\textcircled{S} = 9999$	38	16	16	16
BDSQR BDSQRP	$\textcircled{S} = 0$	6.2	2.7	2.7	2.7
	$\textcircled{S} = 99999999$	38	16	16	16
BSIN BSINP	—	12	5.1	5.1	5.1
BCOS BCOSP	—	12	5.2	5.2	5.2
BTAN BTANP	—	12	5.2	5.2	5.2
BASIN BASINP	—	20	8.7	8.7	8.7
BACOS BACOSP	—	21	9.0	9.0	9.0
BATAN BATANP	—	22	9.6	9.6	9.6
LIMIT LIMITP	—	10	4.3	4.3	4.3
DLIMIT DLIMITP	—	11	4.7	4.7	4.7
BAND BANDP	—	9.8	4.2	4.2	4.2
DBAND DBANDP	—	11	4.9	4.9	4.9
ZONE ZONEP	—	9.1	3.9	3.9	3.9
DZONE DZONEP	—	11	4.6	4.6	4.6
RSET RSETP	—	6.8	2.9	2.9	2.9
QDRSET QDRSETP	—	205	88	88	88
QCDSET QCDSETP	—	147	63	63	63
DATERD DATERDP	—	13	5.5	5.5	5.5
DATEWR DATEWRP	—	15	6.4	6.4	6.4

Instruction	Condition (Device)	Processing Time ( $\mu$ s)			
		Qn	QnH	QnPH	QnPRH
DATE+	No digit increase	13	5.4	5.4	5.4
DATE+P	Digit increase	13	5.4	5.4	5.4
DATE -	No digit increase	12	5.2	5.2	5.2
DATE - P	Digit increase	12	5.2	5.2	5.2
SECOND	—	10	4.5	4.5	4.5
SECONDP	—	10	4.5	4.5	4.5
HOUR	—	12	5.2	5.2	5.2
HOURP	—	12	5.2	5.2	5.2
MSG	1 character	3.0	1.3	1.3	1.3
	32 characters	3.0	1.3	1.3	1.3
PKEY	Initial time	20	8.6	8.6	8.6
	No reception	19	8.2	8.2	8.2
PSTOP	—	79	34	34	34
PSTOPP	—	79	34	34	34
POFF	—	79	34	34	34
POFFP	—	79	34	34	34
PSCAN	—	75	32	32	32
PSCANP	—	75	32	32	32
FLOW	—	80	34	34	—
FLOWP	—	80	34	34	—
WDT	—	5.9	2.6	2.6	2.6
WDTP	—	5.9	2.6	2.6	2.6
DUTY	—	9.3	4.0	4.0	4.0
ZRRDB	—	7.9	3.4	3.4	3.4
ZRRDBP	—	7.9	3.4	3.4	3.4
ZRWRB	—	9.4	4.0	4.0	4.0
ZRWRBP	—	9.4	4.0	4.0	4.0
ADRSET	—	4.9	2.1	2.1	2.1
ADRSETP	—	4.9	2.1	2.1	2.1
KEY	—	17	7.3	7.3	—
ZPUSH	—	11	4.7	4.7	4.7
ZPUSHP	—	11	4.7	4.7	4.7
ZPOP	—	5.1	2.2	2.2	2.2
ZPOPP	—	5.1	2.2	2.2	2.2
EROMWR	—	—	—	—	—
EROMWRP	—	—	—	—	—

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
ZCOM	—	691	289	289	289
READ	—	—	—	—	—
SREAD	—	—	—	—	—
WRITE	—	—	—	—	—
SWRITE	—	—	—	—	—
SEND	—	—	—	—	—
RECV	—	—	—	—	—
REQ	—	—	—	—	—
ZNFR	—	—	—	—	—
ZNTO	—	—	—	—	—
ZNRD	MELSECNET/10	—	—	—	—
	MELSECNET (II)	—	—	—	—
ZNWR	MELSECNET/10	—	—	—	—
	MELSECNET (II)	—	—	—	—
RFRP	—	—	—	—	—
RTOP	—	—	—	—	—

## (4) Processing time for QCPU instructions (QCPU instructions only)

## (a) Instructions available from function version A

Instruction	Condition (Device)		Processing Time (μs)			
			Qn	QnH	QnPH	QnPRH
UNIRD	—		79	34	34	34
TRACE	Start		176	76	76	76
	STRA execution completion		6.3	2.7	2.7	2.7
TRACER	—		19	8.2	8.2	8.2
SP.FWRITE	—		84	36	36	36
SP.FREAD	—		82	35	35	35
PLOADP	—		58	25	25	—
PUNLOADP	—		272	117	117	—
PSWAPP	—		308	133	133	—
RBMOV	When standard RAM is used	1 point	45.5	20	20	20
		1000 points	215	91	91	91
	When SRAM card is used	1 point	49.5	22	22	22
		1000 points	540	305	305	305

(b) Instructions available from function version B

Instruction	Condition/Number of Points Processed		Processing Time (μs)				
			Qn	QnH	QnPH	QnPRH	
COM *1	With auto refresh of CPU shared memory	Refresh range: 2k words (0.5k words assigned equally to all CPUs)	720	660	660	—	
		Refresh range: 4k words (1k words assigned equally to all CPUs)	860	730	730	—	
	Without auto refresh of CPU shared memory	—	43	20	20	20	
FROM *1	Reading from CPU shared memory of another CPU	n3 = 1	59	29	29	—	
		n3 = 1000	530	500	500	—	
	Reading buffer memory of intelligent function module*2	n3 = 1	Main base unit	51	24	24	—
			Extension base unit	54	27	27	—
		n3 = 1000	Main base unit	540	480	480	—
Extension base unit			1100	1050	1050	—	
S.TO	Writing to CPU shared memory of host CPU	n3 = 1 ("TO" instruction) n4 = 1 ("S.TO instruction")	74	33	33	—	
		n2 = 256	126	54	54	—	
S (P).DATERD *3	Reading data of the expansion clock	—	25	11	11	11	
S (P).DATE+ *3	Expansion clock data addition operation	—	38	17	17	17	
S (P).DATE- *3	Expansion clock data subtraction operation	—	38	17	17	17	

\*1 : If the processing overlaps those of the other CPUs in a multiple CPU system, the processing time increases by a maximum of the following time.

For system having only the main base unit

$$(\text{Instruction processing time increase}) = 0.54 \times (\text{number of points processed}) \times (\text{number of other CPUs}) (\mu\text{s})$$

For system including extension base units

$$(\text{Instruction processing time increase}) = 1.30 \times (\text{number of points processed}) \times (\text{number of other CPUs}) (\mu\text{s})$$

\*2 : In a multiple CPU system, the instruction processing time for the intelligent function module under control of the host CPU is equal to that for the intelligent function module under control of another CPU.

\*3 : Products with the first 5 digits of the serial No. "07032" or higher are applicable.

(5) Redundant system instructions (for redundant CPU)

Instruction	Condition (Device)	Processing Time (μs)			
		Qn	QnH	QnPH	QnPRH
SP.CONTSW	—	—	—	—	9.6

(6) Table of the time to be added when file register, module access device or link direct device is used

Instruction		data	Device Specification Location	Processing Time (μs)			
				Qn	QnH	QnPH	QnPRH
File register (ZR)	When standard RAM is used	Bit	Source	5.56	2.40	2.40	2.40
			Destination	4.44	1.91	1.91	1.91
		Word	Source	2.60	1.12	1.12	1.12
			Destination	3.76	1.62	1.62	1.62
		Double word	Source	2.83	1.22	1.22	1.22
			Destination	4.00	1.72	1.72	1.72
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	5.22	2.25	2.25	2.25
			Destination	4.09	1.76	1.76	1.76
		Word	Source	2.25	0.97	0.97	0.97
			Destination	3.42	1.47	1.47	1.47
		Double word	Source	2.49	1.07	1.07	1.07
			Destination	3.65	1.57	1.57	1.57
Module access device (Un\G□, U3En\G0 to G4095)	Bit	Source	35.56	15.31	15.31	15.31	
		Destination	65.08	28.01	28.01	28.01	
	Word	Source	32.76	14.10	14.10	14.10	
		Destination	28.84	12.41	12.41	12.41	
	Double word	Source	32.99	14.20	14.20	14.20	
		Destination	29.07	12.51	12.51	12.51	
Link direct device (Jn\□)	Bit	Source	75.67	32.57	32.57	32.57	
		Destination	138.65	59.67	59.67	59.67	
	Word	Source	72.73	31.30	31.30	31.30	
		Destination	137.32	59.10	59.10	59.10	
	Double word	Source	72.96	31.40	31.40	31.40	
		Destination	137.55	59.20	59.20	59.20	

A

# Appendix 1.4 Operation Processing Time of Universal Model QCPU

The processing time for the individual instructions are shown in the table on the following pages. Operation processing times can vary substantially depending on the nature of the sources and destinations of the instructions, and the values contained in the following tables should therefore be taken as a set of general guidelines to processing time rather than as being strictly accurate.

## Appendix 1.4.1 Subset instruction processing time

The following describes the subset instruction processing time.

### POINT

1. The subset instruction processing time table shown in (1) applies when the device used in an instruction satisfies either of the conditions (a) and (b).
2. Since the processing time of each instruction is not constant due to the cache function in the Universal model QCPU, the minimum value and the maximum value are described.

#### (1) Subset instruction processing time table

(a) When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU.

Category	Instruction	Condition (Device)	Processing Time (μs)								
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Sequence instruction	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	When executed	0.120		0.080		0.060		0.040		
	LDPI LDFI	When executed	0.360		0.240		0.180		0.120		
	ANDPI ANDFI ORPI ORFI	When executed	0.480		0.320		0.240		0.160		
	OUT	When not changed	0.120		0.080		0.060		0.040		
		When changed	0.120		0.080		0.060		0.040		
	SET RST	When not executed	0.120		0.080		0.060		0.040		
		When executed	When not changed	0.120		0.080		0.060		0.040	
			When changed	0.120		0.080		0.060		0.040	



A

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD=	In conductive status	0.360	0.240	0.180	0.120	In non-conductive status			
		In non-conductive status								
	AND=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	OR=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	LD<>	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status		
		In non-conductive status								
	AND<>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	OR<>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	LD>	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status		
		In non-conductive status								
	AND>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	OR>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	LD<=	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status		
		In non-conductive status								
	AND<=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
In non-conductive status										
OR<=	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
LD<	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status			
	In non-conductive status									
AND<	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
OR<	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
LD>=	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status			
	In non-conductive status									
AND>=	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
OR>=	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDD=	In conductive status	0.360	0.240	0.180	0.120	In non-conductive status			
		In non-conductive status								
	ANDD=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	ORD=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	LDD<>	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status		
		In non-conductive status								
	ANDD<>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	ORD<>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	LDD>	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status		
		In non-conductive status								
	ANDD>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	ORD>	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	LDD<=	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status		
		In non-conductive status								
	ANDD<=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
	ORD<=	When not executed		0.360	0.240	0.180	0.120	When executed		
		In conductive status								
		In non-conductive status								
LDD<	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status			
	In non-conductive status									
ANDD<	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
ORD<	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
LDD>=	In conductive status		0.360	0.240	0.180	0.120	In non-conductive status			
	In non-conductive status									
ANDD>=	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									
ORD>=	When not executed		0.360	0.240	0.180	0.120	When executed			
	In conductive status									
	In non-conductive status									

Category	Instruction	Condition (Device)	Processing Time ( $\mu$ s)								
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Basic instruction	+ (S) (D)	When executed	0.360		0.240		0.180		0.120		
	+ (S1) (S2) (D)	When executed	0.480		0.320		0.240		0.160		
	- (S) (D)	When executed	0.360		0.240		0.180		0.120		
	- (S1) (S2) (D)	When executed	0.480		0.320		0.240		0.160		
	D + (S) (D)	When executed	0.360		0.240		0.180		0.120		
	D + (S1) (S2) (D)	When executed	0.480		0.320		0.240		0.160		
	D - (S) (D)	When executed	0.360		0.240		0.180		0.120		
	D - (S1) (S2) (D)	When executed	0.480		0.320		0.240		0.160		
	* (S1) (S2) (D)	When executed	0.420		0.300		0.240		0.180		
	/ (S1) (S2) (D)	When executed	0.520		0.400		0.340		0.280		
	D * (S1) (S2) (D)	When executed	0.500		0.380		0.320		0.260		
	D/ (S1) (S2) (D)	When executed	0.640		0.520		0.460		0.400		
	B + (S) (D)	When executed	3.100	12.300	3.100	12.300	3.100	12.300	3.300	8.300	
	B + (S1) (S2) (D)	When executed	5.900	13.500	5.900	13.500	5.900	13.500	4.600	6.200	
	B - (S) (D)	When executed	3.150	12.300	3.150	12.300	3.150	12.300	3.300	9.000	
	B - (S1) (S2) (D)	When executed	5.950	13.600	5.950	13.600	5.950	13.600	4.600	8.200	
	B * (S1) (S2) (D)	When executed	3.700	12.100	3.700	12.100	3.700	12.100	4.000	8.200	
	B/ (S1) (S2) (D)	When executed	4.000	14.000	4.000	14.000	4.000	14.000	4.200	12.400	
	E + (S) (D)	Single precision	(S) = 0, (D) = 0	0.420		0.300		0.240		0.180	
			(S) = $2^{127}$ , (D) = $2^{127}$	0.420		0.300		0.240		0.180	
	E + (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0	0.540		0.380		0.300		0.220	
			(S1) = $2^{127}$ , (S2) = $2^{127}$	0.540		0.380		0.300		0.220	
	E - (S) (D)	Single precision	(S) = 0, (D) = 0	0.420		0.300		0.240		0.180	
			(S) = $2^{127}$ , (D) = $2^{127}$	0.420		0.300		0.240		0.180	
	E - (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0	0.540		0.380		0.300		0.220	
			(S1) = $2^{127}$ , (S2) = $2^{127}$	0.540		0.380		0.300		0.220	
	E * (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0	0.420		0.300		0.240		0.180	
			(S1) = $2^{127}$ , (S2) = $2^{127}$	0.420		0.300		0.240		0.180	
	E/ (S1) (S2) (D)	Single precision	(S1) = $2^{127}$ , (S2) = $2^{127}$	4.900	18.900	4.900	18.900	4.900	18.900	5.100	14.100
	INC	When executed	0.240		0.160		0.120		0.080		
	DINC	When executed	0.240		0.160		0.120		0.080		
	DEC	When executed	0.240		0.160		0.120		0.080		
DDEC	When executed	0.240		0.160		0.120		0.080			
BCD	When executed	0.320		0.240		0.200		0.160			
DBCD	When executed	0.400		0.320		0.280		0.240			
BIN	When executed	0.260		0.180		0.140		0.100			
DBIN	When executed	0.260		0.180		0.140		0.100			

Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q00UCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	FLT	Single precision	Ⓢ = 0	0.300	0.220	0.180	0.140				
			Ⓢ = 7FFF <sub>H</sub>	0.300	0.220	0.180	0.140				
	DFLT	Single precision	Ⓢ = 0	0.300	0.220	0.180	0.140				
			Ⓢ = 7FFFFFFF <sub>H</sub>	0.300	0.220	0.180	0.140				
	INT	Single precision	Ⓢ = 0	0.300	0.220	0.180	0.140				
			Ⓢ = 32766.5	0.300	0.220	0.180	0.140				
	DINT	Single precision	Ⓢ = 0	0.300	0.220	0.180	0.140				
			Ⓢ = 1234567890.3	0.300	0.220	0.180	0.140				
	MOV		—	0.240	0.160	0.120	0.080				
	DMOV		—	0.240	0.160	0.120	0.080				
	EMOV		—	0.240	0.160	0.120	0.080				
	CML		—	0.240	0.160	0.120	0.080				
	DCML		—	0.240	0.160	0.120	0.080				
	BMOV	SM237= ON	n=1	4.200	4.600	4.200	4.600	4.200	4.600	4.100	4.500
			n=96	4.850	5.150	4.850	5.150	4.850	5.150	4.700	5.100
		SM237= OFF	n=1	6.800	11.300	6.800	11.300	6.800	11.300	6.300	8.900
			n=96	7.450	11.900	7.450	11.900	7.450	11.900	5.900	9.500
	FMOV	SM=237 =ON	n=1	4.100	4.600	4.100	4.600	4.100	4.600	4.100	4.600
			n=96	4.800	5.200	4.800	5.200	4.800	5.200	4.800	5.200
		SM237= OFF	n=1	4.600	8.250	4.600	8.250	4.600	8.250	4.600	7.900
n=96			6.150	10.600	6.150	10.600	6.150	10.600	5.300	8.500	
XCH		—	2.250	8.100	2.250	8.100	2.250	8.100	2.500	6.000	
DXCH		—	2.400	8.200	2.400	8.200	2.400	8.200	2.800	7.900	
DFMOV	SM237= ON	n=1	2.700	2.800	2.700	2.800	2.700	2.800	2.350	2.450	
		n=96	6.500	6.800	6.500	6.800	6.500	6.800	5.950	6.000	
	SM237= OFF	n=1	4.000	8.150	4.000	8.150	4.000	8.150	3.000	6.950	
		n=96	8.000	12.200	8.000	12.200	8.000	12.200	6.600	10.600	
CJ		—	3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100	
SCJ		—	3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100	
JMP		—	3.500	10.100	3.500	10.100	3.500	10.100	1.900	10.100	
Application instruction	WAND Ⓢ Ⓣ Ⓞ	When executed		0.360	0.240	0.180	0.120				
	WAND Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓞ	When executed		0.480	0.320	0.240	0.160				
	DAND Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	DAND Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				
	WOR Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	WOR Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				
	DOR Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	DOR Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				
	WXOR Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	WXOR Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				
	DXOR Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	DXOR Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				
	WXNR Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	WXNR Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				
	DXNR Ⓢ Ⓣ	When executed		0.360	0.240	0.180	0.120				
	DXNR Ⓢ <sub>1</sub> Ⓢ <sub>2</sub> Ⓣ	When executed		0.480	0.320	0.240	0.160				

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ROR $\text{D}_n$	n = 1	2.250	10.800	2.250	10.800	2.250	10.800	2.300	7.800
		n = 15	2.250	10.800	2.350	10.800	2.350	10.800	2.400	7.800
	RCR $\text{D}_n$	n = 1	2.250	10.800	2.250	10.800	2.250	10.800	2.300	3.900
		n = 15	2.250	10.800	2.250	10.800	2.250	10.800	2.400	4.100
	ROL $\text{D}_n$	n = 1	2.250	10.800	2.350	10.800	2.350	10.800	2.500	4.600
		n = 15	2.250	10.800	2.350	10.800	2.350	10.800	2.400	4.600
	RCL $\text{D}_n$	n = 1	2.250	11.500	2.300	11.500	2.300	11.500	2.400	7.500
		n = 15	2.250	11.500	2.300	11.500	2.300	11.500	2.500	7.500
	DROR $\text{D}_n$	n = 1	2.350	11.500	2.350	11.500	2.350	11.500	2.400	10.300
		n = 31	2.350	11.500	2.350	11.500	2.350	11.500	2.500	10.300
	DRCR $\text{D}_n$	n = 1	2.350	13.300	2.350	13.300	2.350	13.300	2.500	12.700
		n = 31	2.350	14.900	2.350	14.900	2.350	14.900	2.500	12.700
	DROL $\text{D}_n$	n = 1	2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
		n = 31	2.350	10.800	2.350	10.800	2.350	10.800	2.500	11.800
	DRCL $\text{D}_n$	n = 1	2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
		n = 31	2.350	13.300	2.350	13.300	2.350	13.300	2.500	5.100
	SFR $\text{D}_n$	n = 1	2.350	9.900	2.350	9.900	2.350	9.900	2.400	6.100
		n = 15	2.350	9.900	2.350	9.900	2.350	9.900	2.300	5.700
	SFL $\text{D}_n$	n = 1	2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
		n = 15	2.350	9.850	2.350	9.850	2.350	9.850	2.400	4.300
	DSFR $\text{D}_n$	n = 1	3.250	15.500	3.250	15.500	3.250	15.500	3.300	12.000
		n = 96	32.600	45.000	32.600	45.000	32.600	45.000	32.600	42.200
	DSFL $\text{D}_n$	n = 1	3.200	15.500	3.200	15.500	3.200	15.500	3.300	8.200
		n = 96	32.600	45.100	32.600	45.100	32.600	45.100	32.600	37.700
SUM	$\text{S} = 0$	3.100	8.950	3.100	8.950	3.100	8.950	3.400	6.700	
	$\text{S} = \text{FFFFH}$	3.000	8.850	3.000	8.850	3.000	8.850	3.500	6.700	
SEG	When executed	2.100	7.700	2.100	7.700	2.100	7.700	2.100	5.900	
FOR	—	1.500	7.500	1.500	7.500	1.500	7.500	1.200	6.300	
CALL Pn	Internal file pointer	4.800	5.400	4.800	5.400	4.800	5.400	2.700	4.800	
	Common pointer	7.100	30.500	7.100	30.500	7.100	30.500	4.400	5.700	
CALL Pn $\text{S1}$ to $\text{S5}$	—	50.200	62.000	50.200	62.000	50.200	62.000	28.700	42.600	

### Remark

For the instructions for which a leading edge instruction ( $\square P$ ) is not described, the processing time is the same as an ON execution instruction.

**Example** MOV P instruction, WAND P instruction etc.

(b) When using Q03UD(E)HCPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU, Q26UD(E)HCPU

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/Q06UD(E)HCPU		Q10/Q13/Q20/Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	LD LDI AND ANI OR ORI LDP LDF ANDP ANDF ORP ORF	When executed	0.020		0.0095		0.0095	
	LDPI LDFI	When executed	0.060		0.0285		0.0285	
	ANDPI ANDFI ORPI ORFI	When executed	0.080		0.038		0.038	
	OUT	When not changed	0.020		0.0095		0.0095	
		When changed						
SET RST	When not executed	0.020		0.0095		0.0095		

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LD=	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	AND=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	OR=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	LD<>	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	AND<>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	OR<>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	LD>	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	AND>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	OR>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	LD<=	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	AND<=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	OR<=	When not executed		0.060	0.0285	0.0285		
When executed		In conductive status						
		In non-conductive status						
LD<	In conductive status		0.060	0.0285	0.0285			
	In non-conductive status							
AND<	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						
OR<	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						
LD>=	In conductive status		0.060	0.0285	0.0285			
	In non-conductive status							
AND>=	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						
OR>=	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	LDD=	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	ANDD=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	ORD=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	LDD<>	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	ANDD<>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	ORD<>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	LDD>	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	ANDD>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	ORD>	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	LDD<=	In conductive status		0.060	0.0285	0.0285		
		In non-conductive status						
	ANDD<=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
	ORD<=	When not executed		0.060	0.0285	0.0285		
		When executed	In conductive status					
			In non-conductive status					
LDD<	In conductive status		0.060	0.0285	0.0285			
	In non-conductive status							
ANDD<	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						
ORD<	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						
LDD>=	In conductive status		0.060	0.0285	0.0285			
	In non-conductive status							
ANDD>=	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						
ORD>=	When not executed		0.060	0.0285	0.0285			
	When executed	In conductive status						
		In non-conductive status						



Category	Instruction	Condition (Device)	Processing Time (μs)						
			Q03UD(E)CPU		Q04/Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	
Basic instruction	+ (S) (D)	When executed	0.060		0.0285		0.0285		
	+ (S1) (S2) (D)	When executed	0.080		0.038		0.038		
	- (S) (D)	When executed	0.060		0.0285		0.0285		
	- (S1) (S2) (D)	When executed	0.080		0.038		0.038		
	D + (S) (D)	When executed	0.060		0.0285		0.0285		
	D + (S1) (S2) (D)	When executed	0.080		0.038		0.038		
	D - (S) (D)	When executed	0.060		0.0285		0.0285		
	D - (S1) (S2) (D)	When executed	0.080		0.038		0.038		
	* (S1) (S2) (D)	When executed	0.120		0.057		0.057		
	/ (S1) (S2) (D)	When executed	0.220		0.110		0.110		
	D * (S1) (S2) (D)	When executed	0.200		0.095		0.095		
	D / (S1) (S2) (D)	When executed	0.340		0.170		0.170		
	B + (S) (D)	When executed	3.300	5.500	3.000	4.100	3.000	4.100	
	B + (S1) (S2) (D)	When executed	4.600	6.200	4.200	5.900	4.200	5.900	
	B - (S) (D)	When executed	3.300	4.400	2.900	3.800	2.900	3.800	
	B - (S1) (S2) (D)	When executed	4.600	6.300	4.200	4.600	4.200	4.600	
	B * (S1) (S2) (D)	When executed	4.000	4.800	3.400	4.800	3.400	4.800	
	B / (S1) (S2) (D)	When executed	4.200	5.700	3.700	5.200	3.700	5.200	
	E + (S) (D)	Single precision	(S) = 0, (D) = 0	0.120		0.057		0.057	
			(S) = 2 <sup>127</sup> , (D) = 2 <sup>127</sup>	0.120		0.057		0.057	
	E + (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0	0.140		0.0665		0.0665	
			(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>	0.140		0.0665		0.0665	
	E - (S) (D)	Single precision	(S) = 0, (D) = 0	0.120		0.057		0.057	
			(S) = 2 <sup>127</sup> , (D) = 2 <sup>127</sup>	0.120		0.057		0.057	
	E - (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0	0.140		0.0665		0.0665	
			(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>	0.140		0.0665		0.0665	
	E * (S1) (S2) (D)	Single precision	(S1) = 0, (S2) = 0	0.120		0.057		0.057	
			(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>	0.120		0.057		0.057	
	E / (S1) (S2) (D)	Single precision	(S1) = 2 <sup>127</sup> , (S2) = 2 <sup>127</sup>	4.500	5.600	3.900	4.900	0.285	
	INC	When executed	0.040		0.019		0.019		
	DINC	When executed	0.040		0.019		0.019		
	DEC	When executed	0.040		0.019		0.019		
DDEC	When executed	0.040		0.019		0.019			
BCD	When executed	0.120		0.057		0.057			
DBCD	When executed	0.200		0.095		0.095			
BIN	When executed	0.060		0.0285		0.0285			
DBIN	When executed	0.060		0.0285		0.0285			

Category	Instruction	Condition (Device)		Processing Time (μs)					
				Q03UD(E)CPU		Q04/Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	FLT	Single precision	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 7FFF <sub>H</sub>	0.100		0.0475		0.0475	
	DFLT	Single precision	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 7FFFFFFF <sub>H</sub>	0.100		0.0475		0.0475	
	INT	Single precision	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 32766.5	0.100		0.0475		0.0475	
	DINT	Single precision	Ⓢ = 0	0.100		0.0475		0.0475	
			Ⓢ = 1234567890.3	0.100		0.0475		0.0475	
	MOV		—	0.040		0.019		0.019	
	DMOV		—	0.040		0.019		0.019	
	EMOV		—	0.040		0.019		0.019	
	CML		—	0.040		0.019		0.019	
	DCML		—	0.040		0.019		0.019	
	BMOV	n = 1		6.300	8.200	5.400	7.000	5.400	7.000
			SM237=OFF <sup>*1</sup>	8.200	10.600	3.900	5.100	3.900	5.100
			SM237=ON <sup>*1</sup>	6.000	7.800	2.900	3.700	2.900	3.700
				7.100	8.800	5.900	7.600	5.900	7.600
		n = 96		9.300	11.900	4.400	5.700	4.400	5.700
			SM237=OFF <sup>*1</sup>	7.100	9.100	3.400	4.300	3.400	4.300
			SM237=ON <sup>*1</sup>						
				5.300	5.900	4.200	4.800	4.200	4.800
	FMOV	n = 1		7.000	8.000	3.400	3.800	3.400	3.800
			SM237=OFF <sup>*1</sup>	5.900	6.800	2.800	3.200	2.800	3.200
			SM237=ON <sup>*1</sup>						
		n = 96		5.300	7.600	4.400	6.800	4.400	6.800
			SM237=OFF <sup>*1</sup>	7.400	12.200	3.600	5.800	3.600	5.800
SM237=ON <sup>*1</sup>			6.300	11.000	3.000	5.200	3.000	5.200	
XCH		—	2.500	2.900	1.800	2.300	1.800	2.300	
DXCH		—	2.800	3.700	2.100	2.900	2.100	2.900	
DFMOV <sup>*2</sup>	n=1	SM237=OFF	2.600	3.750	2.250	3.150	2.250	3.150	
		SM237=ON	2.050	2.250	1.750	1.750	1.750	1.750	
	n=96	SM237=OFF	5.850	7.350	4.200	5.500	4.200	5.500	
		SM237=ON	5.300	6.000	3.650	4.150	3.650	4.150	
CJ		—	1.800	2.800	1.400	2.400	1.400	2.400	
SCJ		—	1.800	2.800	1.400	2.400	1.400	2.400	
JMP		—	1.800	2.800	1.100	2.400	1.100	2.400	

\*1 : Can be used only for the Q03UDCPU, Q04UDHCPU and Q06UDHCPU whose first 5 digits of serial number is "10012" or later.

\*2 : Can be used only for the Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q13UD(E)HCPU and Q26UD(E)HCPU whose first 5 digits of serial number is "10012" or later.

Category	Instruction	Condition (Device)	Processing Time (µs)					
			Q03UD(E)CPU		Q04/Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	WAND (S) (D)	When executed	0.060		0.0285		0.0285	
	WAND (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	DAND (S) (D)	When executed	0.060		0.0285		0.0285	
	DAND (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	WOR (S) (D)	When executed	0.060		0.0285		0.0285	
	WOR (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	DOR (S) (D)	When executed	0.060		0.0285		0.0285	
	DOR (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	WXOR (S) (D)	When executed	0.060		0.0285		0.0285	
	WXOR (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	DXOR (S) (D)	When executed	0.060		0.0285		0.0285	
	DXOR (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	WXNR (S) (D)	When executed	0.060		0.0285		0.0285	
	WXNR (S1) (S2) (D)	When executed	0.080		0.038		0.038	
	DXNR (S) (D)	When executed	0.060		0.0285		0.0285	
	DXNR (S1) (S2) (D)	When executed	0.080		0.038		0.038	

A

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ROR $\text{\textcircled{D}}$ n	n = 1	2.300	3.100	1.700	2.500	1.700	2.500
		n = 15	2.400	3.100	1.800	2.500	1.800	2.500
	RCR $\text{\textcircled{D}}$ n	n = 1	2.300	3.900	1.700	3.200	1.700	3.200
		n = 15	2.400	4.100	1.700	3.200	1.700	3.200
	ROL $\text{\textcircled{D}}$ n	n = 1	2.400	3.300	1.800	3.200	1.800	3.200
		n = 15	2.400	3.300	1.800	3.200	1.800	3.200
	RCL $\text{\textcircled{D}}$ n	n = 1	2.400	2.700	1.800	2.100	1.800	2.100
		n = 15	2.400	2.800	1.800	2.200	1.800	2.200
	DROR $\text{\textcircled{D}}$ n	n = 1	2.400	3.400	1.900	2.700	1.900	2.700
		n = 31	2.500	3.400	1.900	2.700	1.900	2.700
	DRCR $\text{\textcircled{D}}$ n	n = 1	2.500	4.800	1.900	4.200	1.900	4.200
		n = 31	2.500	4.900	1.900	4.200	1.900	4.200
	DROL $\text{\textcircled{D}}$ n	n = 1	2.500	3.900	1.800	3.200	1.800	3.200
		n = 31	2.500	3.900	1.800	3.300	1.800	3.300
	DRCL $\text{\textcircled{D}}$ n	n = 1	2.500	4.800	1.900	3.800	1.900	3.800
		n = 31	2.500	4.600	1.900	3.800	1.900	3.800
	SFR $\text{\textcircled{D}}$ n	n = 1	2.400	3.900	1.700	2.600	1.700	2.600
		n = 15	2.300	3.900	1.800	2.600	1.800	2.600
	SFL $\text{\textcircled{D}}$ n	n = 1	2.400	4.300	1.800	2.700	1.800	2.700
		n = 15	2.400	4.300	1.800	2.700	1.800	2.700
	DSFR $\text{\textcircled{D}}$ n	n = 1	2.700	4.800	2.200	4.300	2.200	4.300
		n = 96	32.600	35.900	23.900	26.100	23.900	26.100
	DSFL $\text{\textcircled{D}}$ n	n = 1	2.700	4.600	2.100	4.000	2.100	4.000
		n = 96	32.600	35.300	23.700	25.800	23.700	25.800
SUM	$\text{\textcircled{S}} = 0$	3.400	4.300	2.900	3.600	2.900	3.600	
	$\text{\textcircled{S}} = \text{FFFFH}$	3.500	4.200	2.900	3.600	2.900	3.600	
SEG	When executed	2.100	2.800	1.500	2.100	1.500	2.100	
FOR	—	1.200	2.400	0.870	2.100	0.870	2.100	
CALL Pn	Internal file pointer	2.600	4.000	2.300	3.600	2.300	3.600	
	Common pointer	4.000	5.300	3.200	4.900	3.200	4.900	
CALL Pn $\text{\textcircled{S1}}$ to $\text{\textcircled{S5}}$	—	28.700	33.400	26.100	29.300	26.100	29.300	

**Remark**

For the instructions for which a leading edge instruction ( $\square P$ ) is not described, the processing time is the same as an ON execution instruction.

**Example** MOVDP instruction, WANDP instruction etc.

(2) Table of the time to be added when file register, module access device is used

(a) When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Device name		data	Device Specification Location	Processing Time (μs)			
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.100	0.100
			Destination	0.220	0.220	0.220	0.220
		Word	Source	0.100	0.100	0.100	0.100
			Destination	0.100	0.100	0.100	0.100
		Double word	Source	0.200	0.200	0.200	0.200
			Destination	0.200	0.200	0.200	0.200
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.220
			Destination	—	—	—	0.420
		Word	Source	—	—	—	0.220
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.440
			Destination	—	—	—	0.380
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.160
			Destination	—	—	—	0.320
		Word	Source	—	—	—	0.160
			Destination	—	—	—	0.140
		Double word	Source	—	—	—	0.320
			Destination	—	—	—	0.300
File register (ZR)/ Extended data register (D)/Extended link register (W))	When standard RAM is used	Bit	Source	0.220	0.180	0.160	0.140
			Destination	0.280	0.320	0.300	0.280
		Word	Source	0.220	0.180	0.160	0.140
			Destination	0.220	0.180	0.160	0.140
		Double word	Source	0.320	0.280	0.260	0.240
			Destination	0.320	0.280	0.260	0.240
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.260
			Destination	—	—	—	0.480
		Word	Source	—	—	—	0.260
			Destination	—	—	—	0.220
		Double word	Source	—	—	—	0.480
			Destination	—	—	—	0.420
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.200
			Destination	—	—	—	0.380
		Word	Source	—	—	—	0.200
			Destination	—	—	—	0.180
		Double word	Source	—	—	—	0.360
			Destination	—	—	—	0.340
Module access device (Multiple CPU high speed trans- mission area) (U3EnG10000)	Bit	Source	—	—	—	—	
		Destination	—	—	—	—	
	Word	Source	—	—	—	—	
		Destination	—	—	—	—	
	Double word	Source	—	—	—	—	
		Destination	—	—	—	—	

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

(b) When using Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UDE(H)CPU, Q20UD(E)HCPU and Q26UD(E)HCPU

Device name	data	Device Specification Location	Processing Time (μs)			
			Q03UD(E)CPU	Q04/Q06UD(E)HCPU	Q10/Q13/Q20/Q26UD(E)HCPU	
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048	0.048
			Destination	0.100	0.038	0.038
		Word	Source	0.100	0.048	0.048
			Destination	0.100	0.038	0.038
	Double word	Source	0.200	0.095	0.095	
		Destination	0.200	0.086	0.086	
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.220	0.200	0.200
			Destination	0.180	0.162	0.162
		Word	Source	0.220	0.200	0.200
			Destination	0.180	0.162	0.162
	Double word	Source	0.440	0.399	0.399	
		Destination	0.380	0.361	0.361	
When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.160	0.152	0.152	
		Destination	0.140	0.133	0.133	
	Word	Source	0.160	0.152	0.152	
		Destination	0.140	0.133	0.133	
Double word	Source	0.320	0.304	0.304		
	Destination	0.300	0.295	0.295		
File register (ZR)/ Extended data register (D)/Extended link register (W)	When standard RAM is used	Bit	Source	0.120	0.057	0.057
			Destination	0.120	0.048	0.048
		Word	Source	0.120	0.057	0.057
			Destination	0.120	0.048	0.048
	Double word	Source	0.220	0.105	0.105	
		Destination	0.220	0.095	0.095	
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.240	0.209	0.209
			Destination	0.200	0.171	0.171
		Word	Source	0.240	0.209	0.209
			Destination	0.200	0.171	0.171
	Double word	Source	0.460	0.409	0.409	
		Destination	0.400	0.371	0.371	
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.180	0.162	0.162
			Destination	0.160	0.143	0.143
		Word	Source	0.180	0.162	0.162
			Destination	0.160	0.143	0.143
Double word	Source	0.340	0.314	0.314		
	Destination	0.320	0.304	0.304		
Module access device (Multiple CPU high speed transmission area) (U3En\G10000)	Bit	Source	0.220	0.181	0.181	
		Destination	0.140	0.105	0.105	
	Word	Source	0.220	0.181	0.181	
		Destination	0.140	0.105	0.105	
	Double word	Source	0.500	0.437	0.437	
		Destination	0.340	0.285	0.285	

(3) Table of the time to be added when F/T(ST)/C device is used in OUT/SET/RST instruction

(a) When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU.

Instruction name	Device name	Condition	Processing Time (μs)				
			Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU	
OUT	F	When not executed		2.900	2.900	2.900	2.100
		When executed	When displayed	116.000	116.000	116.000	68.800
			Display completed	116.000	116.000	116.000	61.600
	T(ST), C	When not executed		0.360	0.240	0.180	0.120
		When executed	After time up	0.360	0.240	0.180	0.120
			When added	0.360	0.240	0.180	0.120
SET	F	When not executed		0.120	0.080	0.006	0.004
		When executed	When displayed	116.000	116.000	116.000	68.600
			Display completed	116.000	116.000	116.000	65.700
RST	F	When not executed		0.120	0.080	0.006	0.004
		When executed	When displayed	55.800	55.800	55.800	26.500
			Display completed	29.200	29.200	29.200	21.600
	T(ST), C	When not executed		0.360	0.240	0.180	0.120
		When executed		0.360	0.240	0.180	0.120

(b) When using Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU and Q26UD(E)HCPU

Instruction name	Device name	Condition	Processing Time (μs)			
			Q03UD(E)CPU	Q04/Q06UD(E)HCPU	Q10/Q13/Q20/Q26UD(E)HCPU	
OUT	F	When not executed		1.940	1.570	1.570
		When executed	When displayed	39.930	38.090	38.090
			Display completed	39.750	37.980	37.980
	T(ST), C	When not executed		0.060	0.030	0.030
		When executed	After time up	0.060	0.030	0.030
SET	F	When not executed		0.000	0.000	0.000
		When executed	When displayed	42.900	40.600	40.600
			Display completed	39.270	37.900	37.900
RST	F	When not executed		0.000	0.000	0.000
		When executed	When displayed	45.260	36.600	36.600
			Display completed	19.020	16.190	16.190
	T(ST), C	When not executed		0.060	0.030	0.030
		When executed		0.060	0.030	0.030

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

## Appendix 1.4.2 Processing time of instructions other than subset instruction

The following table shows the processing time of instructions other than subset instructions.

(1) Table of the processing time of instructions other than subset instructions

(a) When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	ANB ORB MPS MRD MPP	—	0.120		0.080		0.060		0.040	
	INV	When not executed	0.120		0.080		0.060		0.040	
		When executed	0.120		0.080		0.060		0.040	
	MEP MEF	When not executed	0.120		0.080		0.060		0.040	
		When executed	0.120		0.080		0.060		0.040	
	EGP EGF	When not executed	0.120		0.080		0.060		0.040	
		When executed	0.120		0.080		0.060		0.040	
	PLS	—	1.800	1.900	1.800	1.900	1.800	1.900	1.300	1.600
	PLF	—	1.800	1.900	1.800	1.900	1.800	1.900	1.600	1.700
	FF	When not executed	0.240		0.160		0.120		0.080	
		When executed	1.700	1.800	1.700	1.800	1.700	1.800	1.200	1.500
	DELTA	When not executed	0.240		0.160		0.120		0.080	
		When executed	4.000	14.700	4.000	14.700	4.000	14.700	2.800	3.600
	SFT	When not executed	0.240		0.160		0.120		0.800	
		When executed	1.800	12.600	1.800	12.600	1.800	12.600	1.600	6.600
	MC	—	0.240		0.160		0.120		0.080	
	MCR	—	0.120		0.080		0.060		0.040	
	FEND	Error check performed	250.000	250.000	250.000	250.000	250.000	250.000	175.000	252.000
	END	No error check performed	250.000	250.000	250.000	250.000	250.000	250.000	175.000	221.000



Category	Instruction	Condition (Device)		Processing Time (μs)									
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU			
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.		
Sequence instruction	NOP NOPLF PAGE	—		0.120		0.080		0.060		0.040			
Basic instruction	LDE=	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100	
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.100	
	ANDE=	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.200	12.500
				In non-conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.400	11.900
	ORE=	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.800
				In non-conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.500	9.800
	LDE<>	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	7.700	
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.600	8.200	
	ANDE<>	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.300	14.200
				In non-conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.400	14.200
	ORE<>	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.600	6.700
				In non-conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.400	6.600
	LDE>	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	13.700	
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.600	13.700	
	ANDE>	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.300	8.100
				In non-conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.200	8.100
	ORE>	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.600	8.500
				In non-conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.400	8.100
	LDE<=	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.100	
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	9.600	
	ANDE<=	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.100	7.800
In non-conductive status				4.200	19.600	4.200	19.600	4.200	19.600	4.400	8.200		
ORE<=	Single precision	When not executed		0.360		0.240		0.180		0.120			
		When executed	In conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.500	10.300	
			In non-conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800	
LDE<	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.500		
		In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	10.900		
ANDE<	Single precision	When not executed		0.360		0.240		0.180		0.120			
		When executed	In conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.300	9.200	
			In non-conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.400	9.400	
ORE<	Single precision	When not executed		0.360		0.240		0.180		0.120			
		When executed	In conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.600	10.400	
			In non-conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.400	9.800	

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)		Processing Time (μs)									
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU			
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.		
Basic instruction	LDE>=	Single precision	In conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	12.200	
			In non-conductive status		4.400	20.900	4.400	20.900	4.400	20.900	4.700	11.800	
	ANDE>=	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.100	6.700
				In non-conductive status		4.200	19.600	4.200	19.600	4.200	19.600	4.400	7.000
	ORE>=	Single precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.600	14.000
				In non-conductive status		4.200	17.400	4.200	17.400	4.200	17.400	4.500	14.300
	LDED=	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	21.000	
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	21.900	
	ANDED=	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.500	34.700	4.500	34.700	4.500	34.700	3.800	17.800
				In non-conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.100	18.100
	ORED=	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.800
				In non-conductive status		4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.500
	LDED<>	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	23.500	
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.600	
	ANDED<>	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.800
				In non-conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.000	18.700
	ORED<>	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200
				In non-conductive status		4.700	33.200	4.700	33.200	4.700	33.200	4.100	23.400
	LDED>	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.100	
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	23.400	
	ANDED>	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.500
In non-conductive status				4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700		
ORED>	Double precision	When not executed		0.360		0.240		0.180		0.120			
		When executed	In conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	24.200	
			In non-conductive status		4.700	33.200	4.700	33.200	4.700	33.200	4.900	25.800	
LDED<=	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	22.500		
		In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.500		
ANDED<=	Double precision	When not executed		0.360		0.240		0.180		0.120			
		When executed	In conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.600	
			In non-conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700	
ORED<=	Double precision	When not executed		0.360		0.240		0.180		0.120			
		When executed	In conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	26.300	
			In non-conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.200	

Category	Instruction	Condition (Device)		Processing Time (µs)									
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU			
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.		
Basic instruction	LDED<	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	5.100	25.000	
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	24.100	
	ANDED<	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.000	19.400
				In non-conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.700
	ORED<	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
				In non-conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
	LDED>=	Double precision	In conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.200	13.100	
			In non-conductive status		4.700	37.400	4.700	37.400	4.700	37.400	4.300	13.100	
	ANDED>=	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.500	34.700	4.500	34.700	4.500	34.700	3.900	19.500
				In non-conductive status		4.500	34.700	4.500	34.700	4.500	34.700	4.100	19.800
	ORED>=	Double precision	When not executed		0.360		0.240		0.180		0.120		
			When executed	In conductive status		4.700	33.200	4.700	33.200	4.700	33.200	5.000	25.100
				In non-conductive status		4.700	33.200	4.700	33.200	4.700	33.200	4.200	18.500
	LD\$=			In conductive status		8.300	38.500	8.300	38.500	8.300	38.500	5.500	14.900
				In non-conductive status		8.300	38.500	8.300	38.500	8.300	38.500	5.500	15.600
	AND\$=			When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status		7.200	37.300	7.200	37.300	7.200	37.300	5.200	13.800	
			In non-conductive status		7.200	37.300	7.200	37.300	7.200	37.300	5.300	14.500	
	OR\$=			When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status		7.500	36.600	7.500	36.600	7.500	36.600	5.500	14.900	
			In non-conductive status		7.500	36.600	7.500	36.600	7.500	36.600	5.300	14.600	
	LD\$<>			In conductive status		8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.200
				In non-conductive status		8.300	39.300	8.300	39.300	8.300	39.300	5.600	15.400
	AND\$<>			When not executed		0.360		0.240		0.180		0.120	
		When executed	In conductive status		8.000	38.200	8.000	38.200	8.000	38.200	4.300	21.500	
			In non-conductive status		8.000	38.200	8.000	38.200	8.000	38.200	4.500	23.400	
	OR\$<>			When not executed		0.360		0.240		0.180		0.120	
When executed		In conductive status		8.300	37.300	8.300	37.300	8.300	37.300	5.400	17.700		
		In non-conductive status		8.300	37.300	8.300	37.300	8.300	37.300	5.300	19.400		
LD\$>			In conductive status		8.300	41.600	8.300	41.600	8.300	41.600	6.400	19.200	
			In non-conductive status		8.300	41.600	8.300	41.600	8.300	41.600	5.600	20.100	
AND\$>			When not executed		0.360		0.240		0.180		0.120		
	When executed	In conductive status		8.000	38.100	8.000	38.100	8.000	38.100	4.500	15.400		
		In non-conductive status		8.000	38.100	8.000	38.100	8.000	38.100	4.600	15.300		
OR\$>			When not executed		0.360		0.240		0.180		0.120		
	When executed	In conductive status		8.200	35.700	8.200	35.700	8.200	35.700	5.400	20.000		
		In non-conductive status		8.200	35.700	8.200	35.700	8.200	35.700	5.400	22.100		
LD\$<=			In conductive status		8.300	39.200	8.300	39.200	8.300	39.200	5.800	12.800	
			In non-conductive status		8.300	39.200	8.300	39.200	8.300	39.200	6.300	13.900	
AND\$<=			When not executed		0.360		0.240		0.180		0.120		
	When executed	In conductive status		7.100	36.500	7.100	36.500	7.100	36.500	6.000	16.000		
		In non-conductive status		7.100	36.500	7.100	36.500	7.100	36.500	6.100	16.200		

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)	Processing Time (µs)								
			Q00UCPU		Q01UCPU		Q02UCPU				
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Basic instruction	OR\$<=	When not executed	0.360		0.240		0.180		0.120		
		When executed	In conductive status	7.400	35.600	7.400	35.600	7.400	35.600	4.700	14.600
			In non-conductive status	7.400	35.600	7.400	35.600	7.400	35.600	4.600	14.400
	LD\$<	In conductive status	7.400	40.000	7.400	40.000	7.400	40.000	4.800	17.000	
		In non-conductive status	7.400	40.000	7.400	40.000	7.400	40.000	5.500	18.000	
	AND\$<	When not executed	0.360		0.240		0.180		0.120		
		When executed	In conductive status	8.000	37.300	8.000	37.300	8.000	37.300	5.900	13.400
			In non-conductive status	8.000	37.300	8.000	37.300	8.000	37.300	6.200	14.500
	OR\$<	When not executed	0.360		0.240		0.180		0.120		
		When executed	In conductive status	8.300	35.600	8.300	35.600	8.300	35.600	6.200	18.700
			In non-conductive status	8.300	35.600	8.300	35.600	8.300	35.600	5.400	19.700
	LD\$>=	In conductive status	7.400	38.300	7.400	38.300	7.400	38.300	4.800	10.000	
		In non-conductive status	7.400	38.300	7.400	38.300	7.400	38.300	5.500	11.200	
	AND\$>=	When not executed	0.360		0.240		0.180		0.120		
		When executed	In conductive status	7.200	37.300	7.200	37.300	7.200	37.300	4.400	21.600
			In non-conductive status	7.200	37.300	7.200	37.300	7.200	37.300	4.500	21.800
	OR\$>=	When not executed	0.360		0.240		0.180		0.120		
		When executed	In conductive status	8.200	36.400	8.200	36.400	8.200	36.400	5.400	15.400
			In non-conductive status	8.200	36.400	8.200	36.400	8.200	36.400	5.300	15.300
	BKCMP = S1 S2 D n	n = 1	15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.600	
		n = 96	64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.500	
	BKCMP<> S1 S2 D n	n = 1	15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500	
		n = 96	66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500	
	BKCMP> S1 S2 D n	n = 1	15.300	36.100	15.300	36.100	15.300	36.100	8.200	23.100	
		n = 96	66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.400	
	BKCMP<= S1 S2 D n	n = 1	15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500	
		n = 96	64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400	
	BKCMP< S1 S2 D n	n = 1	15.300	36.100	15.300	36.100	15.300	36.100	8.300	23.000	
		n = 96	66.600	87.500	66.600	87.500	66.600	87.500	59.500	74.500	
	BKCMP>= S1 S2 D n	n = 1	15.300	36.100	15.300	36.100	15.300	36.100	8.200	22.500	
		n = 96	64.500	85.500	64.500	85.500	64.500	85.500	57.400	72.400	
	DBKCMPI = S1 S2 D n	n = 1	15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000	
n = 96		64.900	85.700	64.900	85.700	64.900	85.700	60.700	78.400		
DBKCMPI<> S1 S2 D n	n = 1	15.700	36.300	15.700	36.300	15.700	36.300	9.350	28.900		
	n = 96	67.000	87.700	67.000	87.700	67.000	87.700	62.500	80.300		
DBKCMPI> S1 S2 D n	n = 1	15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000		
	n = 96	67.000	87.700	67.000	87.700	67.000	87.700	62.600	80.300		
DBKCMPI<= S1 S2 D n	n = 1	15.700	36.300	15.700	36.300	15.700	36.300	9.350	29.000		
	n = 96	64.800	85.700	64.800	85.700	64.800	85.700	60.800	78.400		
DBKCMPI< S1 S2 D n	n = 1	15.800	36.300	15.800	36.300	15.800	36.300	9.350	29.000		
	n = 96	67.000	87.700	67.000	87.700	67.000	87.700	62.700	80.400		
DBKCMPI>= S1 S2 D n	n = 1	15.700	36.300	15.700	36.300	15.700	36.300	9.300	29.000		
	n = 96	64.800	85.700	64.800	85.700	64.800	85.700	60.700	78.400		

Category	Instruction	Condition (Device)	Processing Time ( $\mu$ s)								
			Q00UCPU		Q01UCPU		Q02UCPU		Q03UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Basic instruction	DB + (S) (D)	When executed	5.750	13.300	5.750	13.300	5.750	13.300	4.900	7.500	
	DB + (S1) (S2) (D)	When executed	5.650	13.200	5.650	13.200	5.650	13.200	5.200	11.000	
	DB - (S) (D)	When executed	5.750	12.700	5.750	12.700	5.750	12.700	4.900	10.200	
	DB - (S1) (S2) (D)	When executed	5.650	12.600	5.650	12.600	5.650	12.600	5.200	8.600	
	DB * (S1) (S2) (D)	When executed	8.750	40.200	8.750	40.200	8.750	40.200	8.300	22.200	
	DB/ (S1) (S2) (D)	When executed	5.750	21.500	5.750	21.500	5.750	21.500	6.100	19.200	
	ED + (S) (D)	Double precision	(S) = 0, (D) = 0	4.500	26.700	4.500	26.700	4.500	26.700	4.800	16.800
			(S) = $2^{1023}$ , (D) = $2^{1023}$	5.800	32.900	5.800	32.900	5.800	32.900	4.800	16.800
	ED + (S1) (S2) (D)	Double precision	(S1) = 0, (S2) = 0	5.450	35.400	5.450	35.400	5.450	35.400	7.100	20.100
			(S1) = $2^{1023}$ , (S2) = $2^{1023}$	6.750	41.400	6.750	41.400	6.750	41.400	7.100	20.100
	ED - (S) (D)	Double precision	(S) = 0, (D) = 0	5.200	25.900	5.200	25.900	5.200	25.900	5.000	17.300
			(S) = $2^{1023}$ , (D) = $2^{1023}$	6.000	27.700	6.000	27.700	6.000	27.700	5.000	17.300
	ED - (S1) (S2) (D)	Double precision	(S1) = 0, (S2) = 0	5.550	32.900	5.550	32.900	5.550	32.900	6.000	16.300
			(S1) = $2^{1023}$ , (S2) = $2^{1023}$	5.750	33.900	5.750	33.900	5.750	33.900	6.000	16.300
	ED * (S1) (S2) (D)	Double precision	(S1) = 0, (S2) = 0	5.550	34.400	5.550	34.400	5.550	34.400	10.500	22.300
			(S1) = $2^{1023}$ , (S2) = $2^{1023}$	5.950	39.100	5.950	39.100	5.950	39.100	10.500	22.300
	ED / (S1) (S2) (D)	Double precision	(S1) = $2^{1023}$ , (S2) = $2^{1023}$	8.050	44.200	8.050	44.200	8.050	44.200	7.500	27.200
	BK + (S1) (S2) (D) n		n = 1	13.500	28.500	13.500	28.500	13.500	28.500	12.100	19.700
			n = 96	63.100	78.200	63.100	78.200	63.100	78.200	61.700	69.300
	BK - (S1) (S2) (D) n		n = 1	13.500	28.500	13.500	28.500	13.500	28.500	12.100	20.600
			n = 96	63.100	78.200	63.100	78.200	63.100	78.200	61.700	70.200
	DBK + (S1) (S2) (D) n		n = 1	10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.200
			n = 96	59.800	73.900	59.800	73.900	59.800	73.900	59.400	68.900
	DBK - (S1) (S2) (D) n		n = 1	10.100	24.200	10.100	24.200	10.100	24.200	7.050	19.900
			n = 96	59.800	73.900	59.800	73.900	59.800	73.900	59.400	69.600
	\$ + (S) (D)		—	15.400	64.300	15.400	64.300	15.400	64.300	14.400	34.000
	\$ + (S1) (S2) (D)		—	19.700	71.000	19.700	71.000	19.700	71.000	9.200	22.900
	FLTD	Double precision	(S) = 0	3.100	19.600	3.100	19.600	3.100	19.600	4.000	8.900
(S) = 7FFFH			3.350	19.900	3.350	19.900	3.350	19.900	3.400	9.000	
DFLTD	Double precision	(S) = 0	3.200	20.400	3.200	20.400	3.200	20.400	4.100	10.800	
		(S) = 7FFFFFFFH	3.450	20.500	3.450	20.500	3.450	20.500	3.600	10.800	
INTD	Double precision	(S) = 0	3.200	22.900	3.200	22.900	3.200	22.900	3.500	9.300	
		(S) = 32766.5	4.100	34.300	4.100	34.300	4.100	34.300	5.100	19.500	
DINTD	Double precision	(S) = 0	3.200	23.000	3.200	23.000	3.200	23.000	2.600	6.800	
		(S) = 1234567890.3	4.050	33.500	4.050	33.500	4.050	33.500	3.400	11.700	

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	DBL	When executed	3.300	5.900	3.300	5.900	3.300	5.900	2.700	3.800
	WORD	When executed	3.000	7.250	3.000	7.250	3.000	7.250	2.900	7.000
	GRY	When executed	3.350	7.500	3.350	7.500	3.350	7.500	2.700	6.100
	DGRY	When executed	3.000	7.200	3.000	7.200	3.000	7.200	2.900	4.600
	GBIN	When executed	4.600	9.700	4.600	9.700	4.600	9.700	4.000	8.200
	DGBIN	When executed	5.550	10.700	5.550	10.700	5.550	10.700	5.500	8.000
	NEG	When executed	3.300	6.850	3.300	6.850	3.300	6.850	2.400	4.100
	DNEG	When executed	3.050	5.700	3.050	5.700	3.050	5.700	2.500	4.300
	ENEG	Floating point = 0	3.100	7.350	3.100	7.350	3.100	7.350	2.500	3.400
		Floating point = -1.0	3.350	11.700	3.350	11.700	3.350	11.700	2.700	4.500
	EDNEG	Floating point = 0	3.000	21.200	3.000	21.200	3.000	21.200	2.200	3.500
		Floating point = -1.0	3.100	22.900	3.100	22.900	3.100	22.900	2.400	3.500
	BKBCD $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	8.700	27.600	8.700	27.600	8.700	27.600	9.700	22.000
		n = 96	84.200	104.000	84.200	104.000	84.200	104.000	74.200	86.500
	BKBIN $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	8.450	28.100	8.450	28.100	8.450	28.100	8.900	16.300
		n = 96	56.100	75.800	56.100	75.800	56.100	75.800	58.500	65.100
	ECON	—	3.100	21.300	3.100	21.300	3.100	21.300	4.300	6.800
	EDCON	—	5.050	24.000	5.050	24.000	5.050	24.000	2.800	5.400
	EDMOV	—	2.900	22.900	2.900	22.900	2.900	22.900	3.200	7.800
	\$MOV	Character string to be transferred = 0	6.250	30.100	6.250	30.100	6.250	30.100	4.500	13.900
		Character string to be transferred = 32	15.500	39.300	15.500	39.300	15.500	39.300	15.400	17.500
	BXCH $\text{\textcircled{D1}}$ $\text{\textcircled{D2}}$ n	n = 1	8.400	20.900	8.400	20.900	8.400	20.900	8.700	15.200
		n = 96	67.100	79.900	67.100	79.900	67.100	79.900	67.200	74.000
	SWAP	—	3.300	3.550	3.300	3.550	3.300	3.550	2.400	2.700
	GOEND	—	0.550		0.550		0.550		0.500	
	DI	—	2.800	8.400	2.800	8.400	2.800	8.400	1.800	2.200
	EI	—	4.300	12.300	4.300	12.300	4.300	12.300	3.100	3.800
	IMASK	—	12.900	40.600	12.900	40.600	12.900	40.600	9.800	25.000
	IRET	—	1.000		1.000		1.000		1.000	
	RSF X n	n = 1	7.500	26.500	7.500	26.500	7.500	26.500	4.300	16.100
		n = 96	11.400	30.400	11.400	30.400	11.400	30.400	11.400	23.700
	RSF Y n	n = 1	7.300	26.300	7.300	26.300	7.300	26.300	3.800	10.000
		n = 96	10.900	29.900	10.900	29.900	10.900	29.900	8.500	15.200
	UDCNT1	—	1.500	7.100	1.500	7.100	1.500	7.100	1.000	2.000
	UDCNT2	—	1.500	6.300	1.500	6.300	1.500	6.300	1.000	4.000
	TTMR	—	5.300	20.900	5.300	20.900	5.300	20.900	3.900	6.100
	STMR	—	8.900	49.800	8.900	49.800	8.900	49.800	7.200	30.000
	ROTC	—	52.300	52.600	52.300	52.600	52.300	52.600	15.200	16.100
	RAMP	—	7.400	30.900	7.400	30.900	7.400	30.900	5.900	18.300
	SPD	—	1.500	6.300	1.500	6.300	1.500	6.300	1.000	2.800
PLSY	—	6.400	7.100	6.400	7.100	6.400	7.100	3.500	4.700	
PWM	—	3.900	4.600	3.900	4.600	3.900	4.600	3.400	3.400	
MTR	—	10.100	61.400	10.100	61.400	10.100	61.400	20.500	28.400	

Category	Instruction	Condition (Device)	Processing Time (μs)								
			Q00UCPU		Q01UCPU		Q02UCPU		Q03UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	BKAND $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	13.600	28.500	13.600	28.500	13.600	28.500	12.100	20.100	
		n = 96	63.200	78.200	63.200	78.200	63.200	78.200	57.400	63.200	
	BKOR $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	13.500	28.500	13.500	28.500	13.500	28.500	7.700	13.200	
		n = 96	63.100	78.200	63.100	78.200	63.100	78.200	57.400	62.800	
	BKXOR $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	13.600	28.300	13.600	28.300	13.600	28.300	7.800	13.200	
		n = 96	63.100	78.000	63.100	78.000	63.100	78.000	57.300	62.800	
	BKXNR $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	13.500	28.300	13.500	28.300	13.500	28.300	7.800	14.100	
		n = 96	63.100	78.000	63.100	78.000	63.100	78.000	57.400	62.900	
	BSFR $\textcircled{D}$ n	n = 1	5.050	21.100	5.050	21.100	5.050	21.100	3.700	6.300	
		n = 96	9.000	34.800	9.000	34.800	9.000	34.800	10.200	12.800	
	BSFL $\textcircled{D}$ n	n = 1	4.800	19.100	4.800	19.100	4.800	19.100	4.500	8.900	
		n = 96	8.550	34.300	8.550	34.300	8.550	34.300	10.100	14.300	
	SFTBR $\textcircled{D}$ n1 n2	n1 = 16 / n2 = 1	10.300	46.500	10.300	46.500	10.300	46.500	8.800	43.400	
		n1 = 16 / n2 = 15	10.300	46.400	10.300	46.400	10.300	46.400	8.750	43.400	
	SFTBL $\textcircled{D}$ n1 n2	n1 = 16 / n2 = 1	10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100	
		n1 = 16 / n2 = 15	10.500	49.800	10.500	49.800	10.500	49.800	8.050	45.100	
	SFTWR $\textcircled{D}$ n1 n2	n1 = 16 / n2 = 1	7.950	24.000	7.950	24.000	7.950	24.000	6.500	22.800	
		n1 = 16 / n2 = 15	7.950	24.100	7.950	24.100	7.950	24.100	6.500	22.800	
	SFTWL $\textcircled{D}$ n1 n2	n1 = 16 / n2 = 1	8.700	23.600	8.700	23.600	8.700	23.600	7.350	23.600	
		n1 = 16 / n2 = 15	8.650	23.700	8.650	23.700	8.650	23.700	7.300	23.700	
	BSET $\textcircled{D}$ n	n = 1	4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.400	
		n = 15	4.550	4.750	4.550	4.750	4.550	4.750	3.000	3.500	
	BRST $\textcircled{D}$ n	n = 1	4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400	
		n = 15	4.600	4.750	4.600	4.750	4.600	4.750	3.000	3.400	
	TEST	When executed	7.250	13.200	7.250	13.200	7.250	13.200	4.400	6.900	
	DTEST	When executed	6.950	12.900	6.950	12.900	6.950	12.900	4.500	7.000	
	BKRST $\textcircled{D}$ n	n = 1	7.350	11.600	7.350	11.600	7.350	11.600	4.300	5.200	
		n = 96	10.100	22.600	10.100	22.600	10.100	22.600	6.500	13.200	
	SER $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	All match	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
			None match	6.650	6.800	6.650	6.800	6.650	6.800	5.000	5.300
		n = 96	All match	34.000	42.300	34.000	42.300	34.000	42.300	32.300	35.900
			None match	34.000	42.300	34.000	42.300	34.000	42.300	32.400	35.900
DSER $\textcircled{S1}$ $\textcircled{S2}$ $\textcircled{D}$ n	n = 1	All match	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200	
		None match	8.000	16.300	8.000	16.300	8.000	16.300	6.800	10.200	
	n = 96	All match	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300	
		None match	54.100	62.600	54.100	62.600	54.100	62.600	52.800	56.300	
DSUM $\textcircled{S}$ $\textcircled{D}$	$\textcircled{S} = 0$	4.100	4.200	4.100	4.200	4.100	4.200	3.700	4.100		
	$\textcircled{S} = \text{FFFFFFFF}_H$	4.100	4.200	4.100	4.200	4.100	4.200	3.800	4.100		
DECO $\textcircled{S}$ $\textcircled{D}$ n	n = 2	8.850	23.000	8.850	23.000	8.850	23.000	6.000	16.400		
	n = 8	13.600	36.600	13.600	36.600	13.600	36.600	8.100	15.200		
ENCO $\textcircled{S}$ $\textcircled{D}$ n	n = 2	M1 = ON	7.650	11.900	7.650	11.900	7.650	11.900	5.300	6.300	
		M4 = ON	7.500	11.700	7.500	11.700	7.500	11.700	5.200	6.200	
	n = 8	M1 = ON	14.600	27.800	14.600	27.800	14.600	27.800	10.400	17.900	
		M256 = ON	10.600	23.700	10.600	23.700	10.600	23.700	5.700	13.300	

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	DIS (S) (D) n	n = 1	6.500	14.800	6.500	14.800	6.500	14.800	5.000	10.900
		n = 4	6.900	15.200	6.900	15.200	6.900	15.200	5.400	11.300
	UNI (S) (D) n	n = 1	6.800	15.100	6.800	15.100	6.800	15.100	5.500	8.900
		n = 4	7.500	15.900	7.500	15.900	7.500	15.900	6.200	9.600
	NDIS	When executed	4.750	18.700	4.750	18.700	4.750	18.700	11.000	16.300
	NUNI	When executed	4.750	18.700	4.750	18.700	4.750	18.700	10.600	16.000
	WTOB (S) (D) n	n = 1	6.600	14.900	6.600	14.900	6.600	14.900	5.000	6.500
		n = 96	37.700	46.100	37.700	46.100	37.700	46.100	36.000	38.400
	BTOW (S) (D) n	n = 1	7.350	15.600	7.350	15.600	7.350	15.600	5.100	6.100
		n = 96	32.100	40.500	32.100	40.500	32.100	40.500	29.900	32.000
	MAX (S) (D) n	n = 1	8.250	24.900	8.250	24.900	8.250	24.900	4.300	6.900
		n = 96	34.200	51.600	34.200	51.600	34.200	51.600	32.000	34.300
	MIN (S) (D) n	n = 1	8.250	24.800	8.250	24.800	8.250	24.800	4.400	6.800
		n = 96	34.200	51.600	34.200	51.600	34.200	51.600	30.300	34.800
	DMAX (S) (D) n	n = 1	6.800	34.900	6.800	34.900	6.800	34.900	4.800	14.200
		n = 96	60.300	89.200	60.300	89.200	60.300	89.200	56.400	68.000
	DMIN (S) (D) n	n = 1	7.600	35.700	7.600	35.700	7.600	35.700	4.800	9.300
		n = 96	59.400	90.000	59.400	90.000	59.400	90.000	55.400	62.800
	SORT (S1) n (S2) (D1) (D2)	n = 1	10.100	28.900	10.100	28.900	10.100	28.900	6.200	12.200
		n = 96	52.100	92.400	52.100	92.400	52.100	92.400	6.200	13.100
	DSORT (S1) n (S2) (D1) (D2)	n = 1	9.300	29.000	9.300	29.000	9.300	29.000	6.200	10.500
		n = 96	43.600	89.600	43.600	89.600	43.600	89.600	6.100	10.500
	WSUM (S) (D) n	n = 1	6.700	15.000	6.700	15.000	6.700	15.000	4.800	6.200
		n = 96	28.900	37.100	28.900	37.100	28.900	37.100	26.900	28.700
	DWSUM (S) (D) n	n = 1	8.600	26.800	8.600	26.800	8.600	26.800	5.500	7.000
		n = 96	56.200	74.700	56.200	74.700	56.200	74.700	53.000	56.300
	MEAN (S) (D) n	n = 1	5.850	19.800	5.850	19.800	5.850	19.800	4.300	17.300
		n = 96	17.300	38.200	17.300	38.200	17.300	38.200	16.000	35.500
	DMEAN (S) (D) n	n = 1	6.900	23.300	6.900	23.300	6.900	23.300	5.750	21.900
		n = 96	29.400	49.900	29.400	49.900	29.400	49.900	29.200	48.600
	NEXT	---	1.000	1.100	1.000	1.100	1.000	1.100	0.980	1.400
	BREAK	---	4.700	25.000	4.700	25.000	4.700	25.000	21.300	17.900
RET	Return to original program	4.100	19.500	4.100	19.500	4.100	19.500	2.000	3.000	
	Return to other program	4.700	16.700	4.700	16.700	4.700	16.700	2.300	4.900	
FCALL Pn	Internal file pointer	5.400	5.400	5.400	5.400	5.400	5.400	3.300	5.300	
	Common pointer	7.600	30.500	7.600	30.500	7.600	30.500	4.900	6.600	
FCALL Pn (S1) to (S5)	---	50.400	62.700	50.400	62.700	50.400	62.700	19.800	23.700	
ECALL * Pn *: Program name	---	105.000	214.000	105.000	214.000	105.000	214.000	75.700	134.000	
ECALL * Pn (S1) to (S5) *: Program name	---	164.000	271.000	164.000	271.000	164.000	271.000	109.000	173.000	
EFCALL * Pn *: Program name	---	105.000	214.000	105.000	214.000	105.000	214.000	76.200	134.000	
EFCALL * Pn (S1) to (S5) *: Program name	---	164.000	271.000	164.000	271.000	164.000	271.000	90.500	170.000	
XCALL	---	5.100	6.700	5.100	6.700	5.100	6.700	3.800	6.400	



Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	COM CCOM	When selecting I/O refresh only	18.100	89.100	18.100	89.100	18.100	89.100	12.800	79.000
		When selecting CC-Link refresh only (Master station side)	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
		When selecting CC-Link refresh only (Local station side)	33.300	132.000	33.300	132.000	33.300	132.000	24.900	119.000
		When selecting MELSECNET/H refresh only (Control station side)	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
		When selecting MELSECNET/H refresh only (Normal station side)	78.600	231.000	78.600	231.000	78.600	231.000	54.000	212.000
		When selecting intelli auto refresh only	18.100	89.000	18.100	89.000	18.100	89.000	12.800	79.000
		When selecting I/O outside the group only (Input only)	15.700	71.600	15.700	71.600	15.700	71.600	8.600	76.500
		When selecting I/O outside the group only (Output only)	40.200	152.000	40.200	152.000	40.200	152.000	26.300	135.000
		When selecting I/O outside the group only (Both I/O)	45.800	153.000	45.800	153.000	45.800	153.000	26.100	135.000
		When selecting refresh of multiple CPU high speed transmission area only	—	—	—	—	—	—	—	—
		When selecting communication with peripheral device	18.200	89.000	18.200	89.000	18.200	89.000	7.250	54.300
		FIFW	Number of data points = 0	6.100	14.200	6.100	14.200	6.100	14.200	3.700
	Number of data points = 96		6.100	14.200	6.100	14.200	6.100	14.200	3.800	5.200
	FIFR	Number of data points = 0	7.500	15.600	7.500	15.600	7.500	15.600	4.400	5.800
		Number of data points = 96	37.000	45.000	37.000	45.000	37.000	45.000	33.500	35.200
	FPOP	Number of data points = 0	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
		Number of data points = 96	7.600	15.600	7.600	15.600	7.600	15.600	4.400	10.800
	FINS	Number of data points = 0	6.900	15.000	6.900	15.000	6.900	15.000	5.000	10.700
		Number of data points = 96	36.600	44.700	36.600	44.700	36.600	44.700	4.400	10.900
	FDEL	Number of data points = 0	8.000	16.100	8.000	16.100	8.000	16.100	4.900	11.300
		Number of data points = 96	37.300	45.500	37.300	45.500	37.300	45.500	34.200	35.900
	FROM n1 n2 (D) n3	n3 = 1	17.400	74.700	17.400	74.700	17.400	74.700	12.100	71.300
		n3 = 1000	406.000	498.500	406.000	498.500	406.000	498.500	402.600	495.100
	DFRO n1 n2 (D) n3	n3 = 1	19.600	85.600	19.600	85.600	19.600	85.600	14.600	81.800
		n3 = 500	406.000	498.500	406.000	498.500	406.000	498.500	402.600	495.100
	TO n1 n2 (S) n3	n3 = 1	16.400	69.600	16.400	69.600	16.400	69.600	11.700	63.400
		n3 = 1000	381.300	471.200	381.300	471.200	381.300	471.200	375.900	464.300
	DTO n1 n2 (S) n3	n3 = 1	18.600	85.100	18.600	85.100	18.600	85.100	14.200	78.500
		n3 = 500	381.300	471.200	381.300	471.200	381.300	471.200	375.900	464.300
	LEDR	No display→no display	1.500	7.100	1.500	7.100	1.500	7.100	5.100	5.100
		LED instruction execution→no display	38.900	109.000	38.900	109.000	38.900	109.000	35.700	89.200
	BINDA (S) (D)	(S) = 1	5.600	13.900	5.600	13.900	5.600	13.900	4.900	6.500
		(S) = -32768	7.800	16.200	7.800	16.200	7.800	16.200	7.200	8.700
	DBINDA (S) (D)	(S) = 1	6.200	14.500	6.200	14.500	6.200	14.500	5.700	7.100
		(S) = -2147483648	11.000	19.200	11.000	19.200	11.000	19.200	10.400	12.200

Category	Instruction	Condition (Device)	Processing Time (μs)							
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	BINHA (S) (D)	(S) = 1	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.900
		(S) = FFFF <sub>H</sub>	5.050	13.400	5.050	13.400	5.050	13.400	4.400	5.800
	DBINHA (S) (D)	(S) = 1	5.600	13.900	5.600	13.900	5.600	13.900	5.200	6.700
		(S) = FFFFFFFF <sub>H</sub>	5.600	13.900	5.600	13.900	5.600	13.900	5.100	6.500
	BCDDA (S) (D)	(S) = 1	4.850	13.200	4.850	13.200	4.850	13.200	4.300	5.800
		(S) = 9999	5.300	13.600	5.300	13.600	5.300	13.600	4.700	6.100
	DBCDDA (S) (D)	(S) = 1	5.300	13.600	5.300	13.600	5.300	13.600	4.800	6.300
		(S) = 99999999	6.200	14.500	6.200	14.500	6.200	14.500	5.600	7.100
	DABIN (S) (D)	(S) = 1	7.000	18.500	7.000	18.500	7.000	18.500	6.500	9.000
		(S) = -32768	6.950	18.500	6.950	18.500	6.950	18.500	6.300	8.900
	DDABIN (S) (D)	(S) = 1	9.450	21.000	9.450	21.000	9.450	21.000	9.400	12.000
		(S) = -2147483648	9.450	21.000	9.450	21.000	9.450	21.000	9.100	11.600
	HABIN (S) (D)	(S) = 1	5.650	17.100	5.650	17.100	5.650	17.100	4.900	7.500
		(S) = FFFF <sub>H</sub>	5.750	17.300	5.750	17.300	5.750	17.300	5.100	8.100
	DHABIN (S) (D)	(S) = 1	6.800	18.200	6.800	18.200	6.800	18.200	6.000	8.500
		(S) = FFFFFFFF <sub>H</sub>	7.100	18.600	7.100	18.600	7.100	18.600	6.300	8.900
	DABCD (S) (D)	(S) = 1	5.650	17.200	5.650	17.200	5.650	17.200	5.000	7.500
		(S) = 9999	5.700	17.200	5.700	17.200	5.700	17.200	5.000	7.500
	DDABCD (S) (D)	(S) = 1	6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
		(S) = 99999999	6.850	18.300	6.850	18.300	6.850	18.300	6.200	8.800
	COMRD	—	185.000	188.000	185.000	188.000	185.000	188.000	97.300	97.400
	LEN	1 character	4.700	16.200	4.700	16.200	4.700	16.200	4.100	6.600
		96 characters	20.600	32.900	20.600	32.900	20.600	32.900	19.800	22.400
	STR	—	9.800	36.500	9.800	36.500	9.800	36.500	6.900	14.400
	DSTR	—	12.100	40.400	12.100	40.400	12.100	40.400	10.200	20.800
	VAL	—	12.200	40.900	12.200	40.900	12.200	40.900	9.800	23.900
	DVAL	—	19.400	45.600	19.400	45.600	19.400	45.600	14.000	33.100
	ESTR	—	29.700	87.800	29.700	87.800	29.700	87.800	22.100	52.400

Category	Instruction	Condition (Device)	Processing Time ( $\mu$ s)								
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	EVAL	Decimal point format all 2-digit specification	23.900	70.400	23.900	70.400	23.900	70.400	23.300	36.500	
		Exponent format all 6-digit specification	23.700	70.300	23.700	70.300	23.700	70.300	23.300	36.400	
	ASC $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	10.200	41.800	10.200	41.800	10.200	41.800	5.600	19.700	
		n = 96	31.900	66.600	31.900	66.600	31.900	66.600	30.200	44.700	
	HEX $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	8.600	43.400	8.600	43.400	8.600	43.400	7.500	23.100	
		n = 96	77.100	115.000	77.100	115.000	77.100	115.000	37.500	53.300	
	RIGHT $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	10.900	29.600	10.900	29.600	10.900	29.600	7.600	11.400	
		n = 96	41.400	60.300	41.400	60.300	41.400	60.300	36.300	46.000	
	LEFT $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	10.600	29.300	10.600	29.300	10.600	29.300	6.500	16.100	
		n = 96	41.300	60.200	41.300	60.200	41.300	60.200	36.200	46.200	
	MIDR	—	11.700	30.600	11.700	30.600	11.700	30.600	9.500	19.100	
	MIDW	—	12.400	24.000	12.400	24.000	12.400	24.000	10.300	18.200	
	INSTR	No match		22.000	38.200	22.000	38.200	22.000	38.200	19.300	29.000
		Match	Head	13.300	29.600	13.300	29.600	13.300	29.600	10.300	20.000
	End		21.900	38.100	21.900	38.100	21.900	38.100	51.100	60.800	
	EMOD	—	11.600	24.000	11.600	24.000	11.600	24.000	10.300	15.300	
	EREXP	—	19.700	28.000	19.700	28.000	19.700	28.000	19.300	22.300	
	STRINS $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	$\text{\textcircled{S}}$ = 128 / $\text{\textcircled{D}}$ = 40 / n = 1	47.000	102.000	47.000	102.000	47.000	102.000	44.300	96.700	
		$\text{\textcircled{S}}$ = 128 / $\text{\textcircled{D}}$ = 40 / n = 48	70.100	134.000	70.100	134.000	70.100	134.000	58.800	112.000	
	STRDEL $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	$\text{\textcircled{S}}$ = 128 / $\text{\textcircled{D}}$ = 40 / n = 1	46.400	93.600	46.400	93.600	46.400	93.600	39.000	78.100	
		$\text{\textcircled{S}}$ = 128 / $\text{\textcircled{D}}$ = 40 / n = 48	44.500	70.600	44.500	70.600	44.500	70.600	36.000	69.200	
	SIN	Single precision	6.400	13.900	6.400	13.900	6.400	13.900	4.500	9.900	
	COS	Single precision	6.100	13.500	6.100	13.500	6.100	13.500	4.300	8.200	
	TAN	Single precision	8.300	15.000	8.300	15.000	8.300	15.000	5.100	7.200	
	ASIN	Single precision	7.300	15.600	7.300	15.600	7.300	15.600	6.100	13.700	
	ACOS	Single precision	8.100	16.500	8.100	16.500	8.100	16.500	6.800	11.100	
	ATAN	Single precision	5.350	12.000	5.350	12.000	5.350	12.000	4.000	6.900	
	SIND	Double precision	13.400	51.300	13.400	51.300	13.400	51.300	9.600	26.000	
	COSD	Double precision	14.700	51.700	14.700	51.700	14.700	51.700	10.000	26.900	
	TAND	Double precision	17.400	54.400	17.400	54.400	17.400	54.400	11.400	25.300	
	ASIND	Double precision	22.600	60.300	22.600	60.300	22.600	60.300	12.100	30.800	
	ACOSD	Double precision	19.700	60.000	19.700	60.000	19.700	60.000	11.700	28.000	
	ATAND	Double precision	15.000	51.800	15.000	51.800	15.000	51.800	9.700	22.000	
	RAD	Single precision	3.200	10.300	3.200	10.300	3.200	10.300	2.500	4.800	
	RADD	Double precision	5.200	43.100	5.200	43.100	5.200	43.100	4.100	16.400	
	DEG	Single precision	3.200	11.500	3.200	11.500	3.200	11.500	2.500	4.700	
	DEGD	Double precision	5.150	43.800	5.150	43.800	5.150	43.800	5.000	18.100	
	SQR	Single precision	3.900	12.300	3.900	12.300	3.900	12.300	3.500	9.300	
	SQRD	Double precision	7.000	45.700	7.000	45.700	7.000	45.700	5.700	25.400	
	EXP $\text{\textcircled{S}}$ $\text{\textcircled{D}}$	Single precision	$\text{\textcircled{S}}$ = -10	6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000
$\text{\textcircled{S}}$ = 1			6.350	13.800	6.350	13.800	6.350	13.800	4.000	13.000	

Category	Instruction	Condition (Device)		Processing Time (µs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	EXPD (S) (D)	Double precision	(S) = -10	15.800	52.700	15.800	52.700	15.800	52.700	8.800	27.600
			(S) = 1	15.400	52.500	15.400	52.500	15.400	52.500	8.500	27.300
	LOG (S) (D)	Single precision	(S) = 1	5.800	14.900	5.800	14.900	5.800	14.900	4.100	8.100
			(S) = 10	7.450	16.500	7.450	16.500	7.450	16.500	6.200	10.300
	LOGD (S) (D)	Double precision	(S) = 1	11.000	48.900	11.000	48.900	11.000	48.900	9.500	28.300
			(S) = 10	12.600	51.300	12.600	51.300	12.600	51.300	11.100	29.900
	RND		—	1.950	5.450	1.950	5.450	1.950	5.450	1.200	2.300
	SRND		—	2.750	4.550	2.750	4.550	2.750	4.550	1.400	2.400
	BSQR (S) (D)		(S) = 0	2.500	6.800	2.500	6.800	2.500	6.800	1.800	3.300
			(S) = 9999	6.400	15.500	6.400	15.500	6.400	15.500	5.100	8.800
	BDSQR (S) (D)		(S) = 0	2.600	6.050	2.600	6.050	2.600	6.050	1.900	3.700
			(S) = 99999999	8.450	17.600	8.450	17.600	8.450	17.600	7.500	10.900
	BSIN		—	11.500	32.800	11.500	32.800	11.500	32.800	8.700	20.200
	BCOS		—	10.400	32.500	10.400	32.500	10.400	32.500	7.800	14.400
	BTAN		—	12.100	33.700	12.100	33.700	12.100	33.700	9.000	17.000
	BASIN		—	13.300	32.800	13.300	32.800	13.300	32.800	12.200	15.100
	BACOS		—	13.400	33.700	13.400	33.700	13.400	33.700	13.100	14.900
	BATAN		—	12.600	31.400	12.600	31.400	12.600	31.400	11.400	15.700
	POW (S1) (S2) (D)	Single precision	(S1) = 12.3 E + 5 (S2) = 3.45 E + 0	12.200	22.100	12.200	22.100	12.200	22.100	8.950	19.500
			(S1) = 12.3 E + 5 (S2) = 3.45 E + 0	27.300	61.000	27.300	61.000	27.300	61.000	19.400	55.200
	LOG10		Single precision	8.200	16.500	8.200	16.500	8.200	16.500	5.950	14.800
	LOG10D		Double precision	15.100	48.000	15.100	48.000	15.100	48.000	12.400	46.500
	LIMIT		—	5.350	5.500	5.350	5.500	5.350	5.500	5.200	5.400
	DLIMIT		—	6.000	6.150	6.000	6.150	6.000	6.150	5.700	5.900
	BAND		—	5.450	12.400	5.450	12.400	5.450	12.400	5.400	6.300
	DBAND		—	6.050	11.900	6.050	11.900	6.050	11.900	5.800	6.900
ZONE		—	6.250	10.700	6.250	10.700	6.250	10.700	5.200	11.100	
DZONE		—	6.000	11.900	6.000	11.900	6.000	11.900	5.700	10.800	

A

Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	SCL (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	14.900	50.100	14.900	50.100	14.900	50.100	14.700	48.000
			Point No.2 < (S1) <	15.800	50.900	15.800	50.900	15.800	50.900	19.600	50.400
		SM750 = OFF	Point No.1 < (S1) <	13.900	53.100	13.900	53.100	13.900	53.100	13.700	51.000
			Point No.2 < (S1) <	16.600	56.600	16.600	56.600	16.600	56.600	20.400	56.200
	DSCL (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	13.400	52.400	13.400	52.400	13.400	52.400	12.800	50.300
			Point No.2 < (S1) <	14.200	54.100	14.200	54.100	14.200	54.100	17.300	53.500
		SM750 = OFF	Point No.1 < (S1) <	12.300	53.200	12.300	53.200	12.300	53.200	11.500	51.100
			Point No.2 < (S1) <	15.000	57.600	15.000	57.600	15.000	57.600	18.100	57.100
	SCL2 (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	14.200	53.300	14.200	53.300	14.200	53.300	13.200	51.200
			Point No.2 < (S1) <	14.900	55.000	14.900	55.000	14.900	55.000	18.000	54.500
		SM750 = OFF	Point No.1 < (S1) <	15.000	53.500	15.000	53.500	15.000	53.500	14.000	51.300
			Point No.2 < (S1) <	16.300	56.400	16.300	56.400	16.300	56.400	19.300	55.800
	DSCL2 (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	13.400	52.700	13.400	52.700	13.400	52.700	13.100	50.500
			Point No.2 < (S1) <	14.200	54.300	14.200	54.300	14.200	54.300	18.100	53.700
		SM750 = OFF	Point No.1 < (S1) <	12.300	53.200	12.300	53.200	12.300	53.200	12.100	51.000
			Point No.2 < (S1) <	15.000	57.600	15.000	57.600	15.000	57.600	18.900	57.100

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)		Processing Time (µs)							
				Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	RSET	Standard RAM		6.800	26.900	6.800	26.900	6.800	26.900	3.000	16.400
		SRAM card		—	—	—	—	—	—	3.000	16.400
	QDRSET	SRAM card to standard RAM		—	—	—	—	—	—	230.000	327.000
		Standard RAM to SRAM card		—	—	—	—	—	—	997.000	1066.000
	QCDSET	SRAM card to standard ROM		—	—	—	—	—	—	525.000	690.000
		Standard ROM to SRAM card		—	—	—	—	—	—	490.000	655.000
	DATERD	—		5.600	27.800	5.600	27.800	5.600	27.800	5.100	14.700
	DATEWR	—		7.800	42.100	7.800	42.100	7.800	42.100	7.100	23.000
	DATE +	No digit increase		14.200	41.200	14.200	41.200	14.200	41.200	6.500	13.100
		Digit increase		14.200	41.200	14.200	41.200	14.200	41.200	5.700	21.200
	DATE -	No digit increase		15.100	41.200	15.100	41.200	15.100	41.200	6.500	11.500
		Digit increase		15.100	41.200	15.100	41.200	15.100	41.200	5.700	17.200
	SECOND	—		5.800	20.500	5.800	20.500	5.800	20.500	2.600	5.900
	HOUR	—		6.200	22.500	6.200	22.500	6.200	22.500	3.000	5.300
	LDDT =	Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	8.200	25.500
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		ORDT=	When not executed		0.480		0.320		0.240		0.160
	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT <>	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400	
	Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
		In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	

Category	Instruction	Condition (Device)		Processing Time (μs)								
				Q00UCPU		Q01UCPU		Q02UCPU		Q03UCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	ANDDT<>	When not executed		0.480		0.320		0.240		0.160		
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400	
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
		ORDT<>	When not executed		0.480		0.320		0.240		0.160	
			Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
				In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
	LDDT>		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
				In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
			Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
	ANDDT>	When not executed		0.480		0.320		0.240		0.160		
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400	
			In nonconductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.200	23.400	
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
		ORDT>	When not executed		0.480		0.320		0.240		0.160	
			Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
				In nonconductive status	8.200	25.500	8.200	25.500	8.200	25.500	7.400	23.300
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
LDDT<=	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400	
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
		In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200		

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)	Processing Time (μs)								
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	ANDDT<=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		ORDT<=	When not executed		0.480		0.320		0.240		0.160
	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDDT<		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400
		In nonconductive status		8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
	ANDDT<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200
		ORDT<	When not executed		0.480		0.320		0.240		0.160
	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDDT>=	Comparison of specified date		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400
		In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.400	
	Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
		In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	



Category	Instruction	Condition (Device)		Processing Time (μs)								
				Q00UCPU		Q01UCPU		Q02UCPU		Q03UCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	ANDDT>=	When not executed		0.480		0.320		0.240		0.160		
		Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.200	23.400	
		Comparison of current date	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.200	
		ORDT>=	When not executed		0.480		0.320		0.240		0.160	
			Comparison of specified date	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
				In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.400	23.300
	Comparison of current date		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
	LDTM=		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
				In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
		In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	
	ANDTM=	When not executed		0.480		0.320		0.240		0.160		
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000	
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900	
		ORTM=	When not executed		0.480		0.320		0.240		0.160	
			Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
				In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current clock		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
LDTM<>	Comparison of specified clock		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300	
		In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300		
	Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100		
		In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100		

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)	Processing Time (μs)								
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	ANDTM<>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		ORTM<>	When not executed		0.480		0.320		0.240		0.160
	Comparison of specified clock		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current clock		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	LDTM>		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300
		In nonconductive status		8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM>	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
		ORTM>	When not executed		0.480		0.320		0.240		0.160
	Comparison of specified clock		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
	Comparison of current clock		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
LDTM<=	Comparison of specified clock		In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300	
	Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	
		In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	

Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q00UCPU		Q01UCPU		Q02UCPU		Q03UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	ANDTM<=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900	
	ORTM<=	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
	LDTM<	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100
	ANDTM<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900
	In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900	
	ORTM<	When not executed		0.480		0.320		0.240		0.160	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.200
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000
	In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.000	
LDTM>=	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300	
		In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.300	23.300	
	Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	
		In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	

Category	Instruction	Condition (Device)	Processing Time (μs)									
			Q00UJCPU		Q00UCPU		Q01UCPU		Q02UCPU			
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.		
Application instruction	ANDTM>=	When not executed		0.480		0.320		0.240		0.160		
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000	
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.600	21.900	
		ORTM>=	When not executed		0.480		0.320		0.240		0.160	
			Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
				In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	7.000	23.000
	Comparison of current clock		In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	5.900	22.100	
	S.DATERD		—		9.250	51.000	9.250	51.000	9.250	51.000	7.500	23.400
	S.DATE +		No digit increase		16.800	75.400	16.800	75.400	16.800	75.400	9.100	23.400
			Digit increase		16.800	75.400	16.800	75.400	16.800	75.400	8.900	22.200
	S.DATE -	No digit increase		17.600	75.300	17.600	75.300	17.600	75.300	9.000	22.200	
		Digit increase		16.900	75.300	16.900	75.300	16.900	75.300	9.800	22.100	
	PSTOP	—		82.200	199.000	82.200	199.000	82.200	199.000	61.400	84.500	
	POFF	—		82.600	198.000	82.600	198.000	82.600	198.000	121.000	246.000	
	PSCAN	—		83.600	200.000	83.600	200.000	83.600	200.000	126.000	232.000	
	WDT	—		2.900	12.000	2.900	12.000	2.900	12.000	1.300	3.000	
	DUTY	—		7.700	27.500	7.700	27.500	7.700	27.500	4.900	24.300	
	TIMCHK	—		5.350	24.500	5.350	24.500	5.350	24.500	7.400	23.300	
	ZRRDB	File register of standard RAM		4.100	4.200	4.100	4.200	4.100	4.200	2.400	2.600	
		File register of SRAM card		—	—	—	—	—	—	2.500	2.800	
	ZRWRB	File register of standard RAM		5.400	5.500	5.400	5.500	5.400	5.500	3.100	3.300	
File register of SRAM card		—	—	—	—	—	—	3.300	3.600			
ADRSET	—		2.400	6.650	2.400	6.650	2.400	6.650	4.200	4.900		
ZPUSH	—		9.200	20.500	9.200	20.500	9.200	20.500	6.900	14.000		
ZPOP	—		9.000	15.500	9.000	15.500	9.000	15.500	7.500	12.500		

Category	Instruction	Condition (Device)	Processing Time (μs)								
			Q00JCPU		Q00UCPU		Q01UCPU		Q02UCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	S.ZCOM	When mounting CC-Link module (Master station side)	29.400	91.700	29.400	91.700	29.400	91.700	20.600	55.000	
		When mounting CC-Link module (Local station side)	29.500	91.600	29.500	91.600	29.500	91.600	20.600	66.100	
		When mounting MELSECNET/H, CC-Link IEcontroller network module(Control station side)	79.900	214.000	79.900	214.000	79.900	214.000	102.000	180.000	
		When mounting MELSECNET/H, CC-Link IEcontroller network module(Normal station side)	79.900	214.000	79.900	214.000	79.900	214.000	29.800	102.000	
	S.RTREAD	—	9.200	57.700	9.200	57.700	9.200	57.700	6.700	33.500	
	S.RTWRITE	—	10.900	67.100	10.900	67.100	10.900	67.100	8.300	26.000	
	UNIRD n1 <sup>Ⓓ</sup> n2	n2 = 1	6.000	33.100	6.000	33.100	6.000	33.100	4.000	29.100	
		n2 = 16	16.500	43.600	16.500	43.600	16.500	43.600	12.500	37.600	
	TYPERD	—	48.50	141.30	43.50	139.90	43.40	139.80	32.40	134.20	
	TRACE	Start	174.000	174.000	174.000	174.000	174.000	174.000	96.600	103.000	
	TRACER	—	5.100	15.500	5.100	15.500	5.100	15.500	3.800	13.600	
	RBNOV <sup>Ⓢ</sup> <sup>Ⓓ</sup> n	When standard RAM is used	1 point	—	—	12.200	34.900	12.200	34.900	9.400	31.300
			1000 points	—	—	121.500	145.100	121.500	145.100	118.500	141.300
		When SRAM card is used	1 point	—	—	—	—	—	—	9.400	31.400
			1000 points	—	—	—	—	—	—	178.500	201.300
	SP.FWRITE	—	—	—	—	—	—	—	9.200	12.100	
	SP.FREAD	—	—	—	—	—	—	—	489.000	544.000	
	SP.DEVST	—	—	—	—	—	—	—	87.000	144.000	
	S.DEVLD	—	—	—	—	—	—	—	127.000	140.000	

Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q00UCPU		Q01UCPU		Q02UCPU		Q03UCPU	
				Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
Multiple CPU dedicated instruction	S.TO n1 n2 n3 n4 (D)	Writing to host CPU shared memory	n4 = 1	64.600	78.100	64.600	78.100	64.600	78.100	64.600	78.100
			n4 = 320	115.000	126.000	115.000	126.000	115.000	126.000	154.000	126.000
	TO n1 n2 (S) n3	Writing to host CPU shared memory	n3 = 1	12.700	62.200	12.700	62.200	12.700	62.200	8.300	58.200
			n3 = 320	63.500	112.300	63.500	112.300	63.500	112.300	56.200	107.800
	DTO n1 n2 (S) n3	Writing to host CPU shared memory	n3 = 1	13.500	62.300	13.500	62.300	13.500	62.300	8.600	58.300
			n3 = 320	112.900	160.800	112.900	160.800	112.900	160.800	106.800	157.300
	FROM n1 n2 (D) n3	Reading from host CPU shared memory	n3 = 1	12.100	58.700	12.100	58.700	12.100	58.700	8.400	52.600
			n3 = 320	56.000	101.700	56.000	101.700	56.000	101.700	51.700	96.600
		Reading from other CPU shared memory	n3 = 1	24.400	82.900	24.400	82.900	24.400	82.900	16.600	37.000
			n3 = 320	152.000	243.000	152.000	243.000	152.000	243.000	153.000	185.000
			n3 = 1000	418.000	518.000	418.000	518.000	418.000	518.000	432.000	485.000
		DFRO n1 n2 (D) n3	Reading from host CPU shared memory	n3 = 1	12.100	58.700	12.100	58.700	12.100	58.700	8.800
	n3 = 320			97.400	143.700	97.400	143.700	97.400	143.700	94.900	139.600
	Reading from other CPU shared memory		n3 = 1	24.800	94.200	24.800	94.200	24.800	94.200	16.600	47.300
n3 = 320			276.000	367.000	276.000	367.000	276.000	367.000	278.000	339.000	
n3 = 1000			799.000	892.000	799.000	892.000	799.000	892.000	841.000	892.000	

**Remark**

For the instructions for which a rise execution instruction (□P) is not specified, the processing time is the same as an ON execution instruction.

**Example** WORDP instruction and TOP instruction

(b) When using Q03UD(E)JCPU, Q04UD(E)HCPU, Q06UD(E)HCPU,  
Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU and Q26UD(E)HCPU

Category	Instruction	Condition (Device)	Processing Time (µs)					
			Q03UD(E)CPU		Q04/Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Sequence instruction	ANB ORB MPS MRD MPP	—	0.020		0.0095		0.0095	
	INV	When not executed	0.020		0.0095		0.0095	
		When executed	0.020		0.0095		0.0095	
	MEP MEF	When not executed	0.020		0.0095		0.0095	
		When executed	0.020		0.0095		0.0095	
	EGP EGF	When not executed	0.020		0.0095		0.0095	
		When executed	0.020		0.0095		0.0095	
	PLS	—	1.300	1.600	0.890	1.100	0.890	1.100
	PLF	—	1.500	1.600	0.940	1.200	0.940	1.200
	FF	When not executed	0.040		0.0185		0.0185	
		When executed	1.200	1.500	0.790	0.910	0.790	0.910
	DELTA	When not executed	0.040		0.0185		0.0185	
		When executed	2.800	3.600	2.400	3.200	2.400	3.200
	SFT	When not executed	0.040		0.0185		0.0185	
		When executed	1.600	3.300	1.100	2.700	1.100	2.700
	MC	—	0.040		0.0185		0.0185	
	MCR	—	0.040		0.0185		0.0185	
	FEND	Error check performed	108.000	130.000	75.800	89.300	75.800	89.300
END	No error check performed	107.000	124.000	75.800	89.800	75.800	89.800	
NOP NOPLF PAGE	—	0.020		0.0095		0.0095		

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU			
				Min.	Max.	Min.	Max.	Min.	Max.		
Basic instruction	LDE=	Single precision	In conductive status		3.700	4.700	3.300	4.300	3.300	4.300	
			In non-conductive status		3.800	5.000	3.400	4.500	3.400	4.500	
	ANDE=	Single precision	When not executed								
			When executed	In conductive status		0.060		0.0285		0.0285	
				In non-conductive status		3.300	5.800	3.000	5.100	3.000	5.100
	ORE=	Single precision	When not executed		3.500	5.600	3.000	5.200	3.000	5.200	
			When executed	In conductive status		0.060		0.0285		0.0285	
				In non-conductive status		3.600	4.500	3.200	4.200		
	LDE<>	Single precision	In conductive status		3.500	4.800	3.200	4.300	0.0285		
			In non-conductive status		4.000	4.700	3.600	4.200			
	ANDE<>	Single precision	When not executed		3.900	4.500	3.500	4.000	0.0285		
			When executed	In conductive status		0.060		0.0285			
				In non-conductive status		3.300	5.100	3.000			4.800
	ORE<>	Single precision	When not executed		3.500	5.000	3.100	4.600	0.0285		
			When executed	In conductive status		0.060		0.0285			
				In non-conductive status		3.600	6.000	3.300			5.500
	LDE>	Single precision	In conductive status		3.500	5.800	3.100	5.300	0.0285		
			In non-conductive status		3.800	5.000	3.300	4.600			
	ANDE>	Single precision	When not executed		3.700	4.900	3.300	4.400	0.0285		
			When executed	In conductive status		0.060		0.0285			
				In non-conductive status		3.500	4.700	3.100			4.200
	ORE>	Single precision	When not executed		3.600	4.500	3.100	4.000	0.0285		
			When executed	In conductive status		0.060		0.0285			
				In non-conductive status		3.600	5.100	3.300			4.600
	LDE<=	Single precision	In conductive status		3.500	4.800	3.200	4.500	0.0285		
			In non-conductive status		3.800	5.600	3.400	5.200			
	ANDE<=	Single precision	When not executed		3.800	5.600	3.400	5.100	0.0285		
			When executed	In conductive status		0.060		0.0285			
In non-conductive status				3.200	4.600	2.800	4.200				
ORE<=	Single precision	When not executed		3.500	5.000	3.100	4.500	0.0285			
		When executed	In conductive status		0.060		0.0285				
			In non-conductive status		3.700	5.800	3.400			5.400	
LDE<	Single precision	In conductive status		3.800	5.700	3.300	5.300	0.0285			
		In non-conductive status		4.000	5.400	3.500	4.900				
ANDE<	Single precision	When not executed		4.000	5.200	3.500	4.900	0.0285			
		When executed	In conductive status		0.060		0.0285				
			In non-conductive status		3.400	4.600	3.000			4.200	
ORE<	Single precision	When not executed		3.500	4.900	3.100	4.400	0.0285			
		When executed	In conductive status		0.060		0.0285				
			In non-conductive status		3.600	5.200	3.300			4.900	



Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU			
				Min.	Max.	Min.	Max.	Min.	Max.		
Basic instruction	LDE>=	Single precision	In conductive status		3.800	6.000	3.300	5.500	0.0285		
			In non-conductive status		3.800	5.900	3.400	5.400			
	ANDE>=	Single precision	When not executed		0.060		0.0285		0.0285		
			When executed	In conductive status		3.200	4.800	2.900			4.600
				In non-conductive status		3.500	5.400	3.100			5.100
	ORE>=	Single precision	When not executed		0.060		0.0285		0.0285		
			When executed	In conductive status		3.600	5.200	3.300			4.700
				In non-conductive status		3.500	5.200	3.200			4.700
	LDED=	Double precision	In conductive status		4.100	7.700	3.500	7.200	3.500	7.200	
			In non-conductive status		4.300	8.100	3.800	7.400	3.800	7.400	
	ANDED=	Double precision	When not executed								0.060
			When executed	In conductive status		3.600	7.600	3.200	7.000	3.200	7.000
				In non-conductive status		3.600	7.600	3.200	7.000	3.200	7.000
	ORED=	Double precision	When not executed		3.900	7.700	3.400	7.400	3.400	7.400	
			When executed	In conductive status		0.060		0.0285		0.0285	
				In non-conductive status		3.800	8.800	3.400	8.300	3.400	8.300
	LDED<>	Double precision	In conductive status		4.000	9.300	3.700	8.800	3.700	8.800	
			In non-conductive status		4.400	8.200	3.900	7.700	3.900	7.700	
	ANDED<>	Double precision	When not executed		4.100	7.900	3.500	7.500	3.500	7.500	
			When executed	In conductive status							0.060
In non-conductive status				0.060		0.0285		0.0285			
ORED<>	Double precision	When not executed		3.800	7.600	3.300	7.200	3.300	7.200		
		When executed	In conductive status		3.800	7.700	3.400	7.300	3.400	7.300	
			In non-conductive status		0.060		0.0285		0.0285		
LDED>	Double precision	In conductive status		4.100	9.300	3.700	8.900	3.700	8.900		
		In non-conductive status		3.800	8.900	3.400	8.400	3.400	8.400		
ANDED>	Double precision	When not executed		4.300	8.100	3.800	7.500	3.800	7.500		
		When executed	In conductive status		4.100	7.800	3.500	7.200	3.500	7.200	
			In non-conductive status								
ORED>	Double precision	When not executed		0.060		0.0285		0.0285			
		When executed	In conductive status		3.800	7.700	3.300	7.300	3.300	7.300	
			In non-conductive status		4.000	7.900	3.500	7.500	3.500	7.500	
LDED<=	Double precision	In conductive status		0.060		0.0285		0.0285			
		In non-conductive status		4.100	9.300	3.700	8.800	3.700	8.800		
ANDED<=	Double precision	When not executed		4.100	9.300	3.700	8.800	3.700	8.800		
		When executed	In conductive status		4.000	8.000	3.500	7.400	3.500	7.400	
			In non-conductive status		4.100	9.400	3.600	8.800	3.600	8.800	
ORED<=	Double precision	When not executed		0.060		0.0285		0.0285			
		When executed	In conductive status		0.060		0.0285		0.0285		
			In non-conductive status		3.800	7.700	3.300	7.200	3.300	7.200	

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)		Processing Time (μs)							
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU			
				Min.	Max.	Min.	Max.	Min.	Max.		
Basic instruction	LDED<	Double precision	In conductive status		4.300	8.300	3.800	7.600	3.800	7.600	
			In non-conductive status		3.700	7.900	3.500	7.400	3.500	7.400	
	ANDED<	Double precision	When not executed								
			When executed	In conductive status		0.060		0.0285		0.0285	
				In non-conductive status		3.800	7.800	3.300	7.300	3.300	7.300
	ORED<	Double precision	When not executed		3.900	7.900	3.400	3.900	3.400	3.900	
			When executed	In conductive status		0.060		0.0285		0.0285	
				In non-conductive status		4.100	9.600	3.700	9.200	3.700	9.200
	LDED>=	Double precision	In conductive status		4.000	9.600	3.700	9.200	3.700	9.200	
			In non-conductive status		4.100	9.600	3.600	9.000	3.600	9.000	
	ANDED>=	Double precision	When not executed		4.100	9.600	3.600	8.900	3.600	8.900	
			When executed	In conductive status		0.060		0.0285		0.0285	
				In non-conductive status		0.060		0.0285		0.0285	
	ORED>=	Double precision	When not executed		3.800	7.900	3.400	7.400	3.400	7.400	
			When executed	In conductive status		3.900	8.100	3.400	7.500	3.400	7.500
				In non-conductive status		0.060		0.0285		0.0285	
	LD\$=	In conductive status		4.100	9.600	3.700	9.200	3.700	9.200		
		In non-conductive status		4.000	7.200	3.600	6.600	3.600	6.600		
	AND\$=	When not executed		5.300	8.900	4.700	8.100	4.700	8.100		
		When executed	In conductive status		4.700	9.000	4.200	8.200	4.200	8.200	
			In non-conductive status		0.060		0.0285		0.0285		
	OR\$=	When not executed		4.400	6.800	3.900	6.400	3.900	6.400		
		When executed	In conductive status		4.500	6.700	4.000	6.300	4.000	6.300	
			In non-conductive status		0.060		0.0285		0.0285		
	LD\$<>	In conductive status		5.100	8.200	4.200	7.600	4.200	7.600		
		In non-conductive status		5.000	8.100	4.000	7.200	4.000	7.200		
	AND\$<>	When not executed		4.800	8.100	4.300	7.500	4.300	7.500		
		When executed	In conductive status		4.700	8.400	4.200	7.800	4.200	7.800	
			In non-conductive status		0.060		0.0285		0.0285		
	OR\$<>	When not executed		4.300	5.500	4.100	5.100	4.100	5.100		
When executed		In conductive status		4.500	5.900	4.400	5.400	4.400	5.400		
		In non-conductive status		0.060		0.0285		0.0285			
LD\$>	In conductive status		5.200	7.300	4.100	6.700	4.100	6.700			
	In non-conductive status		5.100	7.200	4.100	6.700	4.100	6.700			
AND\$>	When not executed		4.800	7.200	4.300	6.700	4.300	6.700			
	When executed	In conductive status		4.800	7.700	4.200	7.100	4.200	7.100		
		In non-conductive status		0.060		0.0285		0.0285			
OR\$>	When not executed		4.500	7.100	4.000	6.700	4.000	6.700			
	When executed	In conductive status		4.600	7.600	4.300	7.000	4.300	7.000		
		In non-conductive status		0.060		0.0285		0.0285			
LD\$<=	In conductive status		5.100	6.800	4.300	6.200	4.300	6.200			
	In non-conductive status		5.200	7.200	4.300	6.600	4.300	6.600			
AND\$<=	When not executed		5.000	6.300	4.400	5.700	4.400	5.700			
	When executed	In conductive status		4.800	6.400	4.200	5.800	4.200	5.800		
		In non-conductive status		0.060		0.0285		0.0285			

Category	Instruction	Condition (Device)	Processing Time (μs)						
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	
Basic instruction	OR\$<=	When not executed	0.060		0.0285		0.0285		
		When executed	In conductive status	4.700	7.700	4.400	7.200	4.400	7.200
		In non-conductive status	4.600	7.600	4.400	7.100	4.400	7.100	
	LD\$<	In conductive status	4.800	8.100	4.500	7.500	4.500	7.500	
		In non-conductive status	5.000	8.300	4.500	7.900	4.500	7.900	
	AND\$<	When not executed	0.060		0.0285		0.0285		
		When executed	In conductive status	4.500	7.100	4.000	6.600	4.000	6.600
		In non-conductive status	4.900	7.500	4.400	7.100	4.400	7.100	
	OR\$<	When not executed	0.060		0.0285		0.0285		
		When executed	In conductive status	5.100	7.800	4.100	7.200	4.100	7.200
		In non-conductive status	5.000	8.100	4.100	7.600	4.100	7.600	
	LD\$>=	In conductive status	4.800	6.700	4.500	6.200	4.500	6.200	
		In non-conductive status	5.000	6.700	4.400	6.300	4.400	6.300	
	AND\$>=	When not executed	0.060		0.0285		0.0285		
		When executed	In conductive status	4.400	6.800	4.100	6.300	4.100	6.300
		In non-conductive status	4.500	7.000	4.200	6.600	4.200	6.600	
	OR\$>=	When not executed	0.060		0.0285		0.0285		
		When executed	In conductive status	5.400	6.600	4.100	5.800	4.100	5.800
		In non-conductive status	5.300	6.300	4.100	5.700	4.100	5.700	
	BKCMP = S1 S2 D n	n = 1	8.200	10.700	7.500	10.000	7.500	10.000	
		n = 96	57.400	61.800	46.400	48.700	46.400	48.700	
	BKCMP<> S1 S2 D n	n = 1	8.200	10.700	7.500	10.000	7.500	10.000	
		n = 96	59.500	63.300	45.600	50.400	45.600	50.400	
	BKCMP> S1 S2 D n	n = 1	8.200	10.800	7.500	10.100	7.500	10.100	
		n = 96	59.500	63.400	47.700	50.500	47.700	50.500	
	BKCMP<= S1 S2 D n	n = 1	8.200	10.600	7.500	10.000	7.500	10.000	
		n = 96	57.400	61.700	46.400	49.000	46.400	49.000	
	BKCMP< S1 S2 D n	n = 1	8.300	10.600	7.500	10.000	7.500	10.000	
		n = 96	59.500	63.600	47.600	50.500	47.600	50.500	
	BKCMP>= S1 S2 D n	n = 1	8.200	10.900	7.500	10.000	7.500	10.000	
n = 96		57.400	62.000	46.400	48.900	46.400	48.900		
DBKCMP = S1 S2 D n	n = 1	9.250	14.000	8.600	13.000	8.600	13.000		
	n = 96	60.700	67.500	47.900	52.800	47.900	52.800		
DBKCMP<> S1 S2 D n	n = 1	9.250	14.000	8.600	13.000	8.600	13.000		
	n = 96	60.700	67.500	47.900	52.800	47.900	52.800		
DBKCMP> S1 S2 D n	n = 1	9.250	14.000	8.600	13.000	8.600	13.000		
	n = 96	60.700	67.500	47.900	52.800	47.900	52.800		
DBKCMP<= S1 S2 D n	n = 1	9.250	14.000	8.600	13.000	8.600	13.000		
	n = 96	60.700	67.500	47.900	52.800	47.900	52.800		
DBKCMP< S1 S2 D n	n = 1	9.250	14.000	8.600	13.000	8.600	13.000		
	n = 96	60.700	67.500	47.900	52.800	47.900	52.800		
DBKCMP>= S1 S2 D n	n = 1	9.250	14.000	8.600	13.000	8.600	13.000		
	n = 96	60.700	67.500	47.900	52.800	47.900	52.800		

Category	Instruction	Condition (Device)		Processing Time (μs)					
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	DB + (S) (D)	When executed		4.900	7.000	4.600	6.400	4.600	6.400
	DB + (S1) (S2) (D)	When executed		5.200	7.300	4.800	6.700	4.800	6.700
	DB - (S) (D)	When executed		4.900	6.600	4.700	6.000	4.700	6.000
	DB - (S1) (S2) (D)	When executed		5.200	7.500	4.800	6.600	4.800	6.600
	DB * (S1) (S2) (D)	When executed		8.300	12.100	8.100	11.600	8.100	11.600
	DB / (S1) (S2) (D)	When executed		6.100	9.100	5.800	8.800	5.800	8.800
	ED + (S) (D)	Double precision	(S) = 0, (D) = 0	4.800	8.000	4.300	7.200	4.300	7.200
			(S) = 2 <sup>1023</sup> , (D) = 2 <sup>1023</sup>	4.800	8.000	4.300	7.200	4.300	7.200
	ED + (S1) (S2) (D)	Double precision	(S1) = 0, (S2) = 0	5.500	9.800	4.800	9.200	4.800	9.200
			(S1) = 2 <sup>1023</sup> , (S2) = 2 <sup>1023</sup>	5.500	9.800	4.800	9.200	4.800	9.200
	ED - (S) (D)	Double precision	(S) = 0, (D) = 0	5.000	8.200	4.400	7.500	4.400	7.500
			(S) = 2 <sup>1023</sup> , (D) = 2 <sup>1023</sup>	5.000	8.200	4.400	7.500	4.400	7.500
	ED - (S1) (S2) (D)	Double precision	(S1) = 0, (S2) = 0	4.400	8.100	3.800	7.500	3.800	7.500
			(S1) = 2 <sup>1023</sup> , (S2) = 2 <sup>1023</sup>	4.400	8.100	3.800	7.500	3.800	7.500
	ED * (S1) (S2) (D)	Double precision	(S1) = 0, (S2) = 0	5.800	9.500	5.100	8.800	5.100	8.800
			(S1) = 2 <sup>1023</sup> , (S2) = 2 <sup>1023</sup>	5.800	9.500	5.100	8.800	5.100	8.800
	ED / (S1) (S2) (D)	Double precision	(S1) = 2 <sup>1023</sup> , (S2) = 2 <sup>1023</sup>	6.600	10.600	5.900	10.000	5.900	10.000
	BK + (S1) (S2) (D) n	n = 1		9.100	11.200	8.500	10.600	8.500	10.600
		n = 96		60.700	62.900	44.600	47.000	44.600	47.000
	BK - (S1) (S2) (D) n	n = 1		9.700	12.000	8.900	11.300	8.900	11.300
		n = 96		61.300	63.600	45.600	47.900	45.600	47.900
	DBK + (S1) (S2) (D) n	n = 1		7.000	10.700	6.450	9.950	6.450	9.950
		n = 96		59.400	63.100	43.700	47.500	43.700	47.500
	DBK - (S1) (S2) (D) n	n = 1		7.000	10.700	6.450	9.950	6.450	9.950
		n = 96		59.400	63.100	43.700	47.500	43.700	47.500
	\$ + (S) (D)	—		8.800	14.600	8.100	13.900	8.100	13.900
	\$ + (S1) (S2) (D)	—		7.300	11.100	6.500	10.300	6.500	10.300
	FLTD	Double precision	(S) = 0	2.300	5.000	1.800	4.700	1.800	4.700
			(S) = 7FFFH	2.500	5.200	2.200	4.800	2.200	4.800
	DFLTD	Double precision	(S) = 0	2.400	5.200	2.000	4.900	2.000	4.900
(S) = 7FFFFFFFH			2.700	5.400	2.300	5.100	2.300	5.100	
INTD	Double precision	(S) = 0	2.700	4.100	2.200	4.100	2.200	4.100	
		(S) = 32766.5	3.700	5.900	3.200	5.600	3.200	5.600	
DINTD	Double precision	(S) = 0	2.600	3.900	2.200	3.400	2.200	3.400	
		(S) = 1234567890.3	3.400	5.600	3.000	5.100	3.000	5.100	

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Basic instruction	DBL	When executed	2.700	3.400	2.300	2.700	2.300	2.700
	WORD	When executed	2.900	4.300	2.600	3.600	2.600	3.600
	GRY	When executed	2.700	3.900	2.300	3.400	2.300	3.400
	DGRY	When executed	2.900	3.500	2.500	3.000	2.500	3.000
	GBIN	When executed	4.000	4.800	3.800	4.300	3.800	4.300
	DGBIN	When executed	5.500	6.100	5.000	5.900	5.000	5.900
	NEG	When executed	2.400	3.900	2.000	3.300	2.000	3.300
	DNEG	When executed	2.500	3.700	2.500	3.300	2.500	3.300
	ENEG	Floating point = 0	2.500	3.300	2.300	2.800	2.300	2.800
		Floating point = -1.0	2.700	4.500	2.500	3.900	2.500	3.900
	EDNEG	Floating point = 0	2.200	3.500	1.800	3.100	1.800	3.100
		Floating point = -1.0	2.400	3.500	1.900	3.000	1.900	3.000
	BKBCD $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	6.600	8.900	5.900	8.200	5.900	8.200
		n = 96	71.300	74.100	61.000	63.400	61.000	63.400
	BKBIN $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 1	6.500	9.800	5.600	9.300	5.600	9.300
		n = 96	56.300	59.500	49.200	52.500	49.200	52.500
	ECON	—	2.600	5.400	2.100	4.500	2.100	4.500
	EDCON	—	2.800	5.400	2.500	5.400	2.500	5.400
	EDMOV	—	2.300	5.500	1.700	5.000	1.700	5.000
	\$MOV	Character string to be transferred = 0	4.000	6.300	3.400	5.600	3.400	5.600
		Character string to be transferred = 32	14.600	16.500	11.400	13.300	11.400	13.300
	BXCH $\text{\textcircled{D1}}$ $\text{\textcircled{D2}}$ n	n = 1	6.200	7.900	5.500	7.300	5.500	7.300
		n = 96	67.000	68.800	47.300	49.300	47.300	49.300
	SWAP	—	2.400	2.700	1.900	2.200	1.900	2.200
	GOEND	—	0.500		0.500		0.500	
	DI	—	1.800	2.200	1.500	1.800	1.500	1.800
	EI	—	3.100	3.800	3.000	3.300	3.000	3.300
	IMASK	—	9.800	13.300	7.200	10.500	7.200	10.500
	IRET	—	1.000		1.000		1.000	
	RSF X n	n = 1	4.200	5.900	3.700	5.600	3.700	5.600
		n = 96	11.400	13.800	10.700	12.400	10.700	12.400
	RSF Y n	n = 1	3.800	4.800	3.400	4.800	3.400	4.800
		n = 96	8.500	9.500	8.100	8.900	8.100	8.900
	UDCNT1	—	0.900	1.500	0.500	0.983	0.500	0.983
	UDCNT2	—	0.900	1.700	0.600	1.300	0.600	1.300
	TTMR	—	3.900	6.100	3.400	5.400	3.400	5.400
STMR	—	6.800	13.500	5.800	12.500	5.800	12.500	
ROTC	—	9.000	10.500	8.000	9.400	8.000	9.400	
RAMP	—	5.900	8.800	5.200	8.400	5.200	8.400	
SPD	—	0.900	1.900	0.500	1.400	0.500	1.400	
PLSY	—	1.900	2.200	1.500	1.800	1.500	1.800	
PWM	—	1.200	1.600	0.900	1.200	0.900	1.200	
MTR	—	10.400	19.800	9.400	10.000	9.400	10.000	

Category	Instruction	Condition (Device)	Processing Time (µs)						
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	BKAND $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1	9.000	11.700	8.300	11.000	8.300	11.000	
		n = 96	57.400	63.100	43.800	47.300	43.800	47.300	
	BKOR $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1	7.700	10.000	7.700	9.500	7.700	9.500	
		n = 96	57.400	61.900	44.300	45.800	44.300	45.800	
	BKXOR $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1	7.800	10.100	7.300	9.200	7.300	9.200	
		n = 96	57.300	61.500	43.800	45.800	43.800	45.800	
	BKXNR $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1	7.800	9.600	7.600	8.900	7.600	8.900	
		n = 96	57.400	61.400	43.900	45.300	43.900	45.300	
	BSFR $\text{\textcircled{D}}$ n	n = 1	3.700	5.400	3.200	4.800	3.200	4.800	
		n = 96	6.900	9.000	5.800	7.700	5.800	7.700	
	BSFL $\text{\textcircled{D}}$ n	n = 1	4.100	5.900	3.400	5.100	3.400	5.100	
		n = 96	7.100	9.100	6.000	7.900	6.000	7.900	
	SFTBR $\text{\textcircled{D}}$ n1 n2	n1 = 16 / n2 = 1	7.950	17.500	7.600	16.900	7.600	16.900	
		n1 = 16 / n2 = 15	7.950	17.500	7.550	16.900	7.550	16.900	
	SFTBL $\text{\textcircled{D}}$ n1 n2	n1 = 16 / n2 = 1	7.950	17.900	7.500	17.400	7.500	17.400	
		n1 = 16 / n2 = 15	7.900	17.800	7.500	17.300	7.500	17.300	
	SFTWR $\text{\textcircled{D}}$ n1 n2	n1 = 16 / n2 = 1	5.950	10.600	4.600	8.700	4.600	8.700	
		n1 = 16 / n2 = 15	5.900	10.600	4.600	8.700	4.600	8.700	
	SFTWL $\text{\textcircled{D}}$ n1 n2	n1 = 16 / n2 = 1	5.950	10.700	4.550	8.700	4.550	8.700	
		n1 = 16 / n2 = 15	5.950	10.700	4.600	8.800	4.600	8.800	
	BSET $\text{\textcircled{D}}$ n	n = 1	3.000	3.400	2.500	2.800	2.500	2.800	
		n = 15	3.000	3.500	2.500	2.800	2.500	2.800	
	BRST $\text{\textcircled{D}}$ n	n = 1	3.000	3.400	2.600	2.800	2.600	2.800	
		n = 15	3.000	3.400	2.500	2.800	2.500	2.800	
	TEST	When executed	4.400	5.300	3.700	4.700	3.700	4.700	
	DTEST	When executed	4.500	5.400	3.900	4.800	3.900	4.800	
	BKRST $\text{\textcircled{D}}$ n	n = 1	4.300	4.600	3.700	4.100	3.700	4.100	
		n = 96	6.000	6.800	5.100	6.000	5.100	6.000	
	SER $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1	All match	4.900	5.300	4.200	4.600	4.200	4.600
			None match	5.000	5.300	4.200	4.600	4.200	4.600
		n = 96	All match	32.300	32.900	25.900	26.300	25.900	26.300
			None match	32.400	32.900	25.900	26.300	25.900	26.300
DSER $\text{\textcircled{S1}}$ $\text{\textcircled{S2}}$ $\text{\textcircled{D}}$ n	n = 1	All match	6.100	6.500	5.400	5.700	5.400	5.700	
		None match	6.200	6.600	5.500	5.900	5.500	5.900	
	n = 96	All match	52.800	54.200	41.200	41.800	41.200	41.800	
		None match	52.800	54.200	41.200	41.800	41.200	41.800	
DSUM $\text{\textcircled{S}}$ $\text{\textcircled{D}}$	$\text{\textcircled{S}} = 0$	3.700	4.100	3.300	3.600	3.300	3.600		
	$\text{\textcircled{S}} = \text{FFFFFFFF}_H$	3.800	4.100	3.200	3.700	3.200	3.700		
DECO $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 2	6.000	7.500	5.300	6.900	5.300	6.900		
	n = 8	8.100	9.300	6.800	7.800	6.800	7.800		
ENCO $\text{\textcircled{S}}$ $\text{\textcircled{D}}$ n	n = 2	M1 = ON	5.300	5.700	4.700	5.100	4.700	5.100	
		M4 = ON	5.200	5.700	4.600	5.000	4.600	5.000	
	n = 8	M1 = ON	10.400	11.400	9.000	10.000	9.000	10.000	
		M256 = ON	5.700	6.800	5.100	6.100	5.100	6.100	

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	DIS (S) (D) n	n = 1	4.400	5.300	3.800	4.600	3.800	4.600
		n = 4	4.800	5.700	4.000	5.000	4.000	5.000
	UNI (S) (D) n	n = 1	5.000	5.300	3.500	4.800	3.500	4.800
		n = 4	5.600	6.000	4.000	5.100	4.000	5.100
	NDIS	When executed	11.000	13.100	11.000	13.200	11.000	13.200
	NUNI	When executed	10.600	12.700	7.300	13.200	7.300	13.200
	WTOB (S) (D) n	n = 1	5.000	6.500	4.400	5.800	4.400	5.800
		n = 96	36.000	38.400	28.200	29.300	28.200	29.300
	BTOW (S) (D) n	n = 1	5.100	6.100	4.600	5.500	4.600	5.500
		n = 96	29.900	32.000	22.800	23.800	22.800	23.800
	MAX (S) (D) n	n = 1	4.300	6.900	4.000	6.100	4.000	6.100
		n = 96	31.200	33.500	24.700	27.000	24.700	27.000
	MIN (S) (D) n	n = 1	4.400	6.800	4.000	6.000	4.000	6.000
		n = 96	30.300	34.800	26.500	28.300	26.500	28.300
	DMAX (S) (D) n	n = 1	4.800	9.100	4.800	8.100	4.800	8.100
		n = 96	56.400	62.200	47.100	49.600	47.100	49.600
	DMIN (S) (D) n	n = 1	4.800	6.800	4.300	5.900	4.300	5.900
		n = 96	55.400	60.200	45.400	47.400	45.400	47.400
	SORT (S1) n (S2) (D1) (D2)	n = 1	6.200	9.300	5.600	8.800	5.600	8.800
		n = 96	6.200	9.400	5.600	8.600	5.600	8.600
	DSORT (S1) n (S2) (D1) (D2)	n = 1	6.200	9.300	5.600	8.200	5.600	8.200
		n = 96	6.100	9.100	5.600	8.400	5.600	8.400
	WSUM (S) (D) n	n = 1	4.800	6.200	4.200	5.500	4.200	5.500
		n = 96	26.900	28.700	21.300	22.300	21.300	22.300
	DWSUM (S) (D) n	n = 1	5.500	7.000	4.800	6.100	4.800	6.100
		n = 96	53.000	56.300	42.700	44.000	42.700	44.000
	MEAN (S) (D) n	n = 1	4.300	8.650	3.900	7.800	3.900	7.800
		n = 96	16.000	21.400	12.900	18.000	12.900	18.000
	DMEAN (S) (D) n	n = 1	5.700	10.600	5.300	9.950	5.300	9.950
		n = 96	29.200	35.200	23.000	28.800	23.000	28.800
NEXT	—	0.940	1.400	0.770	1.200	0.770	1.200	
BREAK	—	10.400	5.500	9.100	5.000	9.100	5.000	
RET	Return to original program	2.000	3.000	1.600	2.600	1.600	2.600	
	Return to other program	2.300	3.700	2.000	3.100	2.000	3.100	
FCALL Pn	Internal file pointer	3.100	4.400	2.700	3.600	2.700	3.600	
	Common pointer	4.000	5.700	3.600	5.100	3.600	5.100	
FCALL Pn (S1) to (S5)	—	19.300	21.500	16.500	18.600	16.500	18.600	
ECALL * Pn *: Program name	—	70.300	82.300	65.900	77.600	65.900	77.600	
ECALL * Pn (S1) to (S5) *: Program name	—	101.000	114.000	91.800	105.000	91.800	105.000	
EFCALL * Pn *: Program name	—	70.700	82.800	66.200	78.100	66.200	78.100	
EFCALL * Pn (S1) to (S5) *: Program name	—	86.500	107.000	78.800	91.600	78.800	91.600	
XCALL	—	3.800	5.700	3.700	5.200	3.700	5.200	

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	COM CCOM	When selecting I/O refresh only	12.800	29.100	12.400	28.600	12.400	28.600
		When selecting CC-Link refresh only (Master station side)	16.000	39.500	15.500	39.100	15.500	39.100
		When selecting CC-Link refresh only (Local station side)	16.100	39.500	15.500	39.100	15.500	39.100
		When selecting MELSECNET/H refresh only (Control station side)	34.700	70.400	34.400	69.800	34.400	69.800
		When selecting MELSECNET/H refresh only (Normal station side)	34.700	70.400	34.400	69.800	34.400	69.800
		When selecting intelli auto refresh only	12.800	33.200	12.800	33.200	12.800	33.200
		When selecting I/O outside the group only (Input only)	7.900	21.100	7.700	20.700	7.700	20.700
		When selecting I/O outside the group only (Output only)	16.900	44.800	16.500	44.200	16.500	44.200
		When selecting I/O outside the group only (Both I/O)	22.600	52.600	22.400	52.600	22.400	52.600
		When selecting refresh of multiple CPU high speed transmission area only	13.000	33.800	12.700	33.200	12.700	33.200
		When selecting communication with peripheral device	7.250	18.800	7.100	18.500	7.100	18.500
		FIFW	Number of data points = 0	3.700	5.300	3.200	4.600	3.200
	Number of data points = 96		3.800	4.400	3.300	3.800	3.300	3.800
	FIFR	Number of data points = 0	4.300	5.000	3.800	4.400	3.800	4.400
		Number of data points = 96	33.500	35.500	24.800	25.700	24.800	25.700
	FPOP	Number of data points = 0	4.300	5.900	3.800	5.300	3.800	5.300
		Number of data points = 96	4.300	5.900	3.700	5.400	3.700	5.400
	FINS	Number of data points = 0	4.800	5.900	3.700	5.300	3.700	5.300
		Number of data points = 96	4.300	5.900	3.700	5.300	3.700	5.300
	FDEL	Number of data points = 0	4.900	6.500	4.200	5.800	4.200	5.800
		Number of data points = 96	34.200	35.900	25.400	25.900	25.400	25.900
	FROM n1 n2 (D) n3	n3 = 1	10.800	24.100	10.700	23.600	10.700	23.600
		n3 = 1000	392.600	413.300	390.900	410.200	390.900	410.200
	DFRO n1 n2 (D) n3	n3 = 1	13.600	27.700	12.600	26.700	12.600	26.700
		n3 = 500	392.600	413.300	390.900	410.200	390.900	410.200
	TO n1 n2 (S) n3	n3 = 1	10.200	21.900	9.600	21.300	9.600	21.300
		n3 = 1000	373.700	394.100	372.500	390.800	372.500	390.800
	DTO n1 n2 (S) n3	n3 = 1	13.000	26.700	12.000	25.700	12.000	25.700
		n3 = 500	373.700	394.100	372.500	390.800	372.500	390.800
	LEDR	No display→no display	2.400	2.600	1.900	2.000	1.900	2.000
		LED instruction execution→no display	28.100	39.400	24.400	35.800	24.400	35.800
	BINDA (S) (D)	(S) = 1	4.900	6.500	4.300	5.600	4.300	5.600
		(S) = -32768	7.200	8.700	6.500	8.000	6.500	8.000
	DBINDA (S) (D)	(S) = 1	5.700	7.100	4.900	6.300	4.900	6.300
		(S) = -2147483648	10.400	12.000	9.600	11.000	9.600	11.000



Category	Instruction	Condition (Device)	Processing Time (μs)					
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
			Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	BINHA (S) (D)	(S) = 1	4.400	5.900	3.800	5.200	3.800	5.200
		(S) = FFFF <sub>H</sub>	4.400	5.800	3.700	5.200	3.700	5.200
	DBINHA (S) (D)	(S) = 1	5.200	6.700	4.600	6.000	4.600	6.000
		(S) = FFFFFFFF <sub>H</sub>	5.100	6.500	4.600	6.000	4.600	6.000
	BCDDA (S) (D)	(S) = 1	4.300	5.800	3.600	5.000	3.600	5.000
		(S) = 9999	4.700	6.100	4.100	5.400	4.100	5.400
	DBCDDA (S) (D)	(S) = 1	4.800	6.300	4.000	5.500	4.000	5.500
		(S) = 99999999	5.600	7.100	4.900	6.300	4.900	6.300
	DABIN (S) (D)	(S) = 1	6.500	8.500	5.800	7.800	5.800	7.800
		(S) = -32768	6.300	8.300	5.600	7.700	5.600	7.700
	DDABIN (S) (D)	(S) = 1	9.400	11.500	8.500	10.500	8.500	10.500
		(S) = -2147483648	9.100	11.200	8.100	10.200	8.100	10.200
	HABIN (S) (D)	(S) = 1	4.900	7.100	4.400	6.400	4.400	6.400
		(S) = FFFF <sub>H</sub>	5.100	7.300	4.600	6.500	4.600	6.500
	DHABIN (S) (D)	(S) = 1	6.000	8.100	5.300	7.300	5.300	7.300
		(S) = FFFFFFFF <sub>H</sub>	6.300	8.500	5.600	7.700	5.600	7.700
	DABCD (S) (D)	(S) = 1	5.000	7.100	4.400	6.300	4.400	6.300
		(S) = 9999	5.000	7.100	4.300	6.300	4.300	6.300
	DDABCD (S) (D)	(S) = 1	6.200	8.300	5.500	7.400	5.500	7.400
		(S) = 99999999	6.200	8.300	5.500	7.500	5.500	7.500
	COMRD	—	51.600	52.400	50.900	51.200	50.900	51.200
	LEN	1 character	4.100	6.200	3.600	5.500	3.600	5.500
		96 characters	19.800	22.200	16.800	18.700	16.800	18.700
STR	—	6.900	11.100	6.600	10.400	6.600	10.400	
DSTR	—	10.200	12.500	9.600	11.500	9.600	11.500	
VAL	—	9.800	14.200	8.900	13.000	8.900	13.000	
DVAL	—	14.000	18.700	12.700	16.800	12.700	16.800	
ESTR	—	18.700	24.100	17.900	23.100	17.900	23.100	

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

Category	Instruction	Condition (Device)	Processing Time (μs)						
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	EVAL	Decimal point format all 2-digit specification	23.300	30.400	22.800	29.000	22.800	29.000	
		Exponent format all 6-digit specification	23.300	30.500	22.500	29.000	22.500	29.000	
	ASC (S) (D) n	n = 1	5.600	9.000	5.400	8.300	5.400	8.300	
		n = 96	28.700	32.100	25.200	28.400	25.200	28.400	
	HEX (S) (D) n	n = 1	6.000	9.700	5.400	9.000	5.400	9.000	
		n = 96	35.600	39.800	31.300	35.000	31.300	35.000	
	RIGHT (S) (D) n	n = 1	7.600	9.400	7.300	6.600	7.300	6.600	
		n = 96	36.300	40.000	29.200	31.600	29.200	31.600	
	LEFT (S) (D) n	n = 1	6.500	8.900	5.900	8.200	5.900	8.200	
		n = 96	36.200	39.700	29.200	31.500	29.200	31.500	
	MIDR	—	9.500	12.100	8.100	10.300	8.100	10.300	
	MIDW	—	10.300	12.000	8.800	10.200	8.800	10.200	
	INSTR	No match		19.300	21.800	16.600	18.400	16.600	18.400
		Match	Head	10.300	12.800	9.100	10.900	9.100	10.900
			End	51.100	54.200	42.700	44.900	42.700	44.900
	EMOD	—	10.300	11.800	9.600	11.000	9.600	11.000	
	EREXP	—	19.300	21.000	18.800	20.100	18.800	20.100	
	STRINS (S) (D) n	(S) = 128 / (D) = 40 / n = 1	41.100	54.200	35.300	47.600	35.300	47.600	
		(S) = 128 / (D) = 40 / n = 48	56.700	81.400	48.600	61.700	48.600	61.700	
	STRDEL (S) (D) n	(S) = 128 / (D) = 40 / n = 1	39.000	49.500	34.800	44.600	34.800	44.600	
		(S) = 128 / (D) = 40 / n = 48	36.000	45.200	29.200	38.100	29.200	38.100	
	SIN	Single precision	4.500	6.200	4.100	5.700	4.100	5.700	
	COS	Single precision	4.300	6.000	4.000	5.600	4.000	5.600	
	TAN	Single precision	5.100	7.200	5.100	6.700	5.100	6.700	
	ASIN	Single precision	6.100	8.900	5.900	8.500	5.900	8.500	
	ACOS	Single precision	6.800	9.300	6.700	8.900	6.700	8.900	
	ATAN	Single precision	4.000	6.500	3.900	6.000	3.900	6.000	
	SIND	Double precision	8.800	14.300	8.500	13.800	8.500	13.800	
	COSD	Double precision	9.300	15.100	8.800	14.600	8.800	14.600	
	TAND	Double precision	11.200	16.900	10.800	16.500	10.800	16.500	
	ASIND	Double precision	12.000	17.100	11.600	16.600	11.600	16.600	
	ACOSD	Double precision	11.700	16.500	11.200	16.200	11.200	16.200	
	ATAND	Double precision	9.500	14.200	9.100	13.800	9.100	13.800	
RAD	Single precision	2.500	4.800	2.100	4.300	2.100	4.300		
RADD	Double precision	4.000	9.600	3.600	9.200	3.600	9.200		
DEG	Single precision	2.500	4.700	2.200	4.400	2.200	4.400		
DEGD	Double precision	4.300	9.000	3.800	9.000	3.800	9.000		
SQR	Single precision	3.000	4.600	2.600	4.300	2.600	4.300		
SQRD	Double precision	5.600	11.500	5.200	11.000	5.200	11.000		

Category	Instruction	Condition (Device)		Processing Time (μs)					
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	EXP (S) (D)	Single precision	(S) = -10	4.000	6.100	3.800	5.500	3.800	5.500
			(S) = 1	4.000	6.100	3.800	5.600	3.800	5.600
	EXPD (S) (D)	Double precision	(S) = -10	8.700	13.900	8.200	13.500	8.200	13.500
			(S) = 1	8.400	13.600	8.000	13.200	8.000	13.200
	LOG (S) (D)	Single precision	(S) = 1	4.100	6.900	3.800	6.400	3.800	6.400
			(S) = 10	5.600	8.200	5.200	7.700	5.200	7.700
	LOGD (S) (D)	Double precision	(S) = 1	8.100	13.000	7.700	12.500	7.700	12.500
			(S) = 10	9.700	14.800	9.200	14.300	9.200	14.300
	RND	—		1.200	2.300	0.800	1.800	0.800	1.800
	SRND	—		1.400	2.400	1.100	2.000	1.100	2.000
	BSQR (S) (D)	(S) = 0		1.800	3.300	1.600	2.800	1.600	2.800
		(S) = 9999		5.100	8.800	5.100	8.000	5.100	8.000
	BDSQR (S) (D)	(S) = 0		1.900	3.400	1.500	3.000	1.500	3.000
		(S) = 99999999		7.500	10.200	7.500	9.900	7.500	9.900
	BSIN	—		8.600	15.100	8.100	14.500	8.100	14.500
	BCOS	—		7.800	14.400	7.800	13.700	7.800	13.700
	BTAN	—		9.000	13.800	9.000	13.300	9.000	13.300
	BASIN	—		10.600	13.400	10.100	12.800	10.100	12.800
	BACOS	—		11.600	14.400	11.100	14.100	11.100	14.100
	BATAN	—		9.800	11.700	9.100	10.900	9.100	10.900
	POW (S1) (S2) (D)	Single precision	(S1) = 12.3 E + 5	8.750	11.400	8.400	10.900	8.400	10.900
			(S2) = 3.45 E + 0						
	POWD (S1) (S2) (D)	Double precision	(S1) = 12.3 E + 5	8.750	11.400	8.400	10.900	8.400	10.900
			(S2) = 3.45 E + 0						
	LOG10	Single precision		18.600	27.200	18.200	26.500	18.200	26.500
	LOG10D	Double precision							
	LIMIT	—		5.900	8.550	5.700	8.050	5.700	8.050
	DLIMIT	—		11.500	19.400	11.100	18.600	11.100	18.600
BAND	—		2.800	3.100	2.400	2.700	2.400	2.700	
DBAND	—		3.200	3.500	2.800	3.000	2.800	3.000	
ZONE	—		3.000	4.300	2.700	3.800	2.700	3.800	
DZONE	—		3.600	5.100	3.300	4.600	3.300	4.600	

Category	Instruction	Condition (Device)		Processing Time (μs)					
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	SCL (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	13.200	23.600	12.300	22.500	12.300	22.500
			Point No.2 < (S1) <						
		SM750 = OFF	Point No.9 < (S1) <	13.300	23.600	12.600	22.700	12.600	22.700
			Point No.10 < (S1) <						
	DSCL (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	12.800	23.800	11.900	23.000	11.900	23.000
			Point No.2 < (S1) <						
		SM750 = OFF	Point No.9 < (S1) <	12.900	23.900	12.100	23.000	12.100	23.000
			Point No.10 < (S1) <						
	SCL2 (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	12.700	24.200	11.900	23.300	11.900	23.300
			Point No.2 < (S1) <						
		SM750 = OFF	Point No.9 < (S1) <	12.900	24.600	12.100	23.300	12.100	23.300
			Point No.10 < (S1) <						
	DSCL2 (S1) (S2) (D)	SM750 = ON	Point No.1 < (S1) <	12.600	23.800	11.800	22.900	11.800	22.900
			Point No.2 < (S1) <						
		SM750 = OFF	Point No.9 < (S1) <	13.000	23.900	12.200	22.800	12.200	22.800
			Point No.10 < (S1) <						

Category	Instruction	Condition (Device)		Processing Time (μs)					
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	RSET	Standard RAM		3.000	6.300	2.700	5.900	2.700	5.900
		SRAM card		3.000	6.400	2.600	5.800	2.600	5.800
	QDRSET	SRAM card to standard RAM		120.000	134.000	115.000	134.000	115.000	134.000
		Standard RAM to SRAM card		533.000	560.000	520.000	553.000	520.000	553.000
	QCDSET	SRAM card to standard ROM		306.000	346.000	305.000	346.000	305.000	346.000
		Standard ROM to SRAM card		311.000	342.000	300.000	334.000	300.000	334.000
	DATERD	—		3.200	5.000	2.500	4.200	2.500	4.200
	DATEWR	—		4.900	9.700	4.100	8.900	4.100	8.900
	DATE +	No digit increase		5.100	8.000	4.700	6.600	4.700	6.600
		Digit increase		5.700	8.000	4.600	6.500	4.600	6.500
	DATE -	No digit increase		5.800	8.500	4.600	7.000	4.600	7.000
		Digit increase		5.700	7.400	4.600	6.500	4.600	6.500
	SECOND	—		2.600	3.900	2.200	3.400	2.200	3.400
	HOURL	—		2.900	4.800	2.400	4.300	2.400	4.300
	LDDT =	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900
			In nonconductive status	7.400	11.600	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700
			In nonconductive status	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT=	When not executed		0.008		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700
			In nonconductive status	7.200	11.400	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300
			In nonconductive status	5.700	9.900	5.300	9.300	5.300	9.300
		ORDT=	When not executed		0.008		0.038		0.038
	Comparison of specified date		In conductive status	7.400	11.500	6.700	10.800	6.700	10.800
			In nonconductive status	7.400	11.500	6.700	10.800	6.700	10.800
	Comparison of current date		In conductive status	5.900	10.000	5.400	9.600	5.400	9.600
			In nonconductive status	5.900	10.000	5.400	9.600	5.400	9.600
	LDDT <>		Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800
		In nonconductive status		7.400	11.600	6.800	10.900	6.800	10.900
Comparison of current date		In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	
		In nonconductive status	5.900	10.100	5.500	9.700	5.500	9.700	

Category	Instruction	Condition (Device)	Processing Time (μs)						
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	ANDDT<>	When not executed		0.008		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700
			In nonconductive status	7.200	11.400	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300
	In nonconductive status		5.700	9.900	5.300	9.300	5.300	9.300	
	ORDT<>	When not executed		0.008		0.038		0.038	
		Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800
			In nonconductive status	7.400	11.500	6.700	10.800	6.700	10.800
		Comparison of current date	In conductive status	5.900	10.000	5.400	9.600	5.400	9.600
	In nonconductive status		5.900	10.000	5.400	9.600	5.400	9.600	
	LDDT>	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900
			In nonconductive status	7.400	11.600	6.800	10.900	6.800	10.900
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700
			In nonconductive status	5.900	10.100	5.500	9.700	5.500	9.700
	ANDDT>	When not executed		0.008		0.038		0.038	
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700
			In nonconductive status	7.200	11.400	6.500	10.700	6.500	10.700
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300
	In nonconductive status		5.700	9.900	5.300	9.300	5.300	9.300	
	ORDT>	When not executed		0.008		0.038		0.038	
Comparison of specified date		In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	
		In nonconductive status	7.400	11.500	6.700	10.800	6.700	10.800	
Comparison of current date		In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	
	In nonconductive status	5.900	10.000	5.400	9.600	5.400	9.600		

Category	Instruction	Condition (Device)		Processing Time ( $\mu$ s)						
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	LDDT<=	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	
			In nonconductive status	7.400	11.600	6.800	10.900	6.800	10.900	
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	
			In nonconductive status	5.900	10.100	5.500	9.700	5.500	9.700	
	ANDDT<=	When not executed		0.008		0.038		0.038		
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	
			In nonconductive status	7.200	11.400	6.500	10.700	6.500	10.700	
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	
			In nonconductive status	5.700	9.900	5.300	9.300	5.300	9.300	
		ORDT<=	When not executed		0.008		0.038		0.038	
			Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800
				In nonconductive status	7.400	11.500	6.700	10.800	6.700	10.800
	Comparison of current date		In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	
			In nonconductive status	5.900	10.000	5.400	9.600	5.400	9.600	
	LDDT<		Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900
				In nonconductive status	7.400	11.600	6.800	10.900	6.800	10.900
			Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700
		In nonconductive status		5.900	10.100	5.500	9.700	5.500	9.700	
	ANDDT<	When not executed		0.008		0.038		0.038		
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	
			In nonconductive status	7.200	11.400	6.500	10.700	6.500	10.700	
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	
			In nonconductive status	5.700	9.900	5.300	9.300	5.300	9.300	
		ORDT<	When not executed		0.008		0.038		0.038	
Comparison of specified date			In conductive status	7.400	11.500	6.700	10.800	6.700	10.800	
			In nonconductive status	7.400	11.500	6.700	10.800	6.700	10.800	
Comparison of current date	In conductive status		5.900	10.000	5.400	9.600	5.400	9.600		
	In nonconductive status	5.900	10.000	5.400	9.600	5.400	9.600			

Category	Instruction	Condition (Device)		Processing Time (μs)						
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	LDDT>=	Comparison of specified date	In conductive status	7.400	11.400	6.800	10.900	6.800	10.900	
			In nonconductive status	7.400	11.600	6.800	10.900	6.800	10.900	
		Comparison of current date	In conductive status	5.900	10.000	5.500	9.700	5.500	9.700	
			In nonconductive status	5.900	10.100	5.500	9.700	5.500	9.700	
	ANDDT>=	When not executed		0.008		0.038		0.038		
		Comparison of specified date	In conductive status	7.200	11.400	6.500	10.700	6.500	10.700	
			In nonconductive status	7.200	11.400	6.500	10.700	6.500	10.700	
		Comparison of current date	In conductive status	5.700	9.900	5.300	9.300	5.300	9.300	
			In nonconductive status	5.700	9.900	5.300	9.300	5.300	9.300	
		ORDT>=	When not executed		0.008		0.038		0.038	
			Comparison of specified date	In conductive status	7.400	11.500	6.700	10.800	6.700	10.800
				In nonconductive status	7.400	11.500	6.700	10.800	6.700	10.800
	Comparison of current date		In conductive status	5.900	10.000	5.400	9.600	5.400	9.600	
			In nonconductive status	5.900	10.000	5.400	9.600	5.400	9.600	
	LDTM=		Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800
				In nonconductive status	7.300	11.500	6.700	10.800	6.700	10.800
			Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500
		In nonconductive status		5.800	9.900	5.400	9.500	5.400	9.500	
	ANDTM=	When not executed		0.008		0.038		0.038		
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	
			In nonconductive status	7.000	11.500	6.300	10.800	6.300	10.800	
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	
			In nonconductive status	5.500	9.900	5.100	9.500	5.100	9.500	
		ORTM=	When not executed		0.008		0.038		0.038	
			Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800
				In nonconductive status	7.300	11.500	6.600	10.800	6.600	10.800
	Comparison of current clock		In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	



Category	Instruction	Condition (Device)		Processing Time (μs)						
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	LDTM<>	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	
			In nonconductive status	7.300	11.500	6.700	10.800	6.700	10.800	
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	
			In nonconductive status	5.800	9.900	5.400	9.500	5.400	9.500	
	ANDTM<>	When not executed		0.008		0.038		0.038		
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	
			In nonconductive status	7.000	11.500	6.300	10.800	6.300	10.800	
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	
			In nonconductive status	5.500	9.900	5.100	9.500	5.100	9.500	
		ORTM<>	When not executed		0.008		0.038		0.038	
			Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800
				In nonconductive status	7.300	11.500	6.600	10.800	6.600	10.800
	Comparison of current clock		In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	
			In nonconductive status	5.900	9.900	5.300	9.500	5.300	9.500	
	LDTM>		Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800
				In nonconductive status	7.300	11.500	6.700	10.800	6.700	10.800
			Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500
		In nonconductive status		5.800	9.900	5.400	9.500	5.400	9.500	
	ANDTM>	When not executed		0.008		0.038		0.038		
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	
			In nonconductive status	7.000	11.500	6.300	10.800	6.300	10.800	
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	
			In nonconductive status	5.500	9.900	5.100	9.500	5.100	9.500	
		ORTM>	When not executed		0.008		0.038		0.038	
			Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800
				In nonconductive status	7.300	11.500	6.600	10.800	6.600	10.800
	Comparison of current clock		In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	
		In nonconductive status	5.900	9.900	5.300	9.500	5.300	9.500		

Category	Instruction	Condition (Device)		Processing Time (μs)						
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	LDTM<=	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800	
			In nonconductive status	7.300	11.500	6.700	10.800	6.700	10.800	
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500	
			In nonconductive status	5.800	9.900	5.400	9.500	5.400	9.500	
	ANDTM<=	When not executed		0.008		0.038		0.038		
		Comparison of specified clock	In conductive status	7.000	11.500	6.300	10.800	6.300	10.800	
			In nonconductive status	7.000	11.500	6.300	10.800	6.300	10.800	
		Comparison of current clock	In conductive status	5.500	9.900	5.100	9.500	5.100	9.500	
			In nonconductive status	5.500	9.900	5.100	9.500	5.100	9.500	
		ORTM<=	When not executed		0.008		0.038		0.038	
			Comparison of specified clock	In conductive status	7.300	11.500	6.600	10.800	6.600	10.800
				In nonconductive status	7.300	11.500	6.600	10.800	6.600	10.800
	Comparison of current clock		In conductive status	5.900	9.900	5.300	9.500	5.300	9.500	
			In nonconductive status	5.900	9.900	5.300	9.500	5.300	9.500	
	LDTM<		Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800
				In nonconductive status	7.300	11.500	6.700	10.800	6.700	10.800
			Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500
		In nonconductive status		5.800	9.900	5.400	9.500	5.400	9.500	
	ANDTM<	When not executed		0.480		0.320		0.240		
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100	
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100	
		ORTM<	When not executed		0.480		0.320		0.240	
Comparison of specified clock			In conductive status	8.200	25.500	8.200	25.500	6.500	25.500	
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500	
Comparison of current clock	In conductive status		6.500	23.100	6.500	23.100	6.500	23.100		
	In nonconductive status		6.500	23.100	6.500	23.100	6.500	23.100		

Category	Instruction	Condition (Device)		Processing Time (μs)					
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU	
				Min.	Max.	Min.	Max.	Min.	Max.
Application instruction	LDTM<	Comparison of specified clock	In conductive status	7.300	11.500	6.700	10.800	6.700	10.800
			In nonconductive status	7.300	11.500	6.700	10.800	6.700	10.800
		Comparison of current clock	In conductive status	5.800	9.900	5.400	9.500	5.400	9.500
			In nonconductive status	5.800	9.900	5.400	9.500	5.400	9.500
	ANDTM<	When not executed		0.480		0.320		0.240	
		Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100
		When not executed		0.480		0.320		0.240	
	ORTM<	Comparison of specified clock	In conductive status	8.200	25.500	8.200	25.500	6.500	25.500
			In nonconductive status	8.200	25.500	8.200	25.500	6.500	25.500
		Comparison of current clock	In conductive status	6.500	23.100	6.500	23.100	6.500	23.100
			In nonconductive status	6.500	23.100	6.500	23.100	6.500	23.100
	S.DATERD	—		9.250	51.000	9.250	51.000	9.250	51.000
	S.DATE +	No digit increase		16.800	75.400	16.800	75.400	16.800	75.400
		Digit increase		16.800	75.400	16.800	75.400	16.800	75.400
	S.DATE -	No digit increase		17.600	75.300	17.600	75.300	17.600	75.300
		Digit increase		16.900	75.300	16.900	75.300	16.900	75.300
	PSTOP	—		82.200	199.000	82.200	199.000	82.200	199.000
	POFF	—		82.600	198.000	82.600	198.000	82.600	198.000
	PSCAN	—		83.600	200.000	83.600	200.000	83.600	200.000
	WDT	—		2.900	12.000	2.900	12.000	2.900	12.000
	DUTY	—		7.700	27.500	7.700	27.500	7.700	27.500
	TIMCHK	—		5.350	24.500	5.350	24.500	5.350	24.500
	ZRRDB	File register of standard RAM		4.100	4.200	4.100	4.200	4.100	4.200
File register of SRAM card		—	—	—	—	—	—		
ZRWRB	File register of standard RAM		5.400	5.500	5.400	5.500	5.400	5.500	
	File register of SRAM card		—	—	—	—	—	—	
ADRSET	—		2.400	6.650	2.400	6.650	2.400	6.650	
ZPUSH	—		9.200	20.500	9.200	20.500	9.200	20.500	
ZPOP	—		9.000	15.500	9.000	15.500	9.000	15.500	

Category	Instruction	Condition (Device)	Processing Time (μs)						
			Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
			Min.	Max.	Min.	Max.	Min.	Max.	
Application instruction	S.ZCOM	When mounting CC-Link module (Master station side)	19.600	26.500	19.300	26.000	19.300	26.000	
		When mounting CC-Link module (Local station side)	19.600	26.500	19.100	26.200	19.100	26.200	
		When mounting MELSECNET/H, CC-Link IEcontroller network module(Control station side)	53.500	73.500	53.000	72.700	53.000	72.700	
		When mounting MELSECNET/H, CC-Link IEcontroller network module(Normal station side)	29.800	41.200	29.800	40.600	29.800	40.600	
	S.RTREAD	—	5.900	11.000	5.400	10.500	5.400	10.500	
	S.RTWRITE	—	6.700	11.100	6.000	10.400	6.000	10.400	
	UNIRD n1 <sup>Ⓧ</sup> n2	n2 = 1	4.000	8.400	3.700	8.000	3.700	8.000	
		n2 = 16	12.500	17.000	12.200	16.600	12.200	16.600	
	TYPERD	—	29.800	53.000	29.500	52.300	29.500	52.300	
	TRACE	Start	46.600	48.300	43.800	44.700	43.800	44.700	
	TRACER	—	3.300	6.800	2.600	6.000	2.600	6.000	
	RBNOV <sup>Ⓧ</sup> <sup>Ⓧ</sup> n	When standard RAM is used	1 point	11.300	16.800	9.200	15.100	9.200	15.100
			1000 points	120.700	127.100	61.000	68.600	61.000	68.600
		When SRAM card is used	1 point	11.200	16.700	9.400	15.600	9.400	15.600
			1000 points	180.700	187.100	165.000	172.600	165.000	172.600
	SP.FWRITE	—	6.700	11.100	6.000	10.400	6.000	10.400	
	SP.FREAD	—	5.900	11.000	5.400	10.500	5.400	10.500	
SP.DEVST	—	4.500	36.500	4.000	34.500	4.000	34.500		
S.DEVLD	—	11.000	17.800	10.000	17.000	10.000	17.000		

Category	Instruction	Condition (Device)		Processing Time (μs)						
				Q03UD(E)CPU		Q04/ Q06UD(E)HCPU		Q10/Q13/Q20/ Q26UD(E)HCPU		
				Min.	Max.	Min.	Max.	Min.	Max.	
Multiple CPU dedicated instruction	S.TO n1 n2 n3 n4 (D)	Writing to host CPU shared memory	n4 = 1	34.700	34.900	33.500	34.400	33.500	34.400	
			n4 = 320	85.900	87.600	75.200	75.500	75.200	75.500	
	TO n1 n2 (S) n3	Writing to host CPU shared memory	n3 = 1	4.700	23.800	5.200	23.300	5.200	23.300	
			n3 = 320	57.500	76.200	47.100	64.500	47.100	64.500	
	DTO n1 n2 (S) n3	Writing to host CPU shared memory	n3 = 1	5.300	23.800	5.800	23.300	5.800	23.300	
			n3 = 320	111.300	128.400	91.500	108.500	91.500	108.500	
	FROM n1 n2 (D) n3	Reading from host CPU shared memory	n3 = 1	5.000	23.800	4.300	23.300	4.300	23.300	
			n3 = 320	51.400	65.600	44.400	60.700	44.400	60.700	
			Reading from other CPU shared memory	n3 = 1	11.600	17.700	10.600	13.900	10.600	13.900
				n3 = 320	142.000	160.000	142.000	149.000	142.000	149.000
	DFRO n1 n2 (D) n3	Reading from host CPU shared memory	n3 = 1	5.200	23.800	5.600	23.300	5.600	23.300	
			n3 = 320	96.400	113.200	83.600	100.800	83.600	100.800	
Reading from other CPU shared memory		n3 = 1	12.900	20.800	12.200	17.100	12.200	17.100		
		n3 = 320	277.000	299.000	274.000	291.000	274.000	291.000		
		n3 = 1000	838.000	860.000	835.000	857.000	835.000	857.000		
			838.000	860.000	835.000	857.000	835.000	857.000		
Multiple CPU high-speed transmission dedicated instruction	D.DDWR n (S1) (S2) (D1) (D2)	Writes devices to another CPU.	n=1	34.700	34.900	33.500	34.400	33.500	34.400	
			n=16	85.900	87.600	75.200	75.500	75.200	75.500	
			n=96	5.600	10.200	3.300	9.900	3.300	9.900	
	DP.DDWR n (S1) (S2) (D1) (D2)		n=1	36.700	42.400	34.300	39.200	34.300	39.200	
			n=16	5.000	12.100	3.100	10.500	3.100	10.500	
			n=96	59.100	66.800	55.300	65.100	55.300	65.100	
	D.DDRD n (S1) (S2) (D1) (D2)		Reads devices from another CPU.	n=1	3.300	12.700	2.400	9.600	2.400	9.600
				n=16	50.900	64.400	45.200	48.200	45.200	48.200
				n=96	11.600	17.700	10.600	13.900	10.600	13.900
	DP.DDRD n (S1) (S2) (D1) (D2)	n=1		142.000	160.000	142.000	149.000	142.000	149.000	
		n=16		431.000	463.000	422.000	448.000	422.000	448.000	
		n=96		6.700	12.600	2.800	9.900	2.800	9.900	

### Remark

the instructions for which a rise execution instruction (□P) is not specified, the processing time is the same as an ON execution instruction.

**Example** WORDP instruction and TOP instruction

(2) Table of the time to be added when file register, module access device or link direct device is used

(a) When using Q00UJCPU, Q00UCPU, Q01UCPU and Q02UCPU

Device name		data	Device Specification Location	Processing Time (μs)				
				Q00UJCPU	Q00UCPU	Q01UCPU	Q02UCPU	
File register (R)	When standard RAM is used	Bit	Source	0.100	0.100	0.100	0.100	
			Destination	0.100	0.100	0.100	0.100	
		Word	Source	0.100	0.100	0.100	0.100	
			Destination	0.100	0.100	0.100	0.100	
		Double word	Source	0.100	0.100	0.100	0.200	
			Destination	0.100	0.100	0.100	0.200	
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	—	—	—	0.220	
			Destination	—	—	—	0.180	
		Word	Source	—	—	—	0.220	
			Destination	—	—	—	0.180	
		Double word	Source	—	—	—	0.440	
			Destination	—	—	—	0.380	
	When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	—	—	—	0.160	
			Destination	—	—	—	0.140	
		Word	Source	—	—	—	0.160	
			Destination	—	—	—	0.140	
		Double word	Source	—	—	—	0.320	
			Destination	—	—	—	0.300	
	File register (ZR)	When standard RAM is used	Bit	Source	0.120	0.120	0.120	0.120
				Destination	0.120	0.120	0.120	0.120
			Word	Source	0.120	0.120	0.120	0.120
				Destination	0.120	0.120	0.120	0.120
			Double word	Source	0.120	0.120	0.120	0.220
				Destination	0.120	0.120	0.120	0.220
When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)		Bit	Source	—	—	—	0.240	
			Destination	—	—	—	0.200	
		Word	Source	—	—	—	0.240	
			Destination	—	—	—	0.200	
		Double word	Source	—	—	—	0.460	
			Destination	—	—	—	0.400	
When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)		Bit	Source	—	—	—	0.180	
			Destination	—	—	—	0.160	
		Word	Source	—	—	—	0.180	
			Destination	—	—	—	0.160	
		Double word	Source	—	—	—	0.340	
			Destination	—	—	—	0.320	
Module access device (Un\G □ , U3En\G0 to G4095)		Bit	Source	—	—	—	12.000	
			Destination	—	—	—	17.300	
		Word	Source	—	—	—	9.700	
			Destination	—	—	—	33.000	
		Double word	Source	—	—	—	24.200	
			Destination	—	—	—	34.800	
	Link direct device (Jn\ □ )	Bit	Source	—	—	—	32.900	
			Destination	—	—	—	67.300	
Word		Source	—	—	—	37.200		
		Destination	—	—	—	37.000		
Double word		Source	—	—	—	39.500		
		Destination	—	—	—	41.900		

(b) When using Q03UD(E)CPU, Q04UD(E)HCPU, Q06UD(E)HCPU, Q10UD(E)HCPU, Q13UD(E)HCPU, Q20UD(E)HCPU and Q26UD(E)HCPU

Device name		data	Device Specification Location	Processing Time (μs)		
				Q03UD(E)CPU	Q04/ Q06UD(E)HCPU	Q10/Q13/Q20/ Q26UD(E)HCPU
File register (R)	When standard RAM is used	Bit	Source	0.100	0.048	0.048
			Destination	0.100	0.038	0.038
		Word	Source	0.100	0.048	0.048
			Destination	0.100	0.038	0.038
		Double word	Source	0.200	0.095	0.095
			Destination	0.200	0.086	0.086
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.220	0.200	0.200
			Destination	0.180	0.162	0.162
		Word	Source	0.220	0.200	0.200
			Destination	0.180	0.162	0.162
		Double word	Source	0.440	0.399	0.399
			Destination	0.380	0.361	0.361
When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.160	0.152	0.152	
		Destination	0.140	0.133	0.133	
	Word	Source	0.160	0.152	0.152	
		Destination	0.140	0.133	0.133	
	Double word	Source	0.320	0.304	0.304	
		Destination	0.300	0.295	0.295	
File rExtended data register (D)/ Extended link register (W) egister (ZR)	When standard RAM is used	Bit	Source	0.120	0.057	0.057
			Destination	0.120	0.048	0.048
		Word	Source	0.120	0.057	0.057
			Destination	0.120	0.048	0.048
		Double word	Source	0.220	0.105	0.105
			Destination	0.220	0.095	0.095
	When SRAM card is used (Q2MEM-1MBS, Q2MEM-2MBS)	Bit	Source	0.240	0.209	0.209
			Destination	0.200	0.171	0.171
		Word	Source	0.240	0.209	0.209
			Destination	0.200	0.171	0.171
		Double word	Source	0.460	0.409	0.409
			Destination	0.400	0.371	0.371
When SRAM card is used (Q3MEM-4MBS, Q3MEM-8MBS)	Bit	Source	0.180	0.162	0.162	
		Destination	0.160	0.143	0.143	
	Word	Source	0.180	0.162	0.162	
		Destination	0.160	0.143	0.143	
	Double word	Source	0.340	0.314	0.314	
		Destination	0.320	0.304	0.304	
Module access device (U\nG □ , U3En\G0 to G4095)	Bit	Source	11.700	11.200	11.200	
		Destination	15.400	15.300	15.300	
	Word	Source	9.460	9.410	9.410	
		Destination	19.000	19.000	19.000	
	Double word	Source	11.000	10.900	10.900	
		Destination	18.800	18.700	18.700	
Link direct device (Jn\ □ )	Bit	Source	32.700	31.300	31.300	
		Destination	52.300	29.900	29.900	
	Word	Source	28.500	17.300	17.300	
		Destination	27.500	14.700	14.700	
	Double word	Source	30.300	18.100	18.100	
		Destination	30.600	15.700	15.700	

A

Appendix 1 OPERATION PROCESSING TIME  
Appendix 1.4 Operation Processing Time of Universal Model QCPU

# Appendix 2 CPU PERFORMANCE COMPARISON

## Appendix 2.1 Comparison of Q with AnNCPU, AnACPU, and AnUCPU

### Appendix 2.1.1 Usable devices

TableApp.2.1 Device Comparison

Device name		QCPU		AnUCPU	AnACPU	AnNCPU		
Number of I/O points <sup>*9</sup>		Q00J: 256 points Q00: 1024 points Q01: 1024 points	Q00UJ: 256 points Q00U: 1024 points Q01U: 1024 points	Q02 Q02H Q06H Q12H Q25H Q02PH Q06PH Q12PH Q25PH Q12PRH Q25PRH Q03UD(E) Q04UD(E)H Q06UD(E)H Q10UD(E)H Q13UD(E)H Q20UD(E)H Q26UD(E)H Q02U : 2048 points	4096 points	A2U: 512 points A2U-S1: 1024 points A3U: 2048 points A4U: 4096 points	A2A: 512 points A2A-S1: 1024 points A3A: 2048 points	A1N: 256 points A2N: 512 points A2N-S1: 1024 points A3N: 2048 points
Number of I/O device points <sup>*8</sup>		2048 points <sup>*1</sup>	8192 points <sup>*1</sup>	8192 points	Same with I/O devices points of each CPU			
Internal relay		8192 points <sup>*1</sup>		Total 8192 points		Total 2048 points		
Latch relay		2048 points <sup>*1</sup>	8192 points <sup>*1</sup>					
Step relay	Sequence program	—						
	SFC	2048 points <sup>*6</sup>	8192 points	—		—		
Annunciator		1024 points <sup>*1</sup>	2048 points <sup>*1</sup>	2048 points	2048 points	256 points		
Edge relay		1024 points <sup>*1</sup>	2048 points <sup>*1</sup>	—				
Link relay		2048 points <sup>*1</sup>	8192 points <sup>*1</sup>	8192 points	4096 points	1024 points		
Link special relay		1024 points	2048 points	56 points				
Timer		512 points <sup>*1</sup>	2048 points <sup>*1</sup>	Total 2048 points		Total 256 points		
Retentive timers		0 points <sup>*1</sup>						
Counter		512 points <sup>*1</sup>	1024 points <sup>*1</sup>	1024 points		256 points		
Data register		11136 points <sup>*1</sup>	12288 points <sup>*1</sup>	8192 points	6144 points	1024 points		
Link register		2048 points <sup>*1</sup>	8192 points <sup>*1</sup>	8192 points	4096 points	1024 points		
Link special register		1024 points	2048 points	56 points				
Function input		16 points (FX0 to FXF) <sup>*7</sup>		—				
Function output		16 points (FY0 to FYF) <sup>*7</sup>		—				
Special relay		1000 points	2048 points	256 points				
Function register		5 points (FD0 to FD4)		—				
Special register		1000 points	2048 points	256 points				
Link direct device		Designated by J □ \ □		—				
Intelligent function module device		Designated by U □ \ G □		—				
Index register	Z	10 points (Z0 to Z9)	16 points (Z0 to Z15)	7 points (Z, Z1 to Z6)		1 point (Z)		
	V <sup>*2</sup>	—		7 points (V, V1 to V6)		1 point (V)		
File register		32768 points/block <sup>*5</sup> (R0 to R32767)	32768 points/block(R0 to R32767) <sup>*10</sup>	8192 points/block(R0 to R8191)				
Accumulator <sup>*3</sup>		—		2 points				
Nesting		15 points		8 points				
Pointer		300 points	512 points	4096 points	256 points			
Interrupt pointers		128 points	128 points	256 points	32 points			
SFC blocks		126 <sup>*6</sup>	320 points		—			
SFC transition devices		—		512 points				
Decimal constants		K - 2147483648 to K2147483647						
Hexadecimal constants		H0 to HFFFFFFF						
Real number constants <sup>*6</sup>		E ± 1.17550-38 to E ± 3.40282+38			—			
Character string		"QnACPU", "ABCD" <sup>*4</sup>			—			



- \*1 : The number of device points can be changed at the parameters.
- \*2 : QCPU uses V as an edge relay.
- \*3 : Instructions that used accumulators with the AnNCP, AnACP, and AnUCPU have different formats with the QCPU.
- \*4 : Can only be used by the \$MOV instruction with the Q00JCPU, Q00CPU, and Q01CPU.
- \*5 : The Q00JCPU does not have file registers.
- \*6 : Applicable to products with the first 5 digits of the serial number 04122 or higher (Q00JCPU, Q00CPU, and QCPU).
- \*7 : Each 5 points of FX0 to FX4 and FY0 to FY4 can be used on the programs.
- \*8 : The number of points that can be used on the programs
- \*9 : The number of accessible points to actual I/O modules
- \*10 : The Q00JCPU does not have file registers.%ParaEnd%

## Appendix 2.1.2 I/O control mode

TableApp.2.2 I/O Control Mode

I/O control mode		QCPU	AnUCPU	AnACP	AnNCP
Refresh mode		○	○	○	○*2
Direct I/O method	Partial refresh instructions	○	○	○	○
	Dedicated instruction*1	—	○	○	—
	Direct access input	○	—	—	—
	Direct access output	○	—	—	—
Direct mode		—	—	—	○*2

Symbol in table ○: Usable, —: Unusable

- \*1 : The DOUT, DSET, and SRST instructions are direct output dedicated instructions. There are no dedicated instructions for direct input.
- \*2 : Switching between the refresh mode and direct mode is conducted with an AnNCP DIP switch.

## Appendix 2.1.3 Data that can be used by instructions

TableApp.2.3 Data That Can Be Used by Instruction

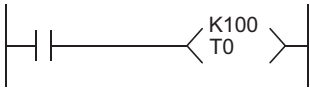
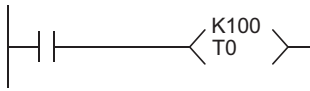
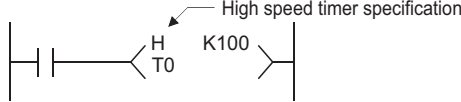
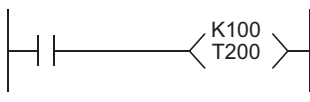
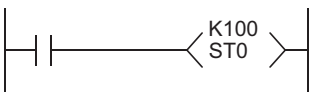
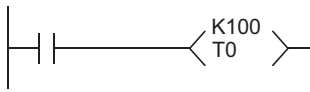
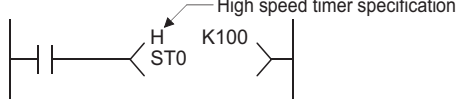
Setting Data		QCPU	AnUCPU	AnACP	AnNCP
Bit data	Bit device	○	○	○	○
	Word device	○ (Bit specification required)	—	—	—
Word data	Bit device	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)
	Word device	○	○	○	○
Double word data	Bit device	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)	○ (Digit specification required)
	Word device	○	○	○	○
Real number data		○*1	○	○	—
Character string data		○*2	—	—	—

Symbols in table ○: Usable, —: Unusable

- \*1 : Applicable to products with the first 5 digits of the serial number 04122 or higher (Q00JCPU, Q00CPU, and Q01CPU).
- \*2 : Usable with only the MOV instruction for the Q00JCPU, Q00CPU, and Q01CPU.

## Appendix 2.1.4 Timer comparison

TableApp.2.4 Timer Comparison

Function		QCPU	AnUCPU	AnACPU	AnNCPU
Low speed timer	Measurement unit	100ms (default value) Change of measurement unit at the parameter is enabled. QCPU : 1 to 1000ms (1ms unit)	Fixed at 100ms		
	Designation method				
High speed timer	Measurement unit	10ms (default value) Change of measurement unit at the parameter is enabled. QnUCPU : 0.01 to 100ms (0.01ms unit) QCPU(Other than QnUCPU) : 0.1 to 100ms (0.1ms unit)	Fixed at 10ms		
	Designation method	 High speed timer setting: Conducted by sequence program	 High speed timer setting: Conducted at parameters		
Retentive timers	Measurement unit	Same measurement unit as low speed timer		Fixed at 100ms	
	Designation method				
High speed retentive timer	Measurement unit	Same measurement unit as high speed timer		None	
	Designation method	 High speed timer setting: Conducted by sequence program			
Setting range for set values		1 to 32767		1 to 32767	
Processing for set value 0		Momentarily ON		No maximum (does not time out)	
Index modification	Contact	Enabled (only Z0 and Z1 are usable)		Enabled	Disabled
	Coil	Enabled (only Z0 and Z1 are usable)		Disabled	Disabled
	Set value	Enabled (Z0 to Z15 are usable)*1		Disabled	Disabled
	Present value	Enabled (Z0 to Z15 are usable)*1		Enabled	Enabled
Update processing for present value		When OUT Tn instruction is executed		When END processing is done	
Contact ON/OFF processing					

\*1 : The Q00J/Q00/Q01CPU can use Z0 to Z9.  
The Universal model QCPU can use Z0 to Z19.

### (1) Cautions on using timers

QCPU updates the present value of timers and turns ON/OFF the contacts of them at the execution of OUT T □ instruction.

Therefore, if "Present value  $\geq$  Set value" when the timer coil is turned ON, the contact of that timer is turned ON.

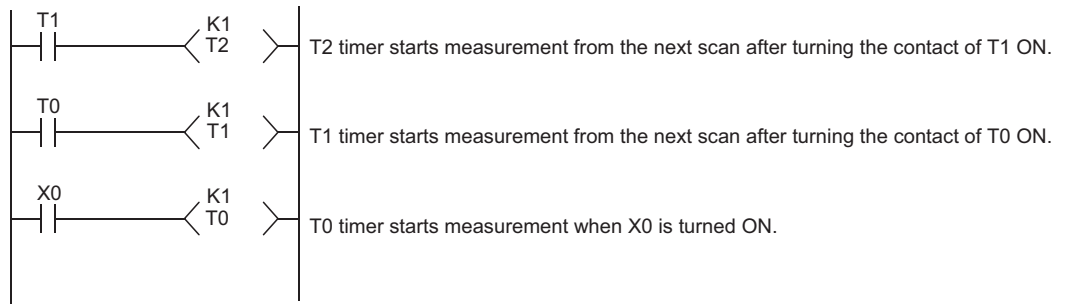
When creating a program in which the operation of the timer contact triggers the operation of another timer, create the program for the timer that operates later first.

In the following cases, all timers go ON at the same scan if the program is created in the order the timers operate.

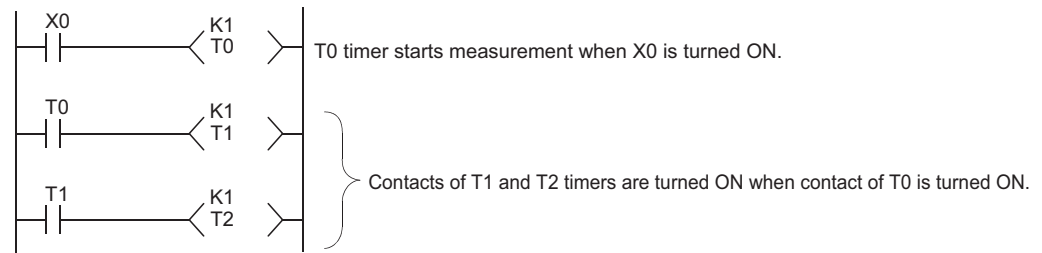
- With high speed timers, if the set value is smaller than a scan time.
- With slow timers, if "1" is set.

**Example**

- For timers T0 to T2, the program is created in the order the timer operates later.



- For timers T0 to T2, the program is created in the order of timer operation.



## Appendix 2.1.5 Comparison of counters

TableApp.2.5 Comparison of Counters

Function		QCPU	AnUCPU	AnACPU	AnNCPU
Designation method					
Index modification	Contact	• Enabled (only Z0 and Z1 are usable)	• Enabled		• Disabled
	Coil	• Enabled (only Z0 and Z1 are usable)	• Disabled		• Disabled
	Set value	• Disabled	• Disabled		• Disabled
	Present value	• Enabled (Z0 to Z15 are usable)*1	• Enabled		• Enabled
Update processing for present value		• When OUT Cn instruction is executed		• When END processing is done	
Contact ON/OFF processing					

\*1: The Q00J/Q00/Q01CPU can use Z0 to Z9.  
The Universal model QCPU can use Z0 to Z19.

## Appendix 2.1.6 Comparison of display instructions

TableApp.2.6 Comparison of Display Instructions

Instruction	QCPU	AnUCPU	AnACPU	AnNCPU
PR <sup>*1</sup>	<ul style="list-style-type: none"> <li>• When SM701 is OFF: Output continued until 00<sub>H</sub> encountered</li> <li>• When SM701 is ON: 16 characters output</li> </ul>	<ul style="list-style-type: none"> <li>• When M9049 is OFF: Output continued until 00<sub>H</sub> encountered</li> <li>• When M9049 is ON: 16 characters output</li> </ul>		
PRC <sup>*1</sup>	<ul style="list-style-type: none"> <li>• When SM701 is OFF: 32-character comment output</li> <li>• When SM701 is ON: Upper 16 characters output</li> </ul>	16-character comment output		

\*1: Unusable for the Q00J/Q00/Q01CPU.

A

Appendix 2.1 Comparison of Q with AnNCPU, AnACPU, and AnUCPU

## Appendix 2.1.7 Instructions whose designation format has been changed (Except dedicated instructions for AnACPU and AnUCPU)

Because the QCPU does not have accumulators (A0, A1), the format of AnUCPU, AnACPU and AnNCPUs instructions that used accumulators has been changed.

TableApp.2.7 Instructions Whose Expression Has Changed

Function	QCPU		AnUCPU/AnACPU/AnNCPUs	
	Instruction Format	Remarks	Instruction Format	Remarks
16-bit rotation to right		• D : Rotation data		• Rotation data are set at A0.
		• D : Rotation data • SM700 is used for carry flag.		• Rotation data are set at A0. • M9012 is used for carry flag.
16-bit rotation to left		• D : Rotation data		• Rotation data are set at A0.
		• D : Rotation data • SM700 is used for carry flag.		• Rotation data are set at A0. • M9012 is used for carry flag.
32-bit rotation to right		• D : Rotation data		• Rotation data are set at A0 and A1.
		• D : Rotation data • SM700 is used for carry flag.		• Rotation data are set at A0 and A1. • M9012 is used for carry flag.
32-bit rotation to left		• D : Rotation data		• Rotation data are set at A0 and A1.
		• D : Rotation data • SM700 is used for carry flag.		• Rotation data are set at A0 and A1. • M9012 is used for carry flag.
16-bit data search		• Search results are stored at the D and D+1 devices.		• Search results are stored at A0 and A1.
32-bit data search		• Search results are stored at the D and D+1 devices.		• Search results are stored at A0 and A1.
16-bit data bit check		• Check results are stored at the D device.		• Check results are stored at A0.
16-bit data bit check		• Check results are stored at the D device.		• Check results are stored at A0.
Partial refresh		• Dedicated instruction is added.		• Only when M9052 is ON
8-character ASCII conversion		—		—
Carry flag set		• No dedicated instruction		—
Carry flag reset		• No dedicated instruction		—
Jump to END instruction		• Dedicated instruction is added.		• P255: END instruction designation
CHK instruction*1		• The CHKST instruction is added.		—

\*1: Unusable for the Q00J/Q00/Q01CPU.

## Appendix 2.1.8 AnACPU and AnUCPU dedicated instructions

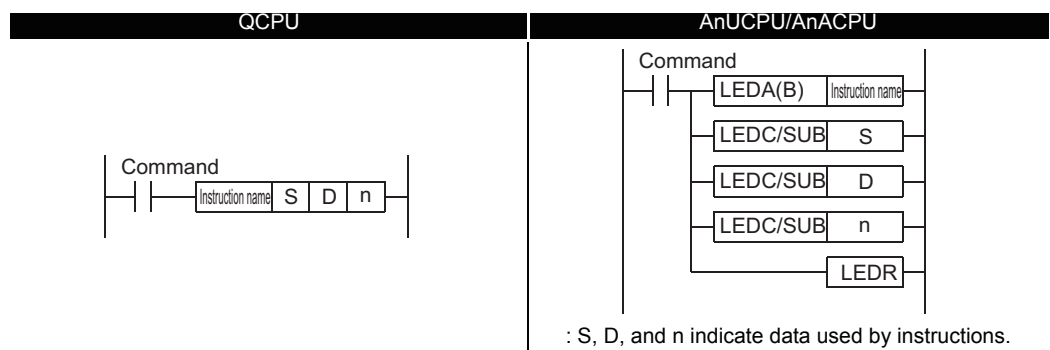
### (1) Method of expression of dedicated instructions

Dedicated instructions based on the LEDA, LEDB, LEDC, SUB, and LEDR instructions, that are used with the AnACPU or AnUCPU have been changed for the same format as the basic instructions and the application instructions for the QCPU.

The instructions that cannot be converted due to the absence of the corresponding instructions in the QCPU are converted into OUT SM1255/OUT SM999 (for the Q00J/Q00/Q01CPU).

The instructions that have been converted into OUT SM1255/OUT SM999 should be replaced by other instructions or deleted.

TableApp.2.8 Method of Expression of Dedicated Instruction



A

### (2) Dedicated instructions whose names have been changed

Dedicated instructions for the AnUCPU or AnACPU which have the same instruction name as is used for basic instructions and application instructions have undergone name changes in the QCPU.

TableApp.2.9 Method of Expression of Dedicated Instruction

Function	QCPU	AnUCPU/AnACPU
Floating point addition	E+	ADD
Floating point subtraction	E—	SUB
Floating point multiplication	E*	MUL
Floating point division	E/	DIV
Data dissociation	NDIS	DIS
Data association	NUNI	UNI
Updating check patterns	CHKCIR,CHKEND	CHK, CHKEND

## Appendix 3 SPECIAL RELAY LIST

Special relays, SM, are internal relays whose applications are fixed in the Programmable Controller.

For this reason, they cannot be used by sequence programs in the same way as the normal internal relays.

However, they can be turned ON or OFF as needed in order to control the CPU module.

The heading descriptions in the following special relay lists are shown in 3.1.

TableApp.3.1 Explanation of special relay list

Item	Function of Item
Number	• Indicates special register number
Name	• Indicates name of special relay
Meaning	• Indicates contents of special relay
Explanation	• Discusses contents of special relay in more detail
Set by (When set)	<ul style="list-style-type: none"> <li>• Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed.</li> </ul> <p>&lt;Set by&gt;</p> <ul style="list-style-type: none"> <li>S : Set by system</li> <li>U : Set by user (sequence programs or test operations from GX Developer)</li> <li>S/U : Set by both system and user</li> </ul> <p>&lt;When set&gt;</p> <p>Indicated only for registers set by system</p> <ul style="list-style-type: none"> <li>Each END : Set during each END processing</li> <li>Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN)</li> <li>Status change : Set only when there is a change in status</li> <li>Error : Set when error occurs</li> <li>Instruction execution : Set when instruction is executed</li> <li>Request : Set only when there is a user request (through SM, etc.)</li> <li>System switching : Set when system switching is executed.</li> </ul>
Corresponding ACPU M9□□□	<ul style="list-style-type: none"> <li>• Indicates the corresponding special relay (M9□□□) of the ACPU.</li> <li>(When the contents are changed, the special relay is represented M9□□□ format change. Incompatible with the Q00J/Q00/Q01 and QnPRH.)</li> <li>• New indicates the special relay newly added to the Q series CPU module.</li> </ul>
Corresponding CPU	<p>Indicates the corresponding CPU module type name.</p> <ul style="list-style-type: none"> <li>QCPU : Indicates all the Q series CPU modules.</li> <li>Q00J/Q00/Q01 : Indicates the Basic model QCPU.</li> <li>Qn(H) : Indicates the High Performance model QCPU.</li> <li>QnPH : Indicates the Process CPU.</li> <li>QnPRH : Indicates the Redundant CPU.</li> <li>QnU : Indicates the Universal model QCPU</li> </ul> <p>Each CPU module model name: Indicates the relevant specific CPU module. (Example: Q02U)</p>

For details on the following items, refer to the following manuals:

- Networks → Manual of the corresponding network module
- SFC → QCPU(Q mode)/QnACPU Programming Manual (SFC)

### POINT

Do not change the values of special relays set by the system with user program or device test operations.

Doing so may result in system downtime or communication fault.

(1) Diagnostic Information

TableApp.3.2 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU	Corresponding CPU
					M9□□□	
SM0	Diagnostic errors	OFF : No error ON : Error	<ul style="list-style-type: none"> <li>Turns ON if an error occurs as a result of diagnosis. (Includes when an annunciator is ON, and when an error is detected with CHK instruction)</li> <li>Remains ON even if the condition is restored to normal thereafter.</li> </ul>	S (Error)	New	Qn(H) QnPH QnPRH
			<ul style="list-style-type: none"> <li>Turns ON if an error occurs as a result of diagnosis. (Includes when an annunciator is ON)</li> <li>Remains ON even if the condition is restored to normal thereafter.</li> </ul>	S (Error)	New	Q00J/Q00/Q01 QnU
SM1	Self-diagnostic error	OFF : No self-diagnosis errors ON : Self-diagnosis	<ul style="list-style-type: none"> <li>Turns ON if an error occurs as a result of diagnosis. (Does not include when an annunciator is ON or when an error is detected by the CHK instruction)</li> <li>Remains ON even if the condition is restored to normal thereafter.</li> </ul>	S (Error)	M9008	Qn(H) QnPH QnPRH
			<ul style="list-style-type: none"> <li>Turns ON if an error occurs as a result of diagnosis. (Does not include when an annunciator is ON)</li> <li>Remains ON even if the condition is restored to normal thereafter.</li> </ul>	S (Error)	New	Q00J/Q00/Q01 QnU
SM5	Error common information	OFF : No error common information ON : Error common information	<ul style="list-style-type: none"> <li>When SM0 is ON, turns ON if there is error common information</li> </ul>	S (Error)	New	QCPU
SM16	Error individual information	OFF : No error individual information ON : Error individual information	<ul style="list-style-type: none"> <li>When SM0 is ON, turns ON if there is error individual information</li> </ul>	S (Error)	New	
SM50	Error reset	OFF → ON: Error reset	<ul style="list-style-type: none"> <li>Conducts error reset operation</li> </ul>	U	New	
SM51	Battery low latch	OFF : Normal ON : Battery low	<ul style="list-style-type: none"> <li>Turns ON if battery voltage at CPU module or memory card drops below rated value.</li> <li>Remains ON even if the battery voltage returns to normal thereafter.</li> <li>Synchronizes with the BAT. LED.</li> </ul>	S (Error)	M9007	Qn(H) QnPH QnPRH QnU
			<ul style="list-style-type: none"> <li>Turns ON if battery voltage at CPU module drops below rated value.</li> <li>Remains ON even if the battery voltage returns to normal thereafter.</li> <li>Synchronous with ERR. LED</li> </ul>	S (Error)	New	Q00J/Q00/Q01
SM52	Battery low	OFF : Normal ON : Battery low	<ul style="list-style-type: none"> <li>Same as SM51, but turns OFF subsequently when battery voltage returns to normal.</li> </ul>	S (Error)	M9006	QCPU
SM53	AC/DC DOWN detection	OFF : AC/DC DOWN not detected ON : AC/DC DOWN detected	<ul style="list-style-type: none"> <li>Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module.</li> <li>Reset when the power supply is switched OFF, then ON.</li> </ul>	S (Error)	M9005	
			<ul style="list-style-type: none"> <li>Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module.</li> <li>Reset when the power supply is switched OFF, then ON.</li> </ul>			

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.2 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU	Corresponding CPU		
					M9□□□			
SM56	Operation error	OFF : Normal ON : Operation error	<ul style="list-style-type: none"> <li>ON when operation error is generated</li> <li>Remains ON if the condition is restored to normal thereafter.</li> </ul>	S (Error)	M9011	QCPU		
SM60	Blown fuse detection	OFF : Normal ON : Module with blown fuse	<ul style="list-style-type: none"> <li>Turns ON if there is at least one output module whose fuse has blown.</li> <li>Remains ON if the condition is restored to normal thereafter.</li> <li>Blown fuse status is checked even for remote I/O station output modules.</li> </ul>	S (Error)	M9000			
SM61	I/O module verify error	OFF : Normal ON : Error	<ul style="list-style-type: none"> <li>Turns ON if the I/O module differs from the status registered at power on.</li> <li>Remains ON if the condition is restored to normal thereafter.</li> <li>I/O module verification is also conducted for remote I/O station modules.</li> </ul>	S (Error)	M9002			
SM62	Annunciator detection	OFF : Not detected ON : Detected	<ul style="list-style-type: none"> <li>Goes ON if even one annunciator (F) goes ON.</li> </ul>	S (Instruction execution)	M9009			
SM80	CHK detection	OFF : Not detected ON : Detected	<ul style="list-style-type: none"> <li>Goes ON if error is detected by CHK instruction.</li> <li>Remains ON if the condition is restored to normal thereafter.</li> </ul>	S (Instruction execution)	New	Qn(H) QnPH QnPRH		
SM90	Startup of monitoring timer for step transition (Enabled only when SFC program exists)	OFF : Not started (monitoring timer reset) ON : Started (monitoring timer started)	Corresponds to SD90	<ul style="list-style-type: none"> <li>Goes ON when measurement of step transition monitoring timer is commenced.</li> <li>Resets step transition monitoring timer when it goes OFF.</li> </ul>	U	M9108	Qn(H) QnPH QnPRH	
SM91			Corresponds to SD91			M9109		
SM92			Corresponds to SD92			M9110		
SM93			Corresponds to SD93			M9111		
SM94			Corresponds to SD94			M9112		
SM95			Corresponds to SD95			M9113		
SM96			Corresponds to SD96			M9114		
SM97			Corresponds to SD97			New		
SM98			Corresponds to SD98			New		
SM99			Corresponds to SD99			New		
SM100	Serial communication function using flag	OFF : Serial communication function is not used. ON : Serial communication function is used.	<ul style="list-style-type: none"> <li>Stores the setting of whether the serial communication function is used or not in the serial communication setting parameter</li> </ul>	S (Power-ON or reset)	New	Q00/Q01 Q00U Q00U Q01U Q02U <sup>*7</sup>		
SM101	Communication protocol status flag	OFF : GX Developer ON : MC protocol communication device	<ul style="list-style-type: none"> <li>Stores whether the device that is communicating via the RS-232 interface is GX Developer or MC protocol communication device</li> </ul>	S (RS232 communication)				
SM110	Protocol error	OFF : Normal ON : Abnormal	<ul style="list-style-type: none"> <li>Turns ON when an abnormal protocol was used to make communication in the serial communication function.</li> <li>Remains ON if the condition is restored to normal thereafter</li> </ul>	S (Error)				
SM111	Communication status	OFF : Normal ON : Abnormal	<ul style="list-style-type: none"> <li>Turns ON when the mode used to make communication was different from the setting in the serial communication function.</li> <li>Remains ON if the condition is restored to normal thereafter.</li> </ul>	S (Error)				
SM112	Error information clear	ON : Cleared	<ul style="list-style-type: none"> <li>Turns ON when the error codes stored in SM110, SM111, SD110 and SD111 are cleared. (Activated when turned from OFF to ON)</li> </ul>	U				
SM113	Overrun error	OFF : Normal ON : Abnormal	<ul style="list-style-type: none"> <li>Turns ON when an overrun error occurred in the serial communication error.</li> </ul>	S (Error)				
SM114	Parity error	OFF : Normal ON : Abnormal	<ul style="list-style-type: none"> <li>Turns ON when a parity error occurred in the serial communication error.</li> </ul>	S (Error)				
SM115	Framing error	OFF : Normal ON : Abnormal	<ul style="list-style-type: none"> <li>Turns ON when a framing error occurred in the serial communication error.</li> </ul>	S (Error)				
SM165	Program memory batch transfer execution status	OFF : Completed ON : Not being executed or Not completed	<ul style="list-style-type: none"> <li>Turns ON when the data is written to the program cache memory.</li> <li>Turns OFF when the program memory batch transfer is completed.</li> <li>Remains ON if the program memory batch transfer is not executed after the data is written to the program cache memory.</li> </ul>	S (When status changed)			New	QnU <sup>*6</sup>

\*6: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10012" or later.
- Q13UDHCPU, Q26UDHCPU

\*7: The module whose first 5 digits of serial No. is "10102" or later.



(2) System information

TableApp.3.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM202	LED OFF command	OFF → ON : LED OFF	• When this relay goes from OFF to ON, the LEDs corresponding to the individual bits at SD202 go off	U	New	Qn(H) QnPH QnPRH QnU
SM203	STOP contact	STOP status	• Goes ON at STOP status	S (Status change)	M9042	QCPU
SM204	PAUSE contact	PAUSE status	• Goes ON at PAUSE status	S (Status change)	M9041	
SM206	PAUSE enable coil	OFF : PAUSE disabled ON : PAUSE enabled	• PAUSE status is entered if this relay is ON when the PAUSE contact goes ON	U	M9040	
SM210	Clock data set request	OFF : Ignored ON : Set request	• When this relay goes from OFF to ON and after END instruction execution of subsequent scan, clock data stored in SD210 to SD213 are written to the CPU module.	U	M9025	
SM211	Clock data error	OFF : No error ON : Error	• ON when error is generated in clock data (SD210 to SD213) value, and OFF if no error is detected.	S (Request)	M9026	QCPU
SM213	Clock data read request	OFF : Ignored ON : Read request	• When this relay is ON, clock data is read to SD210 to SD213 as BCD values.	U	M9028	
SM220	CPU No.1 preparation completed	OFF : CPU No.1 preparation uncompleted ON : CPU No.1 preparation completed	Turned ON when access can be made to the CPU module No.1 from the other CPU module at power-on or reset operation. SM220 is used as interlock for accessing the CPU module No.1 when the multiple CPU synchronous setting is asynchronous.	S (When status changed)	New	QnU
SM221	CPU No.2 preparation completed	OFF : CPU No.2 preparation uncompleted ON : CPU No.2 preparation completed	Turned ON when access can be made to the CPU module No.2 from the other CPU module at power-on or reset operation. SM221 is used as interlock for accessing the CPU module No.2 when the multiple CPU synchronous setting is asynchronous.			QnU <sup>*8</sup>
SM222	CPU No.3 preparation completed	OFF : CPU No.3 preparation uncompleted ON : CPU No.3 preparation completed	Turned ON when access can be made to the CPU module No.3 from the other CPU module at power-on or reset operation. SM222 is used as interlock for accessing the CPU module No.3 when the multiple CPU synchronous setting is asynchronous.			QnU <sup>*5</sup>
SM223	CPU No.4 preparation completed	OFF : CPU No.4 preparation uncompleted ON : CPU No.4 preparation completed	Turned ON when access can be made to the CPU module No.4 from the other CPU module at power-on or reset operation. SM223 is used as interlock for accessing the CPU module No.4 when the multiple CPU synchronous setting is asynchronous.			
SM235	Online module change flag	OFF : Online module change is not in progress ON : Online module change in progress	• Turns on during online module change. (for host CPU)	S (During online module change)	New	QnPH
SM236	Online module change complete flag	OFF : Online module change incomplete ON : Online module change complete	• Turns ON for one scan after online module change is complete. • This contact point can only be used by the scan program. (for host CPU)	S (When online module change is complete)	New	
SM237	Device range check inhibit flag	OFF : Device range checked ON : Device range not checked	• Selects whether to check a device range during execution of the BMOV, FMOV or DFMOV instruction (only when the conditions for subset processing are established).	U	New	QnU <sup>*6</sup>

\*5: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*6: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10012" or later.
- Q13UDHCPU, Q26UDHCPU

\*8: The Universal model QCPU except the Q00UJCPU.

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM240	No. 1 CPU reset flag	OFF : No. 1 CPU reset cancel ON : No. 1 CPU resetting	<ul style="list-style-type: none"> <li>Goes OFF when reset of the No. 1 CPU is canceled.</li> <li>Comes ON when the No. 1 CPU is resetting (including the case where the CPU module is removed from the base).</li> <li>The other CPUs are also put in reset status.</li> </ul>	S (Status change)	New	Q00/Q01 <sup>*1</sup> Qn(H) <sup>*1</sup> QnPH QnU <sup>*8</sup>
SM241	No. 2 CPU reset flag	OFF : No. 2 CPU reset cancel ON : No. 2 CPU resetting	<ul style="list-style-type: none"> <li>Goes OFF when reset of the No. 2 CPU is canceled.</li> <li>Comes ON when the No. 2 CPU is resetting (including the case where the CPU module is removed from the base).</li> <li>The other CPUs result in "MULTI CPU DOWN" (error code: 7000).</li> </ul>			
SM242	No. 3 CPU reset flag	OFF : No. 3 CPU reset cancel ON : No. 3 CPU resetting	<ul style="list-style-type: none"> <li>Goes OFF when reset of the No. 3 CPU is canceled.</li> <li>Comes ON when the No. 3 CPU is resetting (including the case where the CPU module is removed from the base).</li> <li>The other CPUs result in "MULTI CPU DOWN" (error code: 7000).</li> </ul>			
SM243	No. 4 CPU reset flag	OFF : No. 4 CPU reset cancel ON : No. 4 CPU resetting	<ul style="list-style-type: none"> <li>Goes OFF when reset of the No. 4 CPU is canceled.</li> <li>Comes ON when the No. 4 CPU is resetting (including the case where the CPU module is removed from the base).</li> <li>The other CPUs result in "MULTI CPU DOWN" (error code: 7000).</li> </ul>			
SM244	No. 1 CPU error flag	OFF : No. 1 CPU normal ON : No. 1 CPU during stop error	<ul style="list-style-type: none"> <li>Goes OFF when the No. 1 CPU is normal (including a continuation error).</li> <li>Comes ON when the No. 1 CPU is during a stop error.</li> </ul>	S (Status change)	New	Q00/Q01 <sup>*1</sup> Qn(H) <sup>*1</sup> QnPH QnU <sup>*8</sup>
SM245	No. 2 CPU error flag	OFF : No. 2 CPU normal ON : No. 2 CPU during stop error	<ul style="list-style-type: none"> <li>Goes OFF when the No. 2 CPU is normal (including a continuation error).</li> <li>Comes ON when the No. 2 CPU is during a stop error.</li> </ul>			
SM246	No. 3 CPU error flag	OFF : No. 3 CPU normal ON : No. 3 CPU during stop error	<ul style="list-style-type: none"> <li>Goes OFF when the No. 3 CPU is normal (including a continuation error).</li> <li>Comes ON when the No. 3 CPU is during a stop error.</li> </ul>			
SM247	No. 4 CPU error flag	OFF : No. 4 CPU normal ON : No. 4 CPU during stop error	<ul style="list-style-type: none"> <li>Goes OFF when the No. 4 CPU is normal (including a continuation error).</li> <li>Comes ON when the No. 4 CPU is during a stop error.</li> </ul>			
SM250	Max. loaded I/O read	OFF : Ignored ON : Read	<ul style="list-style-type: none"> <li>When this relay goes from OFF to ON, maximum loaded I/O number is read to SD250.</li> </ul>	U	New	
SM254	All stations refresh command	OFF : Refresh arrival station ON : Refresh all stations	<ul style="list-style-type: none"> <li>Effective for the batch refresh (also effective for the low speed cyclic)</li> <li>Designate whether to receive arrival stations only or to receive all slave stations in the MELSECNET/H.</li> </ul>	U	New	Qn(H) QnPH QnPRH
			<ul style="list-style-type: none"> <li>Designate whether to receive arrival stations only or to receive all slave stations in the CC-Link IE controller network .</li> </ul>			Qn(H) <sup>*2</sup> QnPH <sup>*6</sup> QnPRH <sup>*6</sup>
			<ul style="list-style-type: none"> <li>Effective for the batch refresh (also effective for the low speed cyclic)</li> <li>Specify whether to receive only arrival station or all stations in the MELSECNET/H or CC-Link IE controller network.</li> </ul>			QnU

\*1: This applies to the CPU of function version B or later.

\*2: The module whose first 5 digits of serial No. is "09012" or later.

\*5: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*6: The module whose first 5 digits of serial No. is "10042" or later.

\*8: The Universal model QCPU except the Q00UJCPU.

TableApp.3.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM255	MELSECNET/10, MELSECNET/H module 1 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	Qn(H) QnPH QnPRH
SM256		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM257		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM260	MELSECNET/10, MELSECNET/H module 2 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	
SM261		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM262		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM265	MELSECNET/10, MELSECNET/H module 3 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	
SM266		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM267		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM270	MELSECNET/10, MELSECNET/H module 4 information	OFF : Operative network ON : Standby network	• Goes ON for standby network (If no designation has been made concerning active or standby, active is assumed.)	S (Initial)	New	
SM271		OFF : Reads ON : Does not read	• For refresh from link to CPU module (B, W, etc.) indicate whether to read from the link module.	U	New	
SM272		OFF : Writes ON : Does not write	• For refresh from CPU module to link (B, W, etc.), designate whether to write to the link module.	U	New	
SM280	CC-Link error	OFF : Normal ON : Error	• Goes ON when a CC-Link error is detected in any of the installed CC-Link module. Goes OFF when normal operation is restored.	S (Status change)	New	
SM315	Communication reserved time delay enable/disable flag	OFF : Without delay ON : With delay	• This flag is enabled when the time reserved for communication processing is set in SD315. • Turns ON to delay the END processing by the time set in SD315 in order to perform communication processing. (The scan time increases by the period set in SD315.) • Turns OFF to perform the END processing without a delay of the time set in SD315 when there is no communication processing. (Defaults to OFF)	U	New	Q00J/Q00/Q01
SM320	Presence/absence of SFC program	OFF : SFC program absent ON : SFC program present	• Turns ON when an SFC program is registered. • OFF when an SFC program is not registered.	S (Initial)	M9100	Q00J/Q00/Q01 <sup>*1</sup> Qn(H) QnPH QnPRH QnU
SM321	Start/stop SFC program	OFF : SFC program not executed (stop) ON : SFC program executed (start)	• Initial value is set at the same value as SM320. (Goes ON automatically if SFC program is present.) • Turn this relay from ON to OFF to stop program execution. • Turn this relay from OFF to ON to resume program execution.	S (Initial)/U	M9101form at change	

\*1: This applies to the CPU of function version B or later.

TableApp.3.3 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM322	SFC program start status	OFF : Initial start ON : Resume start	• The SFC program starting mode in the SFC setting of the PLC parameter dialog box is set as the initial value. AT initial start: OFF At continued start: ON	S (Initial)/U	M9102form at change	Q00J/Q00/Q01 <sup>*1</sup> Qn(H) QnPH QnPRH QnU
SM323	Presence/absence of continuous transition for entire block	OFF : Continuous transition not effective ON : Continuous transition effective	Set the presence/absence of continuous transition for the block where "Continuous transition bit" of the SFC data device has not been set.	U	M9103	
SM324	Continuous transition prevention flag	OFF : When transition is executed ON : When no transition	• OFF during operation in the continuous transition mode or during continuous transition, and ON when continuous transition is not executed. • Always ON during operation in the no continuous transition mode.	S (Instruction execution)	M9104	
				S (Status change)	New	
SM325	Output mode at block stop	OFF : OFF ON : Preserves	Select whether the coil outputs of the active steps are held or not at the time of a block stop. • As the initial value, the output mode at a block stop in the parameter is OFF when the coil outputs are OFF, and ON when the coil outputs are held. • All coil outputs go OFF when this relay is OFF. • Coil outputs are preserved when this relay is ON.	S (Initial)/U	M9196	
SM326	SFC device clear mode	OFF : Clear device ON : Preserves device	Selects the device status when the stopped CPU is run after the sequence program or SFC program has been modified when the SFC program exists.	U	New	
SM327	Output during end step execution	OFF : Hold step output turned OFF (cleared) ON : Hold step output held	Select the device status at the time of switching from STOP to program write to RUN.(All devices except the step relay)	S (Initial)/U	New	Qn(H) QnPH QnPRH QnU
				U		
SM328	Clear processing mode when end step is reached	OFF : Clear processing is performed. ON : Clear processing is not performed.	Select whether clear processing will be performed or not if active steps other than the ones being held exist in the block when the end step is reached? • When this relay turns OFF, all active steps are forcibly terminated to terminate the block. • When this relay is ON, the execution of the block is continued as-is. • If active steps other than the ones being held do not exist when the end step is reached, the steps being held are terminated to terminate the block.	U	New	Q00J/Q00/Q01 <sup>*1</sup> QnU
SM330	Operation mode for low speed execution type program	OFF : Asynchronous mode ON : Synchronous mode	Select whether the low speed execution type program will be executed in the asynchronous mode or in the synchronous mode. • Asynchronous mode (this relay is turned OFF.) Mode in which the operation of the low speed execution type program is performed continuously within the excess time. • Synchronous mode (this relay is turned ON.) Mode in which the operation of the low speed execution type program is not performed continuously and operation is performed from the next scan if there is excess time.	U	New	Qn(H) QnPH
SM331	Normal SFC program execution status	OFF : Not executed ON : Being executed	• Indicates whether the normal SFC program is being executed or not. • Used as an SFC control instruction execution interlock.	S (Status change)	New	Qn(H) <sup>*3</sup> QnPH <sup>*4</sup> QnPRH
SM332	Program execution management SFC program execution status	OFF : Not executed ON : Being executed	• Indicates whether the program execution management SFC program is being executed or not. • Used as an SFC control instruction execution interlock.			
SM390	Access execution flag	ON indicates completion of intelligent function module access	• The status of the intelligent function module access instruction executed immediately before is stored. (This data is overwritten when the intelligent function module access instruction is executed again.) • Used by the user in a program as a completion bit.	S (Status change)	New	Qn(H) QnPH QnPRH
SM391	GINT instruction execution completion flag	OFF : Not executed ON : Execution completed	Indicates execution status of the S(P).GINT instruction. • Turned OFF before the instruction is executed. • Turned ON after the instruction is completed.	S (Instruction execution)	New	QnU

\*1: This applies to the CPU of function version B or later.

\*3: The module whose first 5 digits of serial No. is "04122" or later.

\*4: The module whose first 5 digits of serial No. is "07032" or later.

(3) System clocks/counters

TableApp.3.4 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM400	Always ON	ON _____ OFF	• Normally is ON	S (Every END processing)	M9036	QCPU
SM401	Always OFF	ON _____ OFF _____	• Normally is OFF	S (Every END processing)	M9037	
SM402	After RUN, ON for 1 scan only	ON _____ OFF ← 1 scan	<ul style="list-style-type: none"> <li>• After RUN, ON for 1 scan only.</li> <li>• This connection can be used for scan execution type programs only.</li> <li>• When an initial execution type program is used, this relay turns OFF at the END processing of the scan execution type program in the first scan after RUN.</li> </ul>	S (Every END processing)	M9038	Qn(H) QnPH QnPRH QnU
			<ul style="list-style-type: none"> <li>• After RUN, ON for 1 scan only.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01
SM403	After RUN, OFF for 1 scan only	ON _____ OFF ← 1 scan	<ul style="list-style-type: none"> <li>• After RUN, OFF for 1 scan only.</li> <li>• This connection can be used for scan execution type programs only.</li> <li>• When an initial execution type program is used, this relay turns OFF at the END processing of the scan execution type program in the first scan after RUN.</li> </ul>	S (Every END processing)	M9039	Qn(H) QnPH QnPRH QnU
			<ul style="list-style-type: none"> <li>• After RUN, OFF for 1 scan only.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01
SM404	Low speed execution type program ON for 1 scan only after RUN	ON _____ OFF ← 1 scan	<ul style="list-style-type: none"> <li>• After RUN, ON for 1 scan only.</li> <li>• This connection can be used for low speed execution type programs only.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH
SM405	Low speed execution type program After RUN, OFF for 1 scan only	ON _____ OFF ← 1 scan	<ul style="list-style-type: none"> <li>• After RUN, OFF for 1 scan only.</li> <li>• This connection can be used for low speed execution type programs only.</li> </ul>	S (Every END processing)	New	
SM409	0.01 second clock		<ul style="list-style-type: none"> <li>• Repeatedly changes between ON and OFF at 5-ms interval.</li> <li>• When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)</li> </ul>	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SM410	0.1 second clock		<ul style="list-style-type: none"> <li>• Repeatedly changes between ON and OFF at each designated time interval.</li> <li>• When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)</li> </ul>	S (Status change)	M9030	QCPU
SM411	0.2 second clock				M9031	
SM412	1 second clock				M9032	
SM413	2 second clock				M9033	
SM414	2n second clock		<ul style="list-style-type: none"> <li>• This relay alternates between ON and OFF at intervals of the time (unit: s) specified in SD414.</li> <li>• When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start. (Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)</li> </ul>	S (Status change)	M9034form at change	

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.4 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU			
SM415	2n (ms) clock		<ul style="list-style-type: none"> <li>This relay alternates between ON and OFF at intervals of the time (unit: ms) specified in SD415.</li> <li>When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start.</li> <li>(Note that the ON-OFF status changes when the designated time has elapsed during the execution of the program.)</li> </ul>	S (Status change)	New	Qn(H) QnPH QnPRH QnU			
SM420	User timing clock No.0		<ul style="list-style-type: none"> <li>Relay repeats ON/OFF switching at fixed scan intervals.</li> <li>When Programmable Controller power supply is turned ON or a CPU module reset is performed, goes from OFF to start.</li> <li>(For the redundant CPU, however, this relay is always OFF after system switching.)</li> <li>The ON/OFF intervals are set with the DUTY instruction</li> </ul>	S (Every END processing)	M9020	QCPU			
SM421	User timing clock No.1				M9021				
SM422	User timing clock No.2				M9022				
SM423	User timing clock No.3				M9023				
SM424	User timing clock No.4				M9024				
SM430	User timing clock No.5				<ul style="list-style-type: none"> <li>For use with SM420 to SM424 low speed programs</li> </ul>		S (Every END processing)	New	Qn(H) QnPH
SM431	User timing clock No.6								
SM432	User timing clock No.7								
SM433	User timing clock No.8								
SM434	User timing clock No.9								

## (4) Scan information

TableApp.3.5 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM510	Low speed program execution flag	OFF : Completed or not executed ON : Execution under way.	<ul style="list-style-type: none"> <li>Goes ON when low speed execution type program is executed.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH
SM551	Reads module service interval	OFF : Ignored ON : Read	<ul style="list-style-type: none"> <li>When this relay goes from OFF to ON, the module service interval designated by SD550 is read to SD551 to SD552.</li> </ul>	U	New	Qn(H) QnPH QnPRH

## (5) I/O refresh

TableApp.3.6 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM580	Program to program I/O refresh	OFF : Not refreshed ON : Refreshed	<ul style="list-style-type: none"> <li>When this special relay is turned ON, I/O refresh is performed after execution of the first program, and the next program is then executed.</li> <li>When a sequence program and an SFC program are to be executed, the sequence program is executed, I/O refresh is performed, and the SFC program is then executed.</li> </ul>	U	New	Q00J/Q00/Q01 <sup>*1</sup>

\*1: This applies to the CPU of function version B or later.

(6) Memory cards

TableApp.3.7 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM600	Memory card usable flags	OFF : Unusable ON : Use enabled	• ON when memory card is ready for use by user	S (Status change)	New	Qn(H) QnPH QnPRH QnU <sup>*1</sup>
SM601	Memory card protect flag	OFF : No protect ON : Protect	• Goes ON when memory card protect switch is ON	S (Status change)	New	
SM602	Drive 1 flag	OFF : No drive 1 ON : Drive 1 present	• Turns ON when the mounted memory card is RAM	S (Status change)	New	
SM603	Drive 2 flag	OFF : No drive 2 ON : Drive 2 present	• Turns ON when the mounted memory card is ROM	S (Status change)	New	
SM604	Memory card in-use flag	OFF : Not used ON : In use	• Goes ON when memory card is in use	S (Status change)	New	
SM605	Memory card remove/insert prohibit flag	OFF : Remove/insert enabled ON : Remove/insert prohibited	• Goes ON when memory card cannot be inserted or removed	U	New	
SM609	Memory card remove/insert enable flag	OFF : Remove/insert prohibited ON : Remove/insert enabled	• Turned ON by user to enable the removal/insertion of memory card. • Turned OFF by the system after the memory card is removed. • This contact can be used only when SM604 and SM605 are OFF.	S/U	New	
SM620	Drive 3/4 usable flags	OFF : Unusable ON : Use enabled	• Always ON	S (Initial)	New	QCPU
SM622	Drive 3 flag	OFF : No drive 3 ON : Drive 3 present	• Always ON	S (Initial)	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU <sup>*2</sup>
SM623	Drive 4 flag	OFF : No drive 4 ON : Drive 4 present	• Always ON	S (Initial)	New	QCPU
SM624	Drive 3/4 in-use flag	OFF : Not used ON : In use	• Goes ON when the file within Drive 3 (standard RAM) or Drive 4 (standard ROM) is used.	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SM640	File register use	OFF : File register not used ON : File register in use	• Goes ON when file register is in use	S (Status change)	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU <sup>*2</sup>

\*1: The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

\*2: The Universal model QCPU except the Q00UJCPU.

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.7 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU	Corresponding CPU
					M9□□□	
SM650	Comment use	OFF : File register not used ON : File register in use	• Goes ON when comment file is in use	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SM660	Boot operation	OFF : Internal memory execution ON : Boot operation in progress	• Goes ON while boot operation is in process • Goes OFF if boot designation switch is OFF	S (Status change)	New	Qn(H) QnPH QnPRH
		OFF : Program memory execution ON : Boot operation in progress	• Goes ON while boot operation is in process	S (Status change)	New	Q00J/Q00/Q01 QnU*1
SM671	Latch data backup to standard ROM completion flag	OFF : Not completed ON : Completed	• Turned ON when latch data backup to the standard ROM is completed. • Time when the latch data backup to the standard ROM was performed is stored in SD672 or later.	S (Status change)	New	QnU
SM672	Memory card file register access range flag	OFF : Within access range ON : Outside access range	• Goes ON when access is made to area outside the range of file register of memory card(Set within END processing.) • Reset at user program	S/U	New	Qn(H) QnPH QnPRH
SM675	Error completion of latch data backup to standard ROM	OFF : No Error ON : Error	• Turned ON when data cannot be backed up to the standard ROM by the latch data backup normally. • Turned OFF when data is backed up to the standard ROM by the latch data backup normally.	S	New	
SM676	Specification of restration repeated execution	OFF : Not specified ON : Specified	• If latch data backup is performed when SM676 is ON, restore the data every time turning ON from OFF the power supply from the next power-on. • Delete the backed up latch data, or restore the data every time turning ON from OFF the power supply until the latch data backup operation will be executed again.	U	New	
SM680	Program memory write error	OFF : Write error ON : Write not executed/normal	• Turns ON if a write error is detected at writing to program memory (flash ROM). Turns OFF by the write direction.	S (At write)	New	
SM681	Program memory writing flag	OFF : During writing ON : Write not executed	• Turns ON when writing to the program memory (flash ROM) is in progress, and turns OFF when writing is completed.	S (At write)	New	QnU
SM682	Program memory overwrite count error flag	OFF : Overwrite count is 100,000 or more ON : Overwrite count is less than 100,000	• Turns ON when the overwrite count of program memory (flash ROM) reaches 100,000.	S (At write)	New	
SM685	Standard ROM write error	OFF : Write error ON : Write not executed/normal	• Turns ON when write error is detected at writing to standard ROM (flash ROM). • Turns OFF by the write direction.	S (At write)	New	
SM686	Standard ROM writing flag	OFF : During overwriting ON : Overwrite not executed	• Turns ON when writing to the standard ROM (flash ROM) is in progress, and turns OFF when writing is completed.	S (At write)	New	
SM687	Standard ROM overwrite count error flag	OFF : Overwrite count is 100,000 or more ON : Overwrite count is less than 100,000	• Turns ON when the overwrite count of standard ROM (flash ROM) reaches 100,000. (It is necessary to change CPU module.)	S (At write)	New	
SM691	Backup start preparation status flag	OFF : Backup start preparation not completed ON : Backup start preparation completed	Turns on when the backup start preparation is completed.	S (Status change)	New	QnU*3
SM692	Restoration complete flag	OFF : Restoration not completed ON : Restoration completed	Turns on when restoration of the backup data in the memory card is completed.	S (Status change)	New	

\*1: The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

\*3: The modules whose serial number (first five digits) is "10102" or later are the relevant models. (Except the Q00UJCPU, Q00UCPU, and Q01UCPU)



## (7) Instruction-Related Special Relays

TableApp.3.8 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres-	Corresponding CPU
					ponding ACPU	
					M9 □ □ □	
SM700	Carry flag	OFF : Carry OFF ON : Carry ON	• Carry flag used in application instruction	S (Instruction execution)	M9012	QCPU
SM701	Number of output characters selection	Switching the number of output characters and the output pattern	• Used for the PR, PRC, BINDA, DBINDA, BINHA, DBINHA, BCDDA, DBCDDA, or COMRD instruction	U	M9049	Qn(H) QnPH QnPRH QnU
SM702	Search method	OFF : Search next ON : 2-part search	• Designates method to be used by search instruction. • Data must be arranged for 2-part search.	U	New	
SM703	Sort order	OFF : Ascending order ON : Descending order	• The sort instruction is used to designate whether data should be sorted in ascending order or in descending order.	U	New	QCPU
SM704	Block comparison	OFF : Non-match found ON : All match	• Goes ON when all data conditions have been met for the BKCMP instruction.	S (Instruction execution)	New	
			• Goes ON when all data conditions have been met for the DBKCMP instruction.	S (Instruction execution)	New	
SM709	DT/TM instruction improper data detection flag	OFF : Improper data not detected ON : Improper data detected	Turns on when the data to be compared by the DT or TM instruction is not recognized as date data or time data, or the device (3 words) to be compared exceeds the specified device range.	S (Instruction execution) or U	New	QnU <sup>2</sup>
SM710	CHK instruction priority ranking flag	OFF : Conditions priority ON : Pattern priority	• Remains as originally set when OFF. • CHK priorities updated when ON.	S (Instruction execution)	New	Qn(H) QnPH QnPRH
SM715	EI flag	OFF : During DI ON : During EI	• ON when EI instruction is being executed.	S (Instruction execution)	New	QCPU
SM716	Block comparison (Except an interrupt program)	OFF : Mismatch found ON : No mismatch	Turns on when all data conditions are confirmed that they are met by the DBKCMP instruction. (Initial execution type program, scan execution type program, stand-by type program executed from initial execution type program or scan execution type program)	S (Instruction execution)	New	QnU <sup>2</sup>
SM717	Block comparison (Interrupt program)	OFF : Mismatch found ON : No mismatch	Turns on when all data conditions are confirmed that they are met by the DBKCMP instruction. (Interrupt program, fixed scan execution type program, stand-by type program executed from interrupt program or fixed scan execution type program)			
SM718	Block comparison (Interrupt program (I45))	OFF : Mismatch found ON : No mismatch	Turns on when all data conditions are confirmed that they are met by the DBKCMP instruction. (Interrupt program (I45) or Stand-by type program executed from interrupt program (I45))			
SM720	Comment read completion flag	OFF : Comment read not completed ON : Comment read completed	• Turns on only during one scan when the processing of the COMRD or PRC instruction is completed.	S (Status change)	New	Qn(H) QnPH
			• Turns on only during one scan when the processing of the COMRD instruction is completed.			QnPRH QnU
SM721	File being accessed	OFF : File not accessed ON : File being accessed	• Switches ON while a file is being accessed by the SP. FWRITE, SP. FREAD, COMRD, PRC, or LEDC instruction.	S (Status change)	New	Qn(H) QnPH
			• Switches ON while a file is being accessed by the SP. FWRITE, SP. FREAD, COMRD or LEDC instruction.			Qn(H) QnPH QnPRH
			• Switches ON while a file is being accessed by the SP. FWRITE, SP. FREAD, COMRD or SP.DEVST instruction.			QnU
			• Turns ON while the ATA card or standard ROM is being accessed.			QnU <sup>1</sup>
SM722	BIN/DBIN instruction error disabling flag	OFF : Error detection performed ON : Error detection not performed	• Turned ON when "OPERATION ERROR" is suppressed for BIN or DBIN instruction.	U	New	QCPU

\*1: The module whose first 5 digits of serial No. is "09042" or later.

\*2: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU

\*3: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UCPU, Q01UCPU

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.8 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM734	XCALL instruction execution condition designation	OFF : Not executed by execution condition risen ON : Executed by execution condition risen	<ul style="list-style-type: none"> <li>During OFF, XCALL instructions will not be executed even if execution condition is risen.</li> <li>During ON, XCALL instructions will be executed when execution condition is risen.</li> </ul>	U	New	Qn(H) <sup>*1</sup>
SM735	SFC comment readout instruction in execution flag	OFF : SFC comment readout instruction is inactivated. ON : SFC comment readout instruction is activating.	<ul style="list-style-type: none"> <li>Turns on the instructions, (S(P).SFCSCOMR) to read the SFC step comments and (S(P).SFACTCOMR) to read the SFC transition condition comments.</li> </ul>	S (status change)	New	Qn(H) <sup>*2</sup> QnPH <sup>*3</sup> QnPRH <sup>*3</sup>
SM738	MSG instruction reception flag	OFF : Instruction not executed ON : Instruction execution	<ul style="list-style-type: none"> <li>Goes ON when MSG instruction is executed</li> </ul>	S (Instruction execution)	New	Qn(H) QnPRH
SM750	Scaling instruction search method setting	OFF : Search next ON : 2-part search	Determines a search method when the scaling instruction is executed.	U	New	QnU <sup>*8</sup>
SM774	PID bumpless processing (for complete derivative)	OFF : Matched ON : Not matched	<ul style="list-style-type: none"> <li>Specifies whether to match the set value (SV) with the process value (PV) or not in the manual mode.</li> </ul>	U	New	Q00J/Q00/Q01 <sup>*4</sup> Qn(H) QnPRH QnU
SM775	Selection of refresh processing during COM/CCOM instruction execution	OFF : Performs link refresh ON : Performs no link refresh	<ul style="list-style-type: none"> <li>Select whether link refresh processing will be performed or not when only communication with the CPU module is made at the execution of the COM instruction.</li> </ul>	U	New	Q00J/Q00/Q01 Qn(H) QnPH
		OFF : Performs refresh processes other than an I/O refresh ON : Performs refresh set by SD778	<ul style="list-style-type: none"> <li>Select whether to perform refresh processes other than an I/O refresh set by SD778 when the COM or CCOM instruction is executed.</li> </ul>	U	New	Q00J/Q00/Q01 <sup>*4</sup> Qn(H) <sup>*5</sup> QnPH <sup>*3</sup> QnPRH QnU
SM776	Enable/disable local device at CALL	OFF : Local device disabled ON : Local device enabled	<ul style="list-style-type: none"> <li>Set whether the local device of the subroutine program called at execution of the CALL instruction is valid or invalid.</li> </ul>	U	New	Qn(H) QnPH QnPRH QnU <sup>*9</sup>
SM777	Enable/disable local device in interrupt program	OFF : Local device disabled ON : Local device enabled	<ul style="list-style-type: none"> <li>Set whether the local device at execution of the interrupt program is valid or invalid.</li> </ul>	U	New	
SM794	PID bumpless processing(for incomplete derivative)	OFF : Matched ON : Not matched	<ul style="list-style-type: none"> <li>Specifies whether to match the set value (SV) with the process value (PV) or not in the manual mode.</li> </ul>	U	New	Q00J/Q00/Q01 <sup>*4</sup> Qn(H) <sup>*6</sup> QnPRH QnU

\*1: The module whose first 5 digits of serial No. is "06082" or later.

\*2: The module whose first 5 digits of serial No. is "07012" or later.

\*3: The module whose first 5 digits of serial No. is "07032" or later.

\*4: This applies to the CPU module of function version B or later.

\*5: The module whose first 5 digits of serial No. is "04012" or later.

\*6: The module whose first 5 digits of serial No. is "05032" or later.

\*8: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU

\*9: The Universal model QCPU except the Q00UJCPU.

TableApp.3.8 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM796	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.1)	OFF : Block is secured ON : Block set by SD796 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction(target CPU= CPU No.1) is less than the number of blocks specified by SD796. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	QnU <sup>7</sup>
SM797	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.2)	OFF : Block is secured ON : Block set by SD797 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.2) is less than the number of blocks specified by SD797. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	
SM798	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.3)	OFF : Block is secured ON : Block set by SD798 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.3) is less than the number of blocks specified by SD798. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	
SM799	Block information using multiple CPU high-speed transmission dedicated instruction (for CPU No.4)	OFF : Block is secured ON : Block set by SD799 cannot be secured	• Turns ON when the number of the remaining blocks of the dedicated instruction transmission area used for the multiple CPU high-speed transmission dedicated instruction (target CPU= CPU No.4) is less than the number of blocks specified by SD799. Turns ON at instruction execution. Turns OFF when the empty area exists at END processing.	S (When instruction/END processing executed)	New	

\*7: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

## (8) Debug

TableApp.3.9 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM800	Trace preparation	OFF : Not ready ON : Ready	• Switches ON when the trace preparation is completed	S (Status change)	New	Qn(H) QnPH QnPRH QnU <sup>1</sup>
SM801	Trace start	OFF : Suspend ON : Start	• Trace is started when this relay switches ON. • Trace is suspended when this relay switches OFF. (All related special Ms switches OFF.)	U	M9047	
SM802	Trace execution in progress	OFF : Suspend ON : Start	• Switches ON during execution of trace.	S (Status change)	M9046	
SM803	Trace trigger	OFF → ON: Start	• Trace is triggered when this relay switches from OFF to ON. (Identical to TRACE instruction execution status)	U	M9044	
SM804	After trace trigger	OFF : Not after trigger ON : After trigger	• Switches ON after trace is triggered.	S (Status change)	New	
SM805	Trace completed	OFF : Not completed ON : End	• Switches ON at completion of trace	S (Status change)	M9043	
SM826	Trace error	OFF : Normal ON : Errors	• Switches ON if error occurs during execution of trace	S (Status change)	New	
SM829	Forced registration specification of trace setting	ON : Forced registration enabled OFF : Forced registration disabled	• Even when the trace condition or the trigger condition is established, the sampling trace setting can be set to the CPU module by turning SM829 ON and registering the sampling trace setting by GX Developer.	U	New	QnU <sup>1</sup>

\*1: The Universal model QCPU except the Q00UJCPU.

(9) A to Q conversion correspondences

Special relays SM1000 to SM1255 are the relays which correspond to ACPU special relays M9000 to M9255 after A to Q conversion.

(However, the Basic model QCPU and Redundant CPU do not support the A to Q conversion.)

These special relays are all set by the system, and cannot be set by the user program.

To turn them ON/OFF by the user program, change the special relays in the program into those of QCPU.

However, some of SM1084 and SM1200 to SM1255 (corresponding to M9084 and M9200 to M9255 before conversion) can be turned ON/OFF by the user program, if they could be turned ON/OFF by the user program before conversion. For details on the ACPU special relays, see the user's manuals for the individual CPUs, and MELSECNET or MELSECNET/B Data Link System Reference Manuals

**POINT**

Check "Use special relay/special register from SM/SD1000" for "A-PLC" on the PLC system tab of PLC parameter in GX Developer when the converted special relays are used with the High Performance model QCPU, Process CPU, and Universal model QCPU.

When not using the converted special relays, uncheck "Use special relay/special register from SM/SD1000" to save the time taken for processing special relays.

**Remark**

The following are additional explanations about the Special Relay for Modification column.

- ① When a special relay for modification is provided, the device number should be changed to the provided QCPU special relay.
- ② When  is provided, the converted special relay can be used for the device number.
- ③ When  is provided, the device number does not work with QCPU.

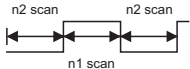
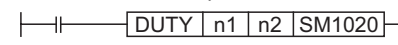

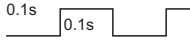
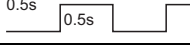

TableApp.3.10 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9000	SM1000	—	Fuse blown	OFF : Normal ON : Module with blown fuse	<ul style="list-style-type: none"> <li>• Turned on when there is one or more output modules of which fuse has been blown.</li> <li>• Remains ON if the condition is restored to normal thereafter.</li> <li>• Output modules of remote I/O stations are also checked for fuse condition.</li> </ul>	Qn(H) QnPH QnU*1
M9002	SM1002	—	I/O module verify error	OFF : Normal ON : Error	<ul style="list-style-type: none"> <li>• Turned on if the status of I/O module is different from entered status when power is turned on.</li> <li>• Remains ON if the condition is restored to normal thereafter.</li> <li>• I/O module verification is done also to remote I/O station modules.</li> <li>• Reset is enabled only when special registers SD1116 to SD1123 are reset.</li> </ul>	

\*1: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU

TableApp.3.11 Special relay



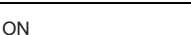

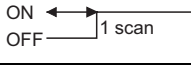
ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU	
M9005	SM1005	—	AC DOWN detection	OFF : AC DOWN not detected ON : AC DOWN detected	<ul style="list-style-type: none"> <li>Turns ON if an instantaneous power failure of within 20ms occurs during use of the AC power supply module.</li> <li>Reset when the power supply is switched OFF, then ON.</li> <li>Turns ON if an instantaneous power failure of within 10ms occurs during use of the DC power supply module.</li> <li>Reset when the power supply is switched OFF, then ON.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>	
M9006	SM1006	—	Battery low	OFF : Normal ON : Battery low	<ul style="list-style-type: none"> <li>Turns ON when the battery voltage drops to or below the specified.</li> <li>Turns OFF when the battery voltage returns to normal thereafter.</li> </ul>		
M9007	SM1007	—	Battery low latch	OFF : Normal ON : Battery low	<ul style="list-style-type: none"> <li>Turns ON when the battery voltage drops to or below the specified.</li> <li>Remains ON if the battery voltage returns to normal thereafter.</li> </ul>		
M9008	SM1008	SM1	Self-diagnosis error	OFF : No error ON : Error	<ul style="list-style-type: none"> <li>Turned on when error is found as a result of self-diagnosis.</li> </ul>		
M9009	SM1009	SM62	Annunciator detection	OFF : No F number detected ON : F number detected	<ul style="list-style-type: none"> <li>Turned on when OUT F of SET F instruction is executed.</li> <li>Switched off when SD1124 data is cleared to zero.</li> </ul>		
M9011	SM1011	SM56	Operation error flag	OFF : No error ON : Error	<ul style="list-style-type: none"> <li>Turned on when operation error occurs during execution of application instruction.</li> <li>Remains ON if the condition is restored to normal thereafter.</li> </ul>		
M9012	SM1012	SM700	Carry flag	OFF : Carry OFF ON : Carry ON	<ul style="list-style-type: none"> <li>Carry flag used in application instruction.</li> </ul>		
M9016	SM1016	×	Data memory clear flag	OFF : Ignored ON : Output cleared	<ul style="list-style-type: none"> <li>Clears the data memory including the latch range (other than special relays and special registers) in remote run mode from computer, etc. when SM1016 is on.</li> </ul>		Qn(H) QnPH
M9017	SM1017	×	Data memory clear flag	OFF : Ignored ON : Output cleared	<ul style="list-style-type: none"> <li>Clears the unlatched data memory (other than special relays and special registers) in remote run mode from computer, etc. when SM1017 is on.</li> </ul>		
M9020	SM1020	—	User timing clock No.0	 <p>n1 scan</p> <p>n2 scan</p>	<ul style="list-style-type: none"> <li>Relay which repeats on/off at intervals of predetermined scan.</li> <li>When power is turned on or reset is performed, the clock starts with off.</li> <li>Set the intervals of on/off by DUTY instruction.</li> </ul> <p style="text-align: center;">  </p> <p>n1: ON scan interval n2: OFF scan interval</p> <p>* : If DUTY instruction, which specified from SM 1020 to SM 1024 of User timing clock in programs other than a program for a Universal model QCPU, changes the programmable controller to the Universal model QCPU, the special relays SM 420 to 424 will be replaced. (Universal model QCPU cannot specify the special relays from SM 1020 to SM1024.)</p>	Qn(H) QnPH QnU <sup>*1</sup>	
M9021	SM1021	—	User timing clock No.1				
M9022	SM1022	—	User timing clock No.2				
M9023	SM1023	—	User timing clock No.3				
M9024	SM1024	—	User timing clock No.4				
M9025	SM1025	—	Clock data set request	OFF : Ignored ON : Set request present used	<ul style="list-style-type: none"> <li>Writes the clock data stored in SD1025 to SD1028 to the CPU module after the END instruction is executed in the scan in which SM1025 turned from OFF to ON.</li> </ul>	Qn(H) QnPH	
M9026	SM1026	—	Clock data error	OFF : No error ON : Error	<ul style="list-style-type: none"> <li>Switched on by clock data (SD1025 to SD1028) error</li> </ul>		
M9028	SM1028	—	Clock data read request	OFF : Ignored ON : Read request	<ul style="list-style-type: none"> <li>Reads clock data to SD1025 to SD1028 in BCD when SD1028 is on.</li> </ul>		
M9029	SM1029	×	Batch processing of data communications requests	OFF : Batch processing not conducted ON : Batch processing conducted	<ul style="list-style-type: none"> <li>The SM1029 relay is turned on using a sequence program to process all data communication requests accepted during one scan in the END processing of that scan.</li> <li>The batch processing of the data communication requests can be turned on and off during running.</li> <li>The default is OFF (processed one at a time for each END processing in the order in which data communication requests are accepted).</li> </ul>	Qn(H) QnPH	
M9030	SM1030	—	0.1 second clock		<ul style="list-style-type: none"> <li>0.1 second, 0.2 second, 1 second and 2 second, clocks are generated.</li> <li>Not turned on or off per scan but turned on and off even during scan if corresponding time has elapsed.</li> <li>Starts with off when Programmable Controller power supply is turned on or CPU module reset is performed.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>	
M9031	SM1031	—	0.2 second clock				
M9032	SM1032	—	1 second clock				
M9033	SM1033	—	2 second clock				

\*1: The relevant modules are as follows:  
 • The Universal model QCPU whose serial number (first five digits) is "10102" or later.  
 • Q00UJCPU, Q00UCPU, Q01UCPU

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9034	SM1034	—	2n minute clock(1 minute clock) <sup>*2</sup>	ns 	<ul style="list-style-type: none"> <li>Alternates between ON and OFF according to the seconds specified at SD414. (Default: n = 30)</li> <li>Not turned on or off per scan but turned on and off even during scan if corresponding time has elapsed.</li> <li>Starts with off when Programmable Controller power supply is turned on or CPU module reset is performed.</li> </ul>	
M9036	SM1036	—	Always ON	ON  OFF	<ul style="list-style-type: none"> <li>Used as dummy contacts of initialization and application instruction in sequence program.</li> <li>SM1038 and SM1037 are turned on and off without regard to position of key switch on CPU module front. SM1038 and SM1039 are under the same condition as RUN status except when the key switch is at STOP position, and turned off and on. Switched off if the key switch is in STOP position. SM1038 is on for one scan only and SM1039 is off for one scan only if the key switch is not in STOP position.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
M9037	SM1037	—	Always OFF	ON  OFF		
M9038	SM1038	—	ON for 1 scan only after RUN	ON  OFF		
M9039	SM1039	—	RUN flag(After RUN, OFF for 1 scan only)	ON  OFF		
M9040	SM1040	SM206	PAUSE enable coil	OFF : PAUSE disabled ON : PAUSE enabled	<ul style="list-style-type: none"> <li>When RUN key switch is at PAUSE position or pause contact has turned on and if SM1040 is on, PAUSE mode is set and SM1041 is turned on.</li> </ul>	Qn(H) QnPH
M9041	SM1041	SM204	PAUSE status contact	OFF : PAUSE not in effect ON : PAUSE in effect		
M9042	SM1042	SM203	STOP status contact	OFF : STOP not in effect ON : STOP in effect	<ul style="list-style-type: none"> <li>Switched on when the RUN key switch or RUN/STOP switch is in STOP position.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
M9043	SM1043	SM805	Sampling trace completed	OFF : Sampling trace in progress ON : Sampling trace completed	<ul style="list-style-type: none"> <li>Turned on upon completion of sampling trace performed the number of times preset by parameter after STRA instruction is executed. Reset when STRAR instruction is executed.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
M9044	SM1044	SM803	Sampling trace	OFF → ON Same as STRA instruction execution ON → OFF Same as STRAR instruction execution	<ul style="list-style-type: none"> <li>Turning on/off SM1044 can execute STRA/STRAR instruction. (SM1044 is forcibly turned on/off by a peripheral device.)</li> <li>When switched from OFF to ON: STRA instruction</li> <li>When switched from ON to OFF: STRAR instruction</li> <li>The value stored in SD1044 is used as the condition for the sampling trace.</li> <li>At scanning, at time → Time (10 ms unit)</li> </ul>	Qn(H) QnPH
M9045	SM1045	×	Watchdog timer (WDT) reset	OFF : Does not reset WDT ON : Resets WDT	<ul style="list-style-type: none"> <li>The SM1045 relay is turned on to reset the WDT when the ZCOM instruction and data communication request batch processing are executed (used when the scan time exceeds 200 ms).</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
M9046	SM1046	SM802	Sampling trace	OFF : Trace not in progress ON : Trace in progress	<ul style="list-style-type: none"> <li>Switched on during sampling trace.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
M9047	SM1047	SM801	Sampling trace preparations	OFF : Sampling trace suspended ON : Sampling trace started	<ul style="list-style-type: none"> <li>Sampling trace is not executed unless SM1047 is turned ON.</li> <li>Sampling trace is suspended when SM1047 goes OFF.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
M9049	SM1049	SM701	Switching the number of output characters	OFF : Output until NULL code encountered ON : 16 characters output	<ul style="list-style-type: none"> <li>When SM1049 is OFF, characters up to NULL (00H) code are output.</li> <li>When SM1049 is ON, ASCII codes of 16 characters are output.</li> </ul>	Qn(H) QnPH
M9051	SM1051	×	CHG instruction execution disable	OFF : Enabled ON : Disable	<ul style="list-style-type: none"> <li>Switched ON to disable the CHG instruction.</li> <li>Switched ON when program transfer is requested. Automatically switched OFF when transfer is complete.</li> </ul>	Qn(H) QnPH
M9052	SM1052	×	SEG instruction switch	OFF : 7SEG segment display ON : I/O partial refresh	<ul style="list-style-type: none"> <li>When SM1052 is ON, the SEG instruction is executed as an I/O partial refresh instruction.</li> <li>When SM1052 is OFF, the SEG instruction is executed as a 7-SEG display instruction.</li> </ul>	

\*1: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU

\*2: minute clock indicates the name of the special relay (M9034) of the ACP.

TableApp.3.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU
M9056	SM1056	×	Main side P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested	• Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	Qn(H) QnPH
M9057	SM1057	×	Sub side P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested		
M9058	SM1058	×	Main side P, I set completion	Momentarily ON at P, I set completion	• Turned ON once when the P, I set has been completed, and then turned OFF again.	
M9059	SM1059	×	Sub program P, I set completion	Momentarily ON at P, I set completion		
M9060	SM1060	×	Sub program 2 P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested	• Provides P, I set request after transfer of the other program (for example subprogram when main program is being run) is complete during run. Automatically switched off when P, I setting is complete.	
M9061	SM1061	×	Sub program 3 P, I set request	OFF : Other than when P, I set being requested ON : P, I set being requested		
M9070	SM1070	×	A8UPU/ A8PUJrequired search time <sup>*3</sup>	OFF : Read time not shortened ON : Read time shortened	• Turned ON to shorten the search time in the A8UPU/ A8PUJ. (In this case, the scan time is extended by 10 %.)	
M9084	SM1084	×	Error check	OFF : Error check executed ON : No error check	It is set whether the error checks below are performed or not when the END instruction is processed (to set the END instruction processing time). • Check for fuse blown. • Check of battery • Collation check of I/O module	
M9091	SM1091	×	Operation error details flag	OFF : No error ON : Error	• Turns ON when the detail factor of the operation error is stored into SD1091. • Remains ON if the condition is restored to normal thereafter.	
M9100	SM1100	SM320	Presence/absence of SFC program	OFF : SFC programs not used ON : SFC programs used	• Turned on if the SFC program is registered. • Turned off if the SFC program is not registered.	
M9101	SM1101	SM321	Start/stop SFC program	OFF : SFC programs stop ON : SFC programs start	• The value in SM1100 is set as the initial value. (The relay automatically turns ON when the SFC program is present.) • When this relay turns from ON to OFF, execution of the SFC program stops. • When this relay turns from OFF to ON, execution of the SFC program resumes.	
M9102	SM1102	SM322	SFC program start status	OFF : Initial start ON : Resume start	• The SFC program start mode in the SFC setting of the PLC parameter dialog box is set as the initial value. At initial start: OFF At continue start: ON	

\*3: The A8UPU/A8PUJ is not available for the QCPU/QnACPU.

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.11 Special relay

ACPU Special Relay	Special Relay after Conversion	Special Relay for Modification	Name	Meaning	Details	Corresponding CPU		
M9103	SM1103	SM323	Presence/absence of continuous transition	OFF : Continuous transition not effective ON : Continuous transition effective	• Set whether continuous transition will be performed for the block where the "continuous transition bit" of the SFC information device is not set.	Qn(H) QnPH		
M9104	SM1104	SM324	Continuous transition suspension flag	OFF : When transition is completed ON : When no transition	• OFF during operation in the continuous transition mode or during continuous transition, and ON when continuous transition is not executed. • Always ON during operation in the no continuous transition mode.			
M9108	SM1108	SM90	Step transition monitoring timer start (equivalent of SD90)	OFF : Monitoring timer reset ON : Monitoring timer reset start	• Turns ON when the measurement of the step transition monitoring timer is started. Turning this relay OFF resets the step transition monitoring timer.			
M9109	SM1109	SM91	Step transition monitoring timer start (equivalent of SD91)					
M9110	SM1110	SM92	Step transition monitoring timer start (equivalent of SD92)					
M9111	SM1111	SM93	Step transition monitoring timer start (equivalent of SD93)					
M9112	SM1112	SM94	Step transition monitoring timer start (equivalent of SD94)					
M9113	SM1113	SM95	Step transition monitoring timer start (equivalent of SD95)					
M9114	SM1114	SM96	Step transition monitoring timer start (equivalent of SD96)					
M9196	SM1196	SM325	Operation output at block stop	OFF : Coil output OFF ON : Coil output ON	• Selects the operation output when block stop is executed. ON : Retains the ON/OFF status of the coil being used by using operation output of the step being executed at block stop. OFF : All coil outputs are turned off. (Operation output by the SET instruction is retained regardless of the ON/OFF status of SM1196.)			
M9197	SM1197	×	Switch between blown fuse and I/O verify error display	SM 1197	SM 1198		I/O numbers to be displayed	Switches I/O numbers in the fuse blow module storage registers (SD1100 to SD1107) and I/O module verify error storage registers (SD1116 to SD1123) according to the combination of ON/OFF of the SM1197 and SM1198.
M9198	SM1198	×		OFF	OFF		X/Y0 to 7F0	
				ON	OFF		X/Y800 to FF0	
				OFF	ON		X/Y1000 to 17F0	
M9199	SM1199	×	ON	ON	X/Y1800 to 1FF0			
			OFF	ON	X/Y1800 to 1FF0			
M9199	SM1199	×	Data recovery of online sampling trace/status latch	OFF : Data recovery disabled ON : Data recovery enabled	• Recovers the setting data stored in the CPU module at restart when sampling trace/status latch is executed. • SM1199 should be ON to execute again. (Unnecessary when writing the data again from peripheral devices.)			

## (10) QCPU with built-in Ethernet port

TableApp.3.12 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM1270	Time setting function (SNTP client) execution	OFF : No time setting function (SNTP client) execution ON : Time setting function (SNTP client) execution	Set this to ON when executing the time setting function (SNTP client). (Only when the time setting function is in "Use" with the time setting parameter.)	U	New	QnU*1
SM1273	Remote password mismatch count clear	OFF : Normal ON : Clear	To clear the accumulated number (SD979 to 999) of mismatched remote passwords, the setting SM1273 is executed.	U	New	

\* 1: This applies to the Built-in Ethernet port QCPU.



(11) Process control instructions

TableApp.3.13 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM1500	Hold mode	OFF : No-hold ON : Hold	• Specifies whether or not to hold the output value when a range over occurs for the S.IN instruction range check.	U	New	QnPH QnPRH
SM1501	Hold mode	OFF : No-hold ON : Hold	• Specifies whether or not the output value is held when a range over occurs for the S.OUT instruction range check.	U	New	

(12) For redundant systems (Host system CPU information \*1)

SM1510 to SM1599 are only valid for redundant systems.

All off for standalone systems.

TableApp.3.14 Special relay


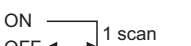
Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU												
SM1510	Operation mode	OFF : Redundant system backup mode, stand-alone system ON : Redundant system separate mode	• Turns on when the operating mode is redundant system separate.	S (Each END)	New	QnPRH												
SM1511	System A identification flag	<table border="1"> <tr> <td></td> <td>System A</td> <td>System B</td> <td>When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)</td> </tr> <tr> <td>SM1511</td> <td>ON</td> <td>OFF</td> <td>OFF</td> </tr> <tr> <td>SM1512</td> <td>OFF</td> <td>ON</td> <td>OFF</td> </tr> </table>		System A	System B		When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)	SM1511	ON	OFF	OFF	SM1512	OFF	ON	OFF	• Distinguishes between system A and system B. • The flag status does not change even if the tracking cable is disconnected.	S (Initial)	New
	System A		System B	When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)														
SM1511	ON		OFF	OFF														
SM1512	OFF	ON	OFF															
SM1512	System B identification flag																	
SM1513	Debug mode status flag	OFF : Not in debug mode ON : Debug mode	• Turns on when the redundant system operating mode is set to debug mode.	S (Initial)	New													
SM1515	Control system judgment flag	<table border="1"> <tr> <td></td> <td>Control system</td> <td>Standby system</td> <td>When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)</td> </tr> <tr> <td>SM1515</td> <td>ON</td> <td>OFF</td> <td>OFF</td> </tr> <tr> <td>SM1516</td> <td>OFF</td> <td>ON</td> <td>OFF</td> </tr> </table>		Control system	Standby system	When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)	SM1515	ON	OFF	OFF	SM1516	OFF	ON	OFF	• Indicates operation system status. • The flag status does not change even if the tracking cable is disconnected.	S (Status change)	New	
	Control system		Standby system	When TRK. CABLE ERR. (error code: 6210) occurs (Unknown)														
SM1515	ON	OFF	OFF															
SM1516	OFF	ON	OFF															
SM1516	Standby system judgment flag																	

\*1: The information of the host CPU module is stored.

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.13 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM1517	CPU module startup status	OFF : Power supply on startup ON : Operation system switch start up	• Turns on when the CPU module is started up by the system switching (switching from the standby system to the control system). Remains OFF when the standby system is switched to the control system by a power-ON startup.	S (Status change)	New	
SM1518	Standby system to control system switching status flag	ON  OFF	• Turns ON once switch between standby system to control system, (ON for 1 scan only) occurs. • This status flag can only be used for scan execution type programs.	S (Each END)	New	
SM1519	Previous Control System Identification Flag	ON  OFF	• On the last operation Control System was System B, if power supply is supplied, or reset is released on both SYSTEM together, After RUN, ON for 1 scan only by System A side.	S (Each END)	New	
SM1520	Data tracking transfer trigger specification	OFF : No trigger ON : Trigger	SM1520 Block 1	<ul style="list-style-type: none"> <li>When data is transferred based on the tracking setting of the redundant parameter dialog box, the target block is specified as trigger.</li> <li>When "Auto Tracking block No.1" is enabled in the tracking setting, SM1520 is turned ON by the system at power ON/ STOP to RUN. In other cases, SM1520 to SM1583 are turned ON by the user.</li> </ul>	S (initial)/U	New
SM1521			Block 2			
SM1522			Block 3			
SM1523			Block 4			
SM1524			Block 5			
SM1525			Block 6			
SM1526			Block 7			
SM1527			Block 8			
SM1528			Block 9			
SM1529			Block 10			
SM1530			Block 11			
SM1531			Block 12			
SM1532			Block 13			
SM1533			Block 14			
SM1534			Block 15			
SM1535			Block 16			
SM1536			Block 17			
SM1537			Block 18			
SM1538			Block 19			
SM1539			Block 20			
SM1540			Block 21			
SM1541			Block 22			
SM1542			Block 23			
SM1543			Block 24			
SM1544			Block 25			
SM1545			Block 26			
SM1546			Block 27			
SM1547			Block 28			
SM1548			Block 29			

QnPRH

TableApp.3.13 Special relay

Number	Name	Meaning	Explanation		Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU
SM1549	Data tracking transfer trigger specification	OFF : No trigger ON : Trigger	SM1549	Block 30	<ul style="list-style-type: none"> <li>When data is transferred based on the tracking setting of the redundant parameter dialog box, the target block is specified as trigger.</li> <li>When "Auto tracking block No. 1" is enabled in the tracking setting, SM1520 is turned ON by the system at power ON/ STOP to RUN. In other cases, SM1520 to SM1583 are turned ON by the user.</li> </ul>	S (initial)/U	New
SM1550			Block 31				
SM1551			Block 32				
SM1552			Block 33				
SM1553			Block 34				
SM1554			Block 35				
SM1555			Block 36				
SM1556			Block 37				
SM1557			Block 38				
SM1558			Block 39				
SM1559			Block 40				
SM1560			Block 41				
SM1561			Block 42				
SM1562			Block 43				
SM1563			Block 44				
SM1564			Block 45				
SM1565			Block 46				
SM1566			Block 47				
SM1567			Block 48				
SM1568			Block 49				
SM1569			Block 50				
SM1570			Block 51				
SM1571			Block 52				
SM1572			Block 53				
SM1573			Block 54				
SM1574			Block 55				
SM1575			Block 56				
SM1576			Block 57				
SM1577			Block 58				
SM1578			Block 59				
SM1579			Block 60				
SM1580			Block 61				
SM1581			Block 62				
SM1582			Block 63				
SM1583	Block 64						
SM1590	System switching enable/disable flag from network module	OFF : System switching request issuing module absent ON : System switching request issuing module present	<ul style="list-style-type: none"> <li>Turns ON when a system switching request is issued from the network module. The module No. that issued system switching can be checked by SD1590.</li> <li>Turns OFF when all bits of SD1590 are OFF.</li> </ul>		S (Each END)	New	QnPRH
SM1591	Standby system error detection disable flag at system switching	ON : Error is not detected by new standby system at system switching OFF : Error is detected by new standby system at system switching	This flag is used to determine if the new standby station detects 6210:STANDBY during system switching. This applies to the following switching methods: <ul style="list-style-type: none"> <li>System switching from GX Developer</li> <li>System switching using dedicated instruction</li> <li>System switching by the intelligent function module</li> </ul>		U	New	
SM1592	Enable/disable user system switching	OFF : Disable user system switching ON : Enable user system switching	<ul style="list-style-type: none"> <li>This flag enables system switching by the user from GX Developer or by dedicated instruction. (SP.CONTSW).</li> </ul>		U	New	

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.13 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM1593	Setting to access extension base unit of standby system CPU	OFF : Error ON : Ignored	Sets the operation for the case accessing buffer memory of the intelligent function module mounted on the extension base unit from the standby system CPU in separate mode. OFF : "OPERATION ERROR" (error code: 4112) will be returned when accessing buffer memory of the intelligent function module on the extension base unit from the standby system CPU. ON : No processing is performed when accessing buffer memory of intelligent function module on the extension base unit from the standby system CPU.	U	New	QnPRH <sup>*2</sup>
SM1595	Memory copy to other system start flag	OFF : Start memory copy ON : No memory copy initiated	• When SM1595 is turned from OFF to ON, memory copy from control system to standby system starts. Note that when SM1595 is turned from OFF to ON, memory copy does not start if the I/O No. of the copy destination (standby system CPU module: 3D1H) is not stored in SD1595.			
SM1596	Memory copy to other system status flag	OFF : Memory copy not executed ON : Memory copy executed	• Turns on while memory is copied to other system. • Turns off when memory copy execution has completed.	S (Starting to copy/finish)		QnPRH
SM1597	Memory copy to other system completion flag	OFF : Memory copy not completed ON : Memory copy completed	• Turns on once the memory copying to the other system has completed.	S (finish)/U	New	
SM1598	Copy contents of standard ROM during memory copy	OFF : Copy standard ROM data ON : Standard ROM data is not copied	• If set to on by user, the standard ROM data is not copied to the other system while memory copy is executing.	U		

\*2: The module whose first 5 digits of serial No. is "09012" or later.

(13) For redundant system (Other system CPU information \*1)

SM1600 to SM1650 only valid for the CPU redundant system backup mode, so they cannot be refreshed during the separate mode.

Either the backup mode or the separate mode is valid for the SM4651 to SM1699.

SM1600 to SM1699 are all turned off for stand-alone system.

TableApp.3.14 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding Host SM□□*2	Corresponding CPU
SM1600	Other system error flag	OFF : No error ON : Error	• Turns on when an error occurs during redundant system. Error check (Turns on single bit of SD1600.) • Is off when no errors are present	S (Each END)	—	QnPRH
SM1610	Other system diagnostics error	OFF : No error ON : Error	• Turns on when a diagnostics error occurs. (Includes error detection when annunciator is ON, and by CHK instruction) • Corresponds to status of SM0 at other system	S (Each END)	SM0	
SM1611	Other systems self diagnostics error.	OFF : No self diagnostics error occurred ON : Self diagnostics error occurred	• Turns on when a self diagnostics error occurs. (Does not include error detection when annunciator is ON, and by CHK instruction) • Corresponds to status of SM1 at other system	S (Each END)	SM1	
SM1615	Other system common error information	OFF : No common error information present ON : Common error information present	• Turns on when there is common error information at other system • Corresponds to status of SM5 at other system	S (Each END)	SM5	QnPRH
SM1626	Error individual information for other systems	OFF : No individual error information present ON : Individual error information present	• Turns on when there is individual error information at other system • Corresponds to status of SM16 at other system	S (Each END)	SM16	
SM1649	Standby system cancel error flag	OFF to ON: Cancels error of standby system	By turning this relay from OFF to ON, the continue error that occurred in the standby system CPU module can be canceled. Use SD1649 to specify the error code of the error to be canceled.	U	—	

\*1 Stores other system CPU diagnostic information and system information.

\*2 This shows the special relay(SM□□) for the host system CPU.

(14) For redundant system (tracking)

Either the backup mode or the second mode is valid for SM1700 to SM1799.

All is turned off for stand-alone system.

TableApp.3.15 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU	Corresponding CPU
					M9□□□	
SM1700	Transfer trigger completion flag	OFF : Transfer not completed ON : Transfer completed	• Turns on for one scan, once transfer of block 1 to block 64 is completed.	S (status change)	New	QnPRH
SM1709	Manual system switching disable/enable setting during online program change redundant tracking	ON : Manual system switching enabled (Disable canceled) OFF : Manual system switching disabled	(1) Turning this relay from OFF to ON enables manual system switching during online program change redundant tracking. After the manual system switching disable status is canceled, the system automatically turns off SM1709. (2) System switching due to any of the following conditions is executed even during online program change redundant tracking, regardless of the status of this relay. •Power off, reset, hardware failure, CPU stop error (3) In either of the following statuses, the system switching disable status can also be canceled by this relay. •Multiple-block online program change redundant tracking execution status •File batch online program change redundant tracking execution status	S (When executed)/U		
SM1710	Transfer tracking data during online program change enable flag	OFF : No device tracking ON : Transfer device memory	(1) Set whether the tracking of the following data will be executed or not during online program change redundant tracking. •Device memory (Including SM/SD that will automatically execute tracking) •PIDINIT information, S.PIDINIT information, SFC information (2) SM1710 can be also used to set whether tracking will be executed or not while online change of multiple program blocks or batch of files is being performed to ensure consistency of both systems. (3) This SM is also transferred from control system CPU module to standby system CPU module by tracking data.	U		

A

Appendix 3 SPECIAL RELAY LIST

TableApp.3.15 Special relay

Number	Name	Meaning	Explanation		Set by (When Set)	Corres- ponding ACPU M9□□□	Corresponding CPU	
SM1712	Transfer trigger completion flag	OFF : Transfer uncompleted ON : Transfer completed	SM1712	Block 1	Turns ON only during one scan when the transmission of the corresponding block is completed.	S (status change)	New	QnPRH
SM1713			SM1713	Block 2				
SM1714			SM1714	Block 3				
SM1715			SM1715	Block 4				
SM1716			SM1716	Block 5				
SM1717			SM1717	Block 6				
SM1718			SM1718	Block 7				
SM1719			SM1719	Block 8				
SM1720			SM1720	Block 9				
SM1721			SM1721	Block 10				
SM1722			SM1722	Block 11				
SM1723			SM1723	Block 12				
SM1724			SM1724	Block 13				
SM1725			SM1725	Block 14				
SM1726			SM1726	Block 15				
SM1727			SM1727	Block 16				
SM1728			SM1728	Block 17				
SM1729			SM1729	Block 18				
SM1730			SM1730	Block 19				
SM1731			SM1731	Block 20				
SM1732			SM1732	Block 21				
SM1733			SM1733	Block 22				
SM1734			SM1734	Block 23				
SM1735			SM1735	Block 24				
SM1736			SM1736	Block 25				
SM1737			SM1737	Block 26				
SM1738			SM1738	Block 27				
SM1739			SM1739	Block 28				
SM1740			SM1740	Block 29				
SM1741			SM1741	Block 30				
SM1742			SM1742	Block 31				
SM1743			SM1743	Block 32				
SM1744			SM1744	Block 33				
SM1745			SM1745	Block 34				
SM1746			SM1746	Block 35				
SM1747			SM1747	Block 36				
SM1748			SM1748	Block 37				
SM1749			SM1749	Block 38				
SM1750			SM1750	Block 39				
SM1751			SM1751	Block 40				
SM1752			SM1752	Block 41				
SM1753			SM1753	Block 42				
SM1754			SM1754	Block 43				
SM1755			SM1755	Block 44				
SM1756			SM1756	Block 45				
SM1757			SM1757	Block 46				
SM1758			SM1758	Block 47				
SM1759			SM1759	Block 48				

TableApp.3.15 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU	
SM1760	Transfer trigger completion flag	OFF : Transmission uncompleted ON : Transmission end	SM1760	Block 49	S (status change)	New	QnPRH
SM1761			Block 50				
SM1762			Block 51				
SM1763			Block 52				
SM1764			Block 53				
SM1765			Block 54				
SM1766			Block 55				
SM1767			Block 56				
SM1768			Block 57				
SM1769			Block 58				
SM1770			Block 59				
SM1771			Block 60				
SM1772			Block 61				
SM1773			Block 62				
SM1774			Block 63				
SM1775			Block 64				

## (15) Redundant power supply module information

TableApp.3.16 Special relay

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU M9□□□	Corresponding CPU
SM1780	Power supply off detection flag	OFF : No redundant power supply module with input power OFF detected ON : Redundant power supply module with input power OFF detected	<ul style="list-style-type: none"> <li>Turns ON when one or more redundant power supply modules with input power OFF are detected.</li> <li>Turns on if any of SD1780 bits is on.</li> <li>Turns off if all bits of SD1780 are off.</li> <li>Turns OFF when the main base unit is not the redundant main base unit (Q38RB).</li> <li>When the multiple CPU system is configured, the flags are stored only to the CPU No.1.</li> </ul>	S (Each END)	New	Qn(H) <sup>*2</sup> QnPH <sup>*2</sup> QnPRH QnU <sup>*3</sup>
SM1781	Power supply failure detection flag	OFF : No faulty redundant power supply module detected ON : Faulty redundant power supply module detected	<ul style="list-style-type: none"> <li>Turns ON when one or more faulty redundant power supply modules are detected.</li> <li>Turns on if any of SD1781 bits is on.</li> <li>Turns off if all bits of SD1781 are off.</li> <li>Turns OFF when the main base unit is not the redundant main base unit (Q38RB).</li> <li>When the multiple CPU system is configured, the flags are stored only to the CPU No.1.</li> </ul>	S (Each END)		
SM1782	Momentary power failure detection flag for power supply 1 <sup>*1</sup>	OFF : No momentary power failure detected ON : Momentary power failure detected	<ul style="list-style-type: none"> <li>Turns ON when a momentary power failure of the input power supply to the power supply 1 or 2 is detected one or more times. After turning ON, remains ON even if the power supply recovers from the momentary power failure.</li> <li>Turns OFF the flag (SM1782, SM1783) of the power supply 1/2 when the CPU module starts.</li> <li>When the input power to one of the redundant power supply modules turns OFF the corresponding flag turns OFF.</li> <li>Turns OFF when the main base unit is not the redundant main base unit (Q38RB).</li> <li>When the multiple CPU system is configured, the flags are stored only to the CPU No.1.</li> </ul>	S (Each END)		
SM1783	Momentary power failure detection flag for power supply 2 <sup>*1</sup>					

\*1: The "power supply 1" indicates the redundant power supply module mounted on the POWER 1 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).  
The "power supply 2" indicates the redundant power supply module mounted on the POWER 2 slot of the redundant base unit (Q38RB/Q68RB/Q65WRB).

\*2: The module whose first 5 digits of serial No. is "04012" or later.

However, for the multiple CPU system configuration, this applies to all CPU modules whose first 5 digits of serial No. are "07032" or later.

\*3: The module whose first 5 digits of serial No. is "10042" or later.

# Appendix 4 SPECIAL REGISTER LIST

The special registers, SD, are internal registers with fixed applications in the Programmable Controller.

For this reason, it is not possible to use these registers in sequence programs in the same way that normal registers are used.

However, data can be written as needed in order to control the CPU modules.

Data stored in the special registers are stored as BIN values if no special designation has been made to the contrary.

The heading descriptions in the following special register lists are shown in 4.1.

TableApp.4.1 Descriptions of the special register lists headings

Item	Function of Item
Number	• Indicates special register number
Name	• Indicates name of special register
Meaning	• Indicates contents of special register
Explanation	• Discusses contents of special register in more detail
Set by (When set)	<ul style="list-style-type: none"> <li>• Indicates whether the relay is set by the system or user, and, if it is set by the system, when setting is performed.</li> </ul> <Set by> S : Set by system U : Set by user (sequence programs or test operations from GX Developer) S/U : Set by both system and user <When set> Indicated only for registers set by system Each END : Set during each END processing Initial : Set only during initial processing (when power supply is turned ON, or when going from STOP to RUN) Status change : Set only when there is a change in status Error : Set when error occurs Instruction execution : Set when instruction is executed Request : Set only when there is a user request (through SM, etc.) System switching : Set when system switching is executed.
Corresponding ACPU M9□□□	<ul style="list-style-type: none"> <li>• Indicates corresponding special register in ACPU</li> <li>(When the contents are changed, the special register is represented D9□□□ format change. Incompatible with the Q00J/Q00/Q01 and QnPRH.)</li> <li>• New indicates the special register newly added to the Q series CPU module.</li> </ul>
Corresponding CPU	Indicates the relevant CPU module. QCPU : Indicates all the Q series CPU modules. Q00J/Q00/Q01 : Indicates the Basic model QCPU. Qn(H) : Indicates the High Performance model QCPU. QnPH : Indicates the Process CPU. QnPRH : Indicates the Redundant CPU. QnU : Indicates the Universal model QCPU Each CPU type name : Can be applied only to the specific CPU. (e.g. Q02U)

For details on the following items, refer to the following manuals:

- Networks → Manual of the corresponding network module
- SFC → QCPU(Q mode)/QnACPU Programming Manual (SFC)

## POINT

Do not change the values of special relays set by the system with user program or device test operations.

Doing so may result in system downtime or communication fault.



(1) Diagnostic Information

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU		
SD0	Diagnostic errors	Diagnosis error code	<ul style="list-style-type: none"> <li>Error codes for errors found by diagnosis are stored as BIN data.</li> <li>Contents identical to latest fault history information.</li> </ul>	S (Error)	D9008 format change			
SD1	Clock time for diagnosis error occurrence	Clock time for diagnosis error occurrence	<ul style="list-style-type: none"> <li>Year (last two digits) and month that SD0 data was updated is stored as BCD 2-digit code.</li> </ul> <p>b15 to b8 b7 to b0 (Example) October, 1995  <span style="border: 1px solid black; padding: 2px;">Year (0 to 99)</span> <span style="border: 1px solid black; padding: 2px;">Month (1 to 12)</span> 9510<sub>H</sub></p>	S (Error)	New			
SD2			<ul style="list-style-type: none"> <li>The day and hour that SD0 was updated is stored as BCD 2-digit code.</li> </ul> <p>b15 to b8 b7 to b0 (Example) 10 a.m. on 25th  <span style="border: 1px solid black; padding: 2px;">Day (1 to 31)</span> <span style="border: 1px solid black; padding: 2px;">Hour (0 to 23)</span> 2510<sub>H</sub></p>					
SD3			<ul style="list-style-type: none"> <li>The minute and second that SD0 data was updated is stored as BCD 2-digit code.</li> </ul> <p>b15 to b8 b7 to b0 (Example) 35 min. 48 sec.  <span style="border: 1px solid black; padding: 2px;">Minutes (0 to 59)</span> <span style="border: 1px solid black; padding: 2px;">Seconds (0 to 59)</span> 3548<sub>H</sub></p>					
SD4	Error information categories	Error information category code	<p>Category codes which help indicate what type of information is being stored in the common information areas (SD5 through SD15) and the individual information areas (SD16 through SD26) are stored here. The category code for judging the error information type is stored.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">b15 to b8 Individual information category codes</td> <td style="text-align: center;">b7 to b0 Common information category codes</td> </tr> </table> <ul style="list-style-type: none"> <li>The common information category codes store the following codes:                      0: No error                      1: Unit/module No./ CPU No./Base No.*                      2: File name/Drive name                      3: Time (value set)                      4: Program error location                      5: System switching cause (for Redundant CPU only)                      6: Reason(s) for tracking capacity excess error (specific to Redundant CPU)                      7: Base No./Power supply No. (The first 5 digits of serial number 10072 or higher are chosen for Universal model QCPU.)                      8: Tracking transmission data classification (specific to Redundant CPU)                      *: For a multiple CPU system that consists of the Basic model QCPU, High Performance model QCPU, Process CPU, Universal model QCPU the module number or CPU number is stored depending on the error that occurred.                      (Refer to the corresponding error code for which number has been stored.)                      CPU No. 1: 1, CPU No. 2: 2, CPU No. 3: 3, CPU No. 4: 4</li> <li>The individual information category codes store the following codes:                      0: No error                      1: (Empty)                      2: File name/Drive name                      3: Time (value actually measured)                      4: Program error location                      5: Parameter number                      6: Annunciator number                      7: CHK instruction failure No. (except for the Basic model QCPU and the Universal model QCPU)                      8: Reason(s) for system switching failure (specific to Redundant CPU)                      12: File diagnostic information (specific to the Universal model QCPU)                      13: Parameter No./CPU No. (specific to the Universal model QCPU)</li> </ul>	b15 to b8 Individual information category codes	b7 to b0 Common information category codes	S (Error)	New	QCPU
b15 to b8 Individual information category codes	b7 to b0 Common information category codes							

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																															
SD5	Error common information	Error common information	<ul style="list-style-type: none"> <li>Common information corresponding to the error codes (SD0) is stored here.</li> <li>The following ten types of information are stored here:</li> <li>The error common information type can be judged by the "common information category code" in SD4. (The values of the "common information category code" stored in SD4 correspond to following 1) to 8).)</li> </ul> <p>1) Slot No.</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Slot No./CPU No./Base No.*1, *2, *3,*4</td> </tr> <tr> <td>SD6</td> <td>I/O No.*5</td> </tr> <tr> <td>SD7</td> <td rowspan="10">(Empty)</td> </tr> <tr> <td>SD8</td> </tr> <tr> <td>SD9</td> </tr> <tr> <td>SD10</td> </tr> <tr> <td>SD11</td> </tr> <tr> <td>SD12</td> </tr> <tr> <td>SD13</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table> <p>*1: For a multiple CPU system that consists of the Basic model QCPU, High Performance model QCPU, Process CPU, Universal model QCPU, the slot number or CPU number is stored depending on the error that occurred. Slot 0 in the multiple CPU system is the one on the slot on the right of the rightmost CPU module. (Refer to the corresponding error code for which number has been stored.) No. 1 CPU: 1, No. 2 CPU: 2, No. 3 CPU: 3, No. 4 CPU: 4</p> <p>*2: If a fuse blown or I/O verify error occurred in the module loaded in the MELSECNET/H remote I/O station, the network number is stored into the upper 8 bits and the station number into the lower 8 bits. Use the I/O No. to check the module where the fuse blown or I/O verify error occurred.</p> <p>*3: 255 is stored into SD5 of the Basic model QCPU when an instruction, etc. has been executed for the module later than the one on the last slot where a module can be mounted.</p> <p>*4: Definitions of base No. and slot No. &lt;Base No.&gt; Value used to identify the base unit on which the CPU module has been mounted. The following shows the definition of the base No.</p> <table border="1"> <thead> <tr> <th>Base No.</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Indicates the main base unit mounted with the CPU module.</td> </tr> <tr> <td>1 to 7</td> <td>Indicates the extension base unit. The stage number setting made by the stage number setting connector on the extension base unit is the base No. When stage number setting is extension 1: Base No. = 1 when stage number setting is extension 7: Base No. = 7</td> </tr> </tbody> </table> <p>&lt;Slot No.&gt; Value used to identify the slot of each base unit and the module mounted on that slot.</p> <ul style="list-style-type: none"> <li>The I/O slot 0 (slot on the right side of the CPU slot) of the main base unit is defined as the slot of "Slot No. = 0".</li> <li>The slot Nos. are consecutively assigned to the slots of the base units in order of the main base unit and extension base units 1 to 7.</li> <li>When the number of base unit slots has been set in the I/O assignment setting of the PLC parameter dialog box, the slot Nos. are assigned for only the number of set slots.</li> </ul> <p>*5: When 0FFFFH is stored into SD6 (I/O No.), the I/O No. cannot be identified due to overlapping I/O No., etc. in the I/O assignment setting of the PLC parameter dialog box. Therefore, identify the error location using SD5.</p> <p>2) File name/Drive name</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> <th>(Example) File name =</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Drive</td> <td>ABCDEF GH. IJK</td> </tr> <tr> <td>SD6</td> <td rowspan="10">File name (ASCII code: 8 characters)</td> <td>b15 to b8 b7 to b0</td> </tr> <tr> <td>SD7</td> <td>42H(B) 41H(A)</td> </tr> <tr> <td>SD8</td> <td>44H(D) 43H(C)</td> </tr> <tr> <td>SD9</td> <td>46H(F) 45H(E)</td> </tr> <tr> <td>SD10</td> <td>48H(H) 47H(G)</td> </tr> <tr> <td>SD11</td> <td>49H(I) 2EH(.)</td> </tr> <tr> <td>SD12</td> <td>4BH(K) 4AH(J)</td> </tr> <tr> <td>SD13</td> <td rowspan="3">(Empty)</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table>	Number	Meaning	SD5	Slot No./CPU No./Base No.*1, *2, *3,*4	SD6	I/O No.*5	SD7	(Empty)	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	Base No.	Definition	0	Indicates the main base unit mounted with the CPU module.	1 to 7	Indicates the extension base unit. The stage number setting made by the stage number setting connector on the extension base unit is the base No. When stage number setting is extension 1: Base No. = 1 when stage number setting is extension 7: Base No. = 7	Number	Meaning	(Example) File name =	SD5	Drive	ABCDEF GH. IJK	SD6	File name (ASCII code: 8 characters)	b15 to b8 b7 to b0	SD7	42H(B) 41H(A)	SD8	44H(D) 43H(C)	SD9	46H(F) 45H(E)	SD10	48H(H) 47H(G)	SD11	49H(I) 2EH(.)	SD12	4BH(K) 4AH(J)	SD13	(Empty)	SD14	SD15	S (Error)	New	QCPU
Number				Meaning																																																	
SD5				Slot No./CPU No./Base No.*1, *2, *3,*4																																																	
SD6				I/O No.*5																																																	
SD7				(Empty)																																																	
SD8																																																					
SD9																																																					
SD10																																																					
SD11																																																					
SD12																																																					
SD13																																																					
SD14																																																					
SD15																																																					
Base No.					Definition																																																
0				Indicates the main base unit mounted with the CPU module.																																																	
1 to 7	Indicates the extension base unit. The stage number setting made by the stage number setting connector on the extension base unit is the base No. When stage number setting is extension 1: Base No. = 1 when stage number setting is extension 7: Base No. = 7																																																				
Number	Meaning	(Example) File name =																																																			
SD5	Drive	ABCDEF GH. IJK																																																			
SD6	File name (ASCII code: 8 characters)	b15 to b8 b7 to b0																																																			
SD7		42H(B) 41H(A)																																																			
SD8		44H(D) 43H(C)																																																			
SD9		46H(F) 45H(E)																																																			
SD10		48H(H) 47H(G)																																																			
SD11		49H(I) 2EH(.)																																																			
SD12		4BH(K) 4AH(J)																																																			
SD13		(Empty)																																																			
SD14																																																					
SD15																																																					
SD6																																																					
SD7																																																					
SD8																																																					
SD9																																																					
SD10																																																					
SD11																																																					
SD12																																																					
SD13																																																					
SD14																																																					
SD15																																																					


TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																																							
SD5	Error common information	Error common information	<p>3) Time (value set)</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Time : 1μs units (0 to 999μs)</td> </tr> <tr> <td>SD6</td> <td>Time : 1ms units (0 to 65535ms)</td> </tr> <tr> <td>SD7</td> <td rowspan="8">(Empty)</td> </tr> <tr> <td>SD8</td> </tr> <tr> <td>SD9</td> </tr> <tr> <td>SD10</td> </tr> <tr> <td>SD11</td> </tr> <tr> <td>SD12</td> </tr> <tr> <td>SD13</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table> <p>4) Program error location</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td rowspan="4">File name (ASCII code: 8 characters)</td> </tr> <tr> <td>SD6</td> </tr> <tr> <td>SD7</td> </tr> <tr> <td>SD8</td> </tr> <tr> <td>SD9</td> <td>Extension *6 2EH(.)</td> </tr> <tr> <td>SD10</td> <td>(ASCII code: 3 characters)</td> </tr> <tr> <td>SD11</td> <td>Pattern *7</td> </tr> <tr> <td>SD12</td> <td>Block No.</td> </tr> <tr> <td>SD13</td> <td>Step No./transition condition</td> </tr> <tr> <td>SD14</td> <td>Sequence step No. (L)</td> </tr> <tr> <td>SD15</td> <td>Sequence step No. (H)</td> </tr> </tbody> </table> <p>*7 : Contents of pattern data</p> <table border="1"> <tr> <td>15</td> <td>14</td> <td>to</td> <td>4</td> <td>3</td> <td>2</td> <td>1</td> <td>0</td> <td>←(Bit number)</td> </tr> <tr> <td>0</td> <td>0</td> <td>to</td> <td>0</td> <td>0</td> <td>*</td> <td>*</td> <td>*</td> <td></td> </tr> </table> <p>(Not used)</p> <ul style="list-style-type: none"> <li>— SFC block designation present (1)/absent (0)</li> <li>— SFC step designation present (1)/absent (0)</li> <li>— SFC transition designation present (1)/absent (0)</li> </ul>	Number	Meaning	SD5	Time : 1μs units (0 to 999μs)	SD6	Time : 1ms units (0 to 65535ms)	SD7	(Empty)	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	Number	Meaning	SD5	File name (ASCII code: 8 characters)	SD6	SD7	SD8	SD9	Extension *6 2EH(.)	SD10	(ASCII code: 3 characters)	SD11	Pattern *7	SD12	Block No.	SD13	Step No./transition condition	SD14	Sequence step No. (L)	SD15	Sequence step No. (H)	15	14	to	4	3	2	1	0	←(Bit number)	0	0	to	0	0	*	*	*		S (Error)	New	QCPU
Number				Meaning																																																									
SD5				Time : 1μs units (0 to 999μs)																																																									
SD6				Time : 1ms units (0 to 65535ms)																																																									
SD7				(Empty)																																																									
SD8																																																													
SD9																																																													
SD10																																																													
SD11																																																													
SD12																																																													
SD13																																																													
SD14																																																													
SD15																																																													
Number				Meaning																																																									
SD5				File name (ASCII code: 8 characters)																																																									
SD6																																																													
SD7																																																													
SD8																																																													
SD9	Extension *6 2EH(.)																																																												
SD10	(ASCII code: 3 characters)																																																												
SD11	Pattern *7																																																												
SD12	Block No.																																																												
SD13	Step No./transition condition																																																												
SD14	Sequence step No. (L)																																																												
SD15	Sequence step No. (H)																																																												
15	14	to	4	3	2	1	0	←(Bit number)																																																					
0	0	to	0	0	*	*	*																																																						
SD6																																																													
SD7																																																													
SD8																																																													
SD9																																																													
SD10																																																													
SD11																																																													
SD12																																																													
SD13																																																													
SD14																																																													
SD15																																																													

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																																																																																																							
SD5	Error common information	Error common information	5) Reason(s) for system switching <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>System switching condition *13</td> </tr> <tr> <td>SD6</td> <td>Control system switching instruction argument</td> </tr> <tr> <td>SD7</td> <td rowspan="9">(Empty)</td> </tr> <tr> <td>SD8</td> </tr> <tr> <td>SD9</td> </tr> <tr> <td>SD10</td> </tr> <tr> <td>SD11</td> </tr> <tr> <td>SD12</td> </tr> <tr> <td>SD13</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table>	Number	Meaning	SD5	System switching condition *13	SD6	Control system switching instruction argument	SD7	(Empty)	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	S (Error)	New	QnPRH																																																																																																							
Number			Meaning																																																																																																																										
SD5			System switching condition *13																																																																																																																										
SD6			Control system switching instruction argument																																																																																																																										
SD7			(Empty)																																																																																																																										
SD8																																																																																																																													
SD9																																																																																																																													
SD10																																																																																																																													
SD11																																																																																																																													
SD12																																																																																																																													
SD13																																																																																																																													
SD14																																																																																																																													
SD15																																																																																																																													
SD6																																																																																																																													
SD7																																																																																																																													
SD8																																																																																																																													
SD9																																																																																																																													
SD10																																																																																																																													
SD11																																																																																																																													
SD12																																																																																																																													
SD13																																																																																																																													
SD14																																																																																																																													
SD15																																																																																																																													
SD10			*13: Details of reason(s) for system switching  <ul style="list-style-type: none"> <li>0 : No system switching condition (default)</li> <li>1 : Power-OFF, reset, hardware failure, watchdog timer error</li> <li>2 : Stop error (except watchdog timer error)</li> <li>3 : System switching request by network module</li> <li>16 : Control system switching instruction</li> <li>17 : Control system switching request from GX Developer</li> </ul>																																																																																																																										
SD10			6) Reason(s) for tracking capacity excess error The block No. when the data amount that can be tracked (100k) is exceeded is indicated by the bit pattern of the corresponding special relay.																																																																																																																										
SD11			<table border="1"> <thead> <tr> <th></th> <th>b15</th> <th>b14</th> <th>b13</th> <th>b12</th> <th>b11</th> <th>b10</th> <th>b9</th> <th>b8</th> <th>b7</th> <th>b6</th> <th>b5</th> <th>b4</th> <th>b3</th> <th>b2</th> <th>b1</th> <th>b0</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>1 (SM1535) (Block16)</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1 (SM1528) (Block9)</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1 (SM1520) (Block1)</td> </tr> <tr> <td>SD6</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>SD7</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>SD8</td> <td>1 (SM1583) (Block64)</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1 (SM1568) (Block49)</td> </tr> <tr> <td>SD9</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>SD15</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> </tbody> </table>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD5	1 (SM1535) (Block16)	0	0	0	0	0	0	1 (SM1528) (Block9)	0	0	0	0	0	0	0	1 (SM1520) (Block1)	SD6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SD7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SD8	1 (SM1583) (Block64)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1 (SM1568) (Block49)	SD9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SD15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																																													
SD5	1 (SM1535) (Block16)	0	0	0	0	0	0	1 (SM1528) (Block9)	0	0	0	0	0	0	0	1 (SM1520) (Block1)																																																																																																													
SD6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																													
SD7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																													
SD8	1 (SM1583) (Block64)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1 (SM1568) (Block49)																																																																																																													
SD9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																													
SD15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																													
SD13			7) Power supply No. <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Base No.</td> </tr> <tr> <td>SD6</td> <td>Power supply No.</td> </tr> <tr> <td>SD7</td> <td rowspan="9">(Empty)</td> </tr> <tr> <td>SD8</td> </tr> <tr> <td>SD9</td> </tr> <tr> <td>SD10</td> </tr> <tr> <td>SD11</td> </tr> <tr> <td>SD12</td> </tr> <tr> <td>SD13</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table>	Number	Meaning	SD5	Base No.	SD6	Power supply No.	SD7	(Empty)	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	S (Error)	New	Qn(H) <sup>*1</sup> QnPH <sup>*1</sup> QnPRH QnU <sup>*2</sup>																																																																																																							
Number	Meaning																																																																																																																												
SD5	Base No.																																																																																																																												
SD6	Power supply No.																																																																																																																												
SD7	(Empty)																																																																																																																												
SD8																																																																																																																													
SD9																																																																																																																													
SD10																																																																																																																													
SD11																																																																																																																													
SD12																																																																																																																													
SD13																																																																																																																													
SD14																																																																																																																													
SD15																																																																																																																													
SD14																																																																																																																													
SD15																																																																																																																													
SD15			1: Power supply 1 fault 2: Power supply 2 fault "Power supply mounted on POWER 1 slot of redundant module 1": Redundant power supply module mounted on POWER 1 slot of redundant base unit (Q38RB, Q68RB, Q65WRB) "Power supply mounted on POWER 2 slot of redundant module 2": Redundant power supply module mounted on POWER 2 slot of redundant base unit (Q38RB, Q68RB, Q65WRB)																																																																																																																										

\*1: The module whose first 5 digits of serial No. is "07032" or later.

\*2: The module whose first 5 digits of serial No. is "10042" or later.

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU																																	
SD5																																							
SD6																																							
SD7																																							
SD8			<p>8) Tracking transmission data classification Stores the data classification during tracking.</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD5</td> <td>Data type *15</td> </tr> <tr> <td>SD6</td> <td rowspan="10">(Empty)</td> </tr> <tr> <td>SD7</td> </tr> <tr> <td>SD8</td> </tr> <tr> <td>SD9</td> </tr> <tr> <td>SD10</td> </tr> <tr> <td>SD11</td> </tr> <tr> <td>SD12</td> </tr> <tr> <td>SD13</td> </tr> <tr> <td>SD14</td> </tr> <tr> <td>SD15</td> </tr> </tbody> </table> <p>*15: Details of data classification</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>b15</td> <td rowspan="15">System data</td> </tr> <tr> <td>b14 to b6</td> <td rowspan="7">Operation mode change request</td> </tr> <tr> <td>b5</td> <td rowspan="2">System switching request</td> </tr> <tr> <td>b4</td> <td rowspan="2">SFC execution data</td> </tr> <tr> <td>b3</td> <td rowspan="2">PIDINIT/S. PIDINIT instruction data</td> </tr> <tr> <td>b2</td> <td rowspan="2">Signal flow</td> </tr> <tr> <td>b1</td> <td rowspan="2">Device data</td> </tr> <tr> <td>b0</td> <td>Each bit 0: Not sent 1: Being sent</td> </tr> </tbody> </table>	Number	Meaning	SD5	Data type *15	SD6	(Empty)	SD7	SD8	SD9	SD10	SD11	SD12	SD13	SD14	SD15	Bit	Meaning	b15	System data	b14 to b6	Operation mode change request	b5	System switching request	b4	SFC execution data	b3	PIDINIT/S. PIDINIT instruction data	b2	Signal flow	b1	Device data	b0	Each bit 0: Not sent 1: Being sent			
Number	Meaning																																						
SD5	Data type *15																																						
SD6	(Empty)																																						
SD7																																							
SD8																																							
SD9																																							
SD10																																							
SD11																																							
SD12																																							
SD13																																							
SD14																																							
SD15																																							
Bit	Meaning																																						
b15	System data																																						
b14 to b6		Operation mode change request																																					
b5			System switching request																																				
b4				SFC execution data																																			
b3			PIDINIT/S. PIDINIT instruction data																																				
b2				Signal flow																																			
b1			Device data																																				
b0				Each bit 0: Not sent 1: Being sent																																			
SD9																																							
SD10		Error common information	Error common information		S (Error)	New	QnPRH																																
SD11																																							
SD12																																							
SD13																																							
SD14																																							
SD15																																							

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																				
SD16	Error individual information	Error individual information	<ul style="list-style-type: none"> <li>Individual information corresponding to error codes (SD0) is stored here.</li> <li>There are the following eight different types of information are stored.</li> <li>The error individual information type can be judged by the "individual information category code" in SD4. (The values of the "individual information category code" stored in SD4 correspond to following 1) to 8), 12), and 13).)</li> </ul>	S (Error)	New	QCPU																																				
SD17			1) (Empty) 2) File name/Drive name (Example) File name = ABCDEFGH, IJK b15 to b8 b7 to b0																																							
SD18			<table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td>Drive</td> </tr> <tr> <td>SD17</td> <td rowspan="2">File name</td> </tr> <tr> <td>SD18</td> <td>(ASCII code: 8 characters)</td> </tr> <tr> <td>SD19</td> <td rowspan="2">Extension *6</td> </tr> <tr> <td>SD20</td> <td>2EH(.)</td> </tr> <tr> <td>SD21</td> <td rowspan="2">(ASCII code: 3 characters)</td> </tr> <tr> <td>SD22</td> <td>42H(B) 41H(A)</td> </tr> <tr> <td>SD23</td> <td rowspan="2">(Empty)</td> </tr> <tr> <td>SD24</td> <td>44H(D) 43H(C)</td> </tr> <tr> <td>SD25</td> <td rowspan="2">(Empty)</td> </tr> <tr> <td>SD26</td> <td>46H(F) 45H(E)</td> </tr> <tr> <td>SD26</td> <td rowspan="2">(Empty)</td> </tr> <tr> <td>SD26</td> <td>48H(H) 47H(G)</td> </tr> <tr> <td>SD26</td> <td rowspan="2">(Empty)</td> </tr> <tr> <td>SD26</td> <td>49H(I) 2EH(.)</td> </tr> <tr> <td>SD26</td> <td rowspan="2">(Empty)</td> </tr> <tr> <td>SD26</td> <td>4BH(K) 4AH(J)</td> </tr> </tbody> </table>				Number	Meaning	SD16	Drive	SD17	File name	SD18	(ASCII code: 8 characters)	SD19	Extension *6	SD20	2EH(.)	SD21	(ASCII code: 3 characters)	SD22	42H(B) 41H(A)	SD23	(Empty)	SD24	44H(D) 43H(C)	SD25	(Empty)	SD26	46H(F) 45H(E)	SD26	(Empty)	SD26	48H(H) 47H(G)	SD26	(Empty)	SD26	49H(I) 2EH(.)	SD26	(Empty)	SD26	4BH(K) 4AH(J)
Number			Meaning																																							
SD16			Drive																																							
SD17			File name																																							
SD18							(ASCII code: 8 characters)																																			
SD19			Extension *6																																							
SD20							2EH(.)																																			
SD21			(ASCII code: 3 characters)																																							
SD22							42H(B) 41H(A)																																			
SD23			(Empty)																																							
SD24							44H(D) 43H(C)																																			
SD25			(Empty)																																							
SD26							46H(F) 45H(E)																																			
SD26			(Empty)																																							
SD26							48H(H) 47H(G)																																			
SD26			(Empty)																																							
SD26							49H(I) 2EH(.)																																			
SD26			(Empty)																																							
SD26							4BH(K) 4AH(J)																																			
SD19			3) Time (value actually measured)																																							
SD20			<table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td>Time : 1μs units (0 to 999μs)</td> </tr> <tr> <td>SD17</td> <td>Time : 1ms units (0 to 65535ms)</td> </tr> <tr> <td>SD18</td> <td rowspan="8">(Empty)</td> </tr> <tr> <td>SD19</td> </tr> <tr> <td>SD20</td> </tr> <tr> <td>SD21</td> </tr> <tr> <td>SD22</td> </tr> <tr> <td>SD23</td> </tr> <tr> <td>SD24</td> </tr> <tr> <td>SD25</td> </tr> <tr> <td>SD26</td> </tr> </tbody> </table>				Number	Meaning	SD16	Time : 1μs units (0 to 999μs)	SD17	Time : 1ms units (0 to 65535ms)	SD18	(Empty)	SD19	SD20	SD21	SD22	SD23	SD24	SD25	SD26																				
Number			Meaning																																							
SD16			Time : 1μs units (0 to 999μs)																																							
SD17			Time : 1ms units (0 to 65535ms)																																							
SD18	(Empty)																																									
SD19																																										
SD20																																										
SD21																																										
SD22																																										
SD23																																										
SD24																																										
SD25																																										
SD26																																										
SD21	4) Program error location																																									
SD22	<table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td rowspan="4">File name</td> </tr> <tr> <td>SD17</td> <td>(ASCII code: 8 characters)</td> </tr> <tr> <td>SD18</td> <td rowspan="2">Extension *6</td> </tr> <tr> <td>SD19</td> <td>2EH(.)</td> </tr> <tr> <td>SD20</td> <td rowspan="2">(ASCII code: 3 characters)</td> </tr> <tr> <td>SD21</td> <td>Pattern *7</td> </tr> <tr> <td>SD22</td> <td>Block No.</td> </tr> <tr> <td>SD23</td> <td>Step No./transition No.</td> </tr> <tr> <td>SD24</td> <td>Sequence step No. (L)</td> </tr> <tr> <td>SD25</td> <td>Sequence step No. (H)</td> </tr> <tr> <td>SD26</td> <td></td> </tr> </tbody> </table>	Number	Meaning	SD16	File name	SD17	(ASCII code: 8 characters)	SD18	Extension *6	SD19	2EH(.)	SD20	(ASCII code: 3 characters)	SD21	Pattern *7	SD22	Block No.	SD23	Step No./transition No.	SD24	Sequence step No. (L)	SD25	Sequence step No. (H)	SD26																		
Number	Meaning																																									
SD16	File name																																									
SD17		(ASCII code: 8 characters)																																								
SD18		Extension *6																																								
SD19			2EH(.)																																							
SD20	(ASCII code: 3 characters)																																									
SD21		Pattern *7																																								
SD22	Block No.																																									
SD23	Step No./transition No.																																									
SD24	Sequence step No. (L)																																									
SD25	Sequence step No. (H)																																									
SD26																																										
SD23	*7 : Contents of pattern data <table border="1"> <tr> <td>15</td><td>14</td><td>to</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td>←(Bit number)</td> </tr> <tr> <td>0</td><td>0</td><td>to</td><td>0</td><td>0</td><td>*</td><td>*</td><td>*</td><td></td> </tr> </table>	15	14	to	4	3	2	1	0	←(Bit number)	0	0	to	0	0	*	*	*																								
15	14	to	4	3	2	1	0	←(Bit number)																																		
0	0	to	0	0	*	*	*																																			
SD24	<table border="1"> <tr> <td>(Not used)</td> <td> <ul style="list-style-type: none"> <li>SFC block designation present (1)/absent (0)</li> <li>SFC step designation present (1)/absent (0)</li> <li>SFC transition designation present (1)/absent (0)</li> </ul> </td> </tr> </table>	(Not used)	<ul style="list-style-type: none"> <li>SFC block designation present (1)/absent (0)</li> <li>SFC step designation present (1)/absent (0)</li> <li>SFC transition designation present (1)/absent (0)</li> </ul>																																							
(Not used)	<ul style="list-style-type: none"> <li>SFC block designation present (1)/absent (0)</li> <li>SFC step designation present (1)/absent (0)</li> <li>SFC transition designation present (1)/absent (0)</li> </ul>																																									
SD25	5) Parameter No. 6) Annunciator number / 7) CHK instruction malfunction number																																									
SD26	<table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td>Parameter No. *16</td> <td>SD16</td> <td>No.</td> </tr> <tr> <td>SD17</td> <td rowspan="8">(Empty)</td> <td>SD17</td> <td rowspan="8">(Empty)</td> </tr> <tr> <td>SD18</td> </tr> <tr> <td>SD19</td> </tr> <tr> <td>SD20</td> </tr> <tr> <td>SD21</td> </tr> <tr> <td>SD22</td> </tr> <tr> <td>SD23</td> </tr> <tr> <td>SD24</td> </tr> <tr> <td>SD25</td> </tr> <tr> <td>SD26</td> </tr> </tbody> </table>	Number	Meaning	Number	Meaning	SD16	Parameter No. *16	SD16	No.	SD17	(Empty)	SD17	(Empty)	SD18	SD19	SD20	SD21	SD22	SD23	SD24	SD25	SD26																				
Number	Meaning	Number	Meaning																																							
SD16	Parameter No. *16	SD16	No.																																							
SD17	(Empty)	SD17	(Empty)																																							
SD18																																										
SD19																																										
SD20																																										
SD21																																										
SD22																																										
SD23																																										
SD24																																										
SD25																																										
SD26																																										
SD26	*16: For details of the parameter No., refer to the User's Manual (Function Explanation, Program Fundamentals) of the CPU module used.																																									

\*6 : Extensions are shown below.

TableApp.4.3 Extension name

SDn Higher 8 bits	SDn+1		Extension Name	File Type
	Lower 8 bits	Higher 8 bits		
51H	50H	41H	QPA	Parameters
51H	50H	47H	QPG	<ul style="list-style-type: none"> <li>• Sequence program</li> <li>• SFC program</li> </ul>
51H	43H	44H	QCD	Device comment
51H	44H	49H	QDI	Initial device value
51H	44H	52H	QDR	File register
51H	44H	4CH	QDL	Local device (Other than the Basic model QCPU)
51H	54H	44H	QTD	Sampling trace data (Other than the Basic model QCPU)
51H	46H	44H	QFD	Breakdown history data (Other than the Basic model QCPU and the Universal model QCPU)
51H	53H	54H	QST	SP.DEVST/S.DEVLD instruction file (For Universal model QCPU only)

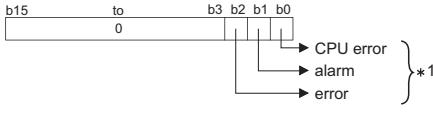
A

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																							
SD26	Error individual information	Error individual information	<p>8) Reason(s) for system switching failure</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td>System switching prohibition condition *14</td> </tr> <tr> <td>SD17</td> <td rowspan="10">(Empty)</td> </tr> <tr> <td>SD18</td> </tr> <tr> <td>SD19</td> </tr> <tr> <td>SD20</td> </tr> <tr> <td>SD21</td> </tr> <tr> <td>SD22</td> </tr> <tr> <td>SD23</td> </tr> <tr> <td>SD24</td> </tr> <tr> <td>SD25</td> </tr> <tr> <td>SD26</td> </tr> </tbody> </table> <p>*14: Details of reason(s) for system switching failure</p> <table border="1"> <tr> <td style="width: 100px; height: 20px;"></td> </tr> </table> <ul style="list-style-type: none"> <li>0 : Normal switching completion (default)</li> <li>1 : Tracking cable fault (cable removal, cable fault, internal circuit fault, hardware fault)</li> <li>2 : Hardware failure, power OFF, reset or watchdog timer error occurring in standby system</li> <li>3 : Hardware failure, power OFF, reset or watchdog timer error occurring in control system</li> <li>4 : Preparing for tracking</li> <li>5 : Time limit exceeded</li> <li>6 : Standby system is in stop error (except watchdog timer error)</li> <li>7 : Operation differs between two systems (in backup mode only)</li> <li>8 : During memory copy from control system to standby system</li> <li>9 : Online program change</li> <li>10 : Error detected by network module of standby system</li> <li>11 : System switching being executed</li> <li>12 : Online module change in progress</li> </ul>	Number	Meaning	SD16	System switching prohibition condition *14	SD17	(Empty)	SD18	SD19	SD20	SD21	SD22	SD23	SD24	SD25	SD26		S (Error)	New	QnPRH																							
			Number	Meaning																																									
SD16	System switching prohibition condition *14																																												
SD17	(Empty)																																												
SD18																																													
SD19																																													
SD20																																													
SD21																																													
SD22																																													
SD23																																													
SD24																																													
SD25																																													
SD26																																													
<p>12) File diagnostic information</p> <table border="1"> <tr> <td>SD16</td> <td>Failuer information (H)</td> <td>drive No.(L)</td> </tr> <tr> <td>SD17</td> <td colspan="2" rowspan="4">File name (ASCII: 8 characters)</td> </tr> <tr> <td>SD18</td> </tr> <tr> <td>SD19</td> </tr> <tr> <td>SD20</td> </tr> <tr> <td>SD21</td> <td>EXtension *6</td> <td>2EH(.)</td> </tr> <tr> <td>SD22</td> <td colspan="2">(ASCII: 3 characters)</td> </tr> <tr> <td>SD23</td> <td colspan="2">Failure information 2</td> </tr> <tr> <td>SD24</td> <td colspan="2">(CRC value that is read)</td> </tr> <tr> <td>SD25</td> <td colspan="2">Failure information 3</td> </tr> <tr> <td>SD26</td> <td colspan="2">(CRC value that is calculated)</td> </tr> </table> <p>13) Parameter No./CPU No.</p> <table border="1"> <thead> <tr> <th>Number</th> <th>Meaning</th> </tr> </thead> <tbody> <tr> <td>SD16</td> <td>Parameter No.*16</td> </tr> <tr> <td>SD17</td> <td>CPU No. (1 to 4)</td> </tr> <tr> <td>SD18</td> <td rowspan="10">(Empty)</td> </tr> <tr> <td>SD19</td> </tr> <tr> <td>SD20</td> </tr> <tr> <td>SD21</td> </tr> <tr> <td>SD22</td> </tr> <tr> <td>SD23</td> </tr> <tr> <td>SD24</td> </tr> <tr> <td>SD25</td> </tr> <tr> <td>SD26</td> </tr> </tbody> </table>	SD16	Failuer information (H)	drive No.(L)	SD17	File name (ASCII: 8 characters)		SD18	SD19	SD20	SD21	EXtension *6	2EH(.)	SD22	(ASCII: 3 characters)		SD23	Failure information 2		SD24	(CRC value that is read)		SD25	Failure information 3		SD26	(CRC value that is calculated)		Number	Meaning	SD16	Parameter No.*16	SD17	CPU No. (1 to 4)	SD18	(Empty)	SD19	SD20	SD21	SD22	SD23	SD24	SD25	SD26		QnU
SD16	Failuer information (H)	drive No.(L)																																											
SD17	File name (ASCII: 8 characters)																																												
SD18																																													
SD19																																													
SD20																																													
SD21	EXtension *6	2EH(.)																																											
SD22	(ASCII: 3 characters)																																												
SD23	Failure information 2																																												
SD24	(CRC value that is read)																																												
SD25	Failure information 3																																												
SD26	(CRC value that is calculated)																																												
Number	Meaning																																												
SD16	Parameter No.*16																																												
SD17	CPU No. (1 to 4)																																												
SD18	(Empty)																																												
SD19																																													
SD20																																													
SD21																																													
SD22																																													
SD23																																													
SD24																																													
SD25																																													
SD26																																													



TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD50	Error reset	Error number that performs error reset	<ul style="list-style-type: none"> <li>Stores error number that performs error reset</li> </ul>	U	New	
SD51	Battery low latch	Bit pattern indicating where battery voltage drop occurred	<ul style="list-style-type: none"> <li>All corresponding bits go 1(ON) when battery voltage drops.</li> <li>Subsequently, these remain 1(ON) even after battery voltage has been returned to normal.</li> </ul>  <p>* 1: This does not apply to Basic model QCPU.</p> <ul style="list-style-type: none"> <li>In the alarm, data can be held within the time specified for battery low.</li> <li>The error indicates the complete discharge of the battery.</li> </ul>	S (Error)	New	QCPU
SD52	Battery low	Bit pattern indicating where battery voltage drop occurred	<ul style="list-style-type: none"> <li>Same configuration as SD51 above</li> <li>After the alarm is detected (ON), the alarm turns OFF by error detection (ON). (For the Universal model QCPU only)</li> <li>Turns to 0 (OFF) when the battery voltage returns to normal thereafter.</li> </ul>	S (Error)	New	
SD53	AC/DC DOWN detection	Number of times for AC/DC DOWN detection	<ul style="list-style-type: none"> <li>Every time the input voltage falls to or below 85% (AC power)/65% (DC power) of the rating during operation of the CPU module, the value is incremented by 1 and stored in BIN code.</li> <li>The counter repeats increment and decrement of the value ; 0 → 32767 → -32768 → 0</li> </ul>	S (Error)	D9005	
SD60	Number of module with blown fuse	Number of module with blown fuse	<ul style="list-style-type: none"> <li>Value stored here is the lowest station I/O number of the module with the blown fuse.</li> </ul>	S (Error)	D9000	
SD61	I/O module verify error number	I/O module verify error module number	<ul style="list-style-type: none"> <li>The lowest I/O number of the module where the I/O module verification number took place.</li> </ul>	S (Error)	D9002	

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU	
SD62	Annunciator number	Annunciator number	• The first annunciator number (F number) to be detected is stored here.	S (Instruction execution)	D9009	QCPU	
SD63	Number of annunciators	Number of annunciators	• Stores the number of annunciators searched.	S (Instruction execution)	D9124		
SD64	Table of detected annunciator numbers	Annunciator detection number	<p>When F goes ON due to OUT F or SET F instruction, the F numbers which go progressively ON from SD64 through SD79 are registered. The F numbers turned OFF by RST F instruction are deleted from SD64 - SD79, and the F numbers stored after the deleted F numbers are shifted to the preceding registers. Execution of the LEDR instruction shifts the contents of SD64 to SD79 up by one. After 16 annunciators have been detected, detection of the 17th will not be stored from SD64 through SD79.</p> <p style="text-align: center;">SET SET SET RST SET SET SET SET SET SET SET SET SET SET SET SET SET SET F50 F25 F99 F25 F15 F70 F65 F38 F110 F151 F210 LEDR</p> <p>SD62   0   50   50   50   50   50   50   50   50   50   50   50   50   50   99   (Number detected)</p> <p>SD63   0   1   2   3   2   3   4   5   6   7   8   9   8   (Number of annunciators detected)</p> <p>SD64   0   50   50   50   50   50   50   50   50   50   50   50   99  </p> <p>SD65   0   0   25   25   99   99   99   99   99   99   99   99   15  </p> <p>SD66   0   0   0   99   0   15   15   15   15   15   15   15   70  </p> <p>SD67   0   0   0   0   0   0   70   70   70   70   70   70   65  </p> <p>SD68   0   0   0   0   0   0   0   65   65   65   65   65   38  </p> <p>SD69   0   0   0   0   0   0   0   0   38   38   38   38   110  </p> <p>SD70   0   0   0   0   0   0   0   0   0   0   110   110   151  </p> <p>SD71   0   0   0   0   0   0   0   0   0   0   151   151   210  </p> <p>SD72   0   0   0   0   0   0   0   0   0   0   210   0  </p> <p>SD73   0   0   0   0   0   0   0   0   0   0   0   0   0  </p> <p>SD74   0   0   0   0   0   0   0   0   0   0   0   0   0  </p> <p>SD75   0   0   0   0   0   0   0   0   0   0   0   0   0  </p> <p>SD76   0   0   0   0   0   0   0   0   0   0   0   0   0  </p> <p>SD77   0   0   0   0   0   0   0   0   0   0   0   0   0  </p> <p>SD78   0   0   0   0   0   0   0   0   0   0   0   0   0  </p> <p>SD79   0   0   0   0   0   0   0   0   0   0   0   0   0  </p>	S (Instruction execution)	D9125		
SD65					D9126		
SD66					D9127		
SD67					D9128		
SD68					D9129		
SD69					D9130		
SD70					D9131		
SD71					D9132		
SD72					New		
SD73					New		
SD74					New		
SD75					New		
SD76					New		
SD77					New		
SD78	New						
SD79	New						
SD80	CHK number	CHK number	• Error codes detected by the CHK instruction are stored as BCD code.	S (Instruction execution)	New		Qn(H) QnPH QnPRH
SD90	Step transition monitoring timer setting value (Enabled only when SFC program exists)	F number for timer set value and time over error	Corresponds to SM90	U	D9108		
SD91			Corresponds to SM91		D9109		
SD92			Corresponds to SM92		D9110		
SD93			Corresponds to SM93		D9111		
SD94			Corresponds to SM94		D9112		
SD95			Corresponds to SM95		D9113		
SD96			Corresponds to SM96		D9114		
SD97			Corresponds to SM97		New		
SD98			Corresponds to SM98		New		
SD99			Corresponds to SM99		New		

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU	Corresponding CPU
					D9□□□	
SD100	Transmission speed storage area	Stores the transmission speed specified in the serial communication setting.	96 : 9.6kbps, 192 : 19.2kbps, 384 : 38.4kbps, 576 : 57.6kbps, 1152 : 115.2kbps	S (Power-ON or reset)	New	
SD101	Communication setting storage area	Stores the communication setting specified in the serial communication setting.	<p>Write during RUN setting 0: Disabled 1: Enabled</p> <p>Sumcheck presence 0: Absent 1: Present</p> <p>* : Since the data is used by the system, it is undefined.</p>	S (Power-ON or reset)	New	Q00/Q01 Q00UJ Q00U Q01U Q02U <sup>*4</sup>
SD102	Transmission wait time storage area	Stores the transmission wait time specified in the serial communication setting.	0 : No waiting time 10 to 150: Waiting time (unit: ms) Defaults to 0.	S (Power-ON or reset)	New	
SD105	CH1 transmission speed setting (RS-232)	Stores the preset transmission speed when GX Developer is used.	96 : 9600bps, 192 : 19.2kbps, 384 : 38.4kbps, 576 : 57.6kbps, 1152 : 115.2kbps *: Other than RS-232 connection holds the data at RS-232 connection. (When disconnected, the default value is 1152.)	S	New	Qn(H) QnPH QnPRH QnU <sup>*3</sup>
SD110	Data sending result storage area	Stores the data sending result when the serial communication function is used.	Stores the error code at the timeout sending data.	S (Error)	New	Q00/Q01 Q00UJ Q00U Q01U Q02U <sup>*4</sup>
SD111	Data receiving result storage area	Stores the data receiving result when the serial communication function is used.	Stores the error code at the time of receiving data.	S (Error)		
SD118	Amount of battery consumption	Amount of battery consumption	Displays the current amount of battery consumption. The value range: 1 to 2(Q00UJCPU, Q00UCPU, Q01UCPU, Q02UCPU, Q03UD(E)CPU, Q04UD(E)HCPU) 1 to 3(Q06UD(E)HCPU) 1 to 4(Q10UD(E)HCPU, Q20UD(E)HCPU, Q13UD(E)HCPU, Q26UD(E)HCPU)	S (Status change)	New	QnU <sup>*4</sup>
SD119	Battery life-prolonging factor	Battery life-prolonging factor	Stores the factor which makes the battery life-prolonging function valid. When SD119 is other than 0, the battery life-prolonging function is valid.  0: No factor 1: Factor  	S (Status change)	New	QnU

\*3: This applies to Universal model QCPUs except for the Built-in Ethernet port QCPU.

\*4: The module whose first 5 digits of serial No. is "10102" or later.

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.2 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																																																				
SD130	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown 0: No blown fuse 1: Blown fuse present	<ul style="list-style-type: none"> <li>The numbers of output modules whose fuses have blown are input as a bit pattern (in units of 16 points). (If the module numbers are set by parameter, the parameter-set numbers are stored.)</li> </ul> <table border="1"> <tr> <td></td> <td>b15</td><td>b14</td><td>b13</td><td>b12</td><td>b11</td><td>b10</td><td>b9</td><td>b8</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td> </tr> <tr> <td>SD130</td> <td>0</td><td>0</td><td>0</td><td>1 (YC0)</td><td>0</td><td>0</td><td>0</td><td>1 (Y80)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td>SD131</td> <td>1 (Y1F0)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (Y1A0)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td>SD137</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>1 (Y7B0)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (Y730)</td><td>0</td><td>0</td><td>0</td> </tr> </table> <p style="text-align: center;">↑ Indicates fuse blow.</p>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD130	0	0	0	1 (YC0)	0	0	0	1 (Y80)	0	0	0	0	0	0	0	0	SD131	1 (Y1F0)	0	0	0	0	1 (Y1A0)	0	0	0	0	0	0	0	0	0	0	SD137	0	0	0	0	1 (Y7B0)	0	0	0	0	0	0	0	1 (Y730)	0	0	0	S (Error)	New	Q00J/Q00/Q01
			b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																								
SD130			0	0	0	1 (YC0)	0	0	0	1 (Y80)	0	0	0	0	0	0	0	0																																																								
SD131			1 (Y1F0)	0	0	0	0	1 (Y1A0)	0	0	0	0	0	0	0	0	0	0																																																								
SD137			0	0	0	0	1 (Y7B0)	0	0	0	0	0	0	0	1 (Y730)	0	0	0																																																								
SD131																																																																										
SD132																																																																										
SD133																																																																										
SD134																																																																										
SD135																																																																										
SD136																																																																										
SD137	<ul style="list-style-type: none"> <li>Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.</li> </ul>																																																																									
SD150	I/O module verify error	Bit pattern, in units of 16 points, indicating the modules with verify errors. 0: No I/O verify errors 1: I/O verify error present	<ul style="list-style-type: none"> <li>When I/O modules, of which data are different from those entered at power-ON, have been detected, the I/O module numbers (in units of 16 points) are entered in bit pattern. (Preset I/O module numbers set in parameters when parameter setting has been performed.)</li> </ul> <table border="1"> <tr> <td></td> <td>b15</td><td>b14</td><td>b13</td><td>b12</td><td>b11</td><td>b10</td><td>b9</td><td>b8</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td> </tr> <tr> <td>SD150</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (X<sub>Y</sub>)</td> </tr> <tr> <td>SD151</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1 (X<sub>Y</sub>)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td>SD157</td> <td>1 (X<sub>Y</sub>)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> </table> <p style="text-align: center;">↑ Indicates an I/O module verify error.</p>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1 (X <sub>Y</sub> )	SD151	0	0	0	0	0	0	1 (X <sub>Y</sub> )	0	0	0	0	0	0	0	0	0	SD157	1 (X <sub>Y</sub> )	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	S (Error)	New	Q00J/Q00/Q01
			b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																								
SD150			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1 (X <sub>Y</sub> )																																																								
SD151			0	0	0	0	0	0	1 (X <sub>Y</sub> )	0	0	0	0	0	0	0	0	0																																																								
SD157			1 (X <sub>Y</sub> )	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																								
SD151																																																																										
SD152																																																																										
SD153																																																																										
SD154																																																																										
SD155																																																																										
SD156																																																																										
SD157	<ul style="list-style-type: none"> <li>Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.</li> </ul>																																																																									

(2) System information

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU												
SD200	Status of switch	Status of CPU switch	<ul style="list-style-type: none"> <li>The CPU switch status is stored in the following format:</li> </ul> <table border="1"> <tr> <td>1): CPU switch status</td> <td>0: RUN 1: STOP 2: L.CLR</td> </tr> <tr> <td>2): Memory card switch</td> <td>Always OFF</td> </tr> <tr> <td>3): DIP switch</td> <td>b8 through b12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON. b13 through b15 are empty.</td> </tr> </table>	1): CPU switch status	0: RUN 1: STOP 2: L.CLR	2): Memory card switch	Always OFF	3): DIP switch	b8 through b12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON. b13 through b15 are empty.	S (Every END processing)	New	Qn(H) QnPH QnPRH						
			1): CPU switch status	0: RUN 1: STOP 2: L.CLR														
			2): Memory card switch	Always OFF														
3): DIP switch	b8 through b12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF, 1: ON. b13 through b15 are empty.																	
<ul style="list-style-type: none"> <li>The CPU switch status is stored in the following format:</li> </ul> <table border="1"> <tr> <td>1): CPU switch status</td> <td>0: RUN 1: STOP</td> </tr> <tr> <td>2): Memory card switch</td> <td>Always OFF</td> </tr> </table>	1): CPU switch status	0: RUN 1: STOP	2): Memory card switch	Always OFF	S (Every END processing)	New	Q00J/Q00/Q01											
1): CPU switch status	0: RUN 1: STOP																	
2): Memory card switch	Always OFF																	
<ul style="list-style-type: none"> <li>The CPU switch status is stored in the following format:</li> </ul> <table border="1"> <tr> <td>1): CPU switch status</td> <td>0: RUN 1: STOP</td> </tr> <tr> <td>2): Memory card switch</td> <td>Always OFF</td> </tr> </table>	1): CPU switch status	0: RUN 1: STOP	2): Memory card switch	Always OFF	S (when RUN/STOP/RESET switch changed)	New	QnU											
1): CPU switch status	0: RUN 1: STOP																	
2): Memory card switch	Always OFF																	
SD201	LED status	Status of CPU-LED	<ul style="list-style-type: none"> <li>The following bit patterns store the status of the LEDs on the CPU module:</li> <li>0 is off, 1 is on, and 2 is flicker.</li> </ul> <table border="1"> <tr> <td>1): RUN</td> <td>5): BOOT</td> <td></td> </tr> <tr> <td>2): ERR.</td> <td>6): Empty</td> <td>Mode bit pattern</td> </tr> <tr> <td>3): USER</td> <td>7): Empty</td> <td>0: OFF 1: Green</td> </tr> <tr> <td>4): BAT.</td> <td>8): MODE</td> <td>2: Orange</td> </tr> </table> <p>(The Basic model QCPU does not include 3) to 8).)</p>	1): RUN	5): BOOT		2): ERR.	6): Empty	Mode bit pattern	3): USER	7): Empty	0: OFF 1: Green	4): BAT.	8): MODE	2: Orange	S (Status change)	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH
			1): RUN	5): BOOT														
2): ERR.	6): Empty	Mode bit pattern																
3): USER	7): Empty	0: OFF 1: Green																
4): BAT.	8): MODE	2: Orange																
<ul style="list-style-type: none"> <li>The following bit patterns store the status of the LEDs on the CPU module:</li> <li>0 is off, 1 is on, and 2 is flicker.</li> </ul> <table border="1"> <tr> <td>1): RUN</td> <td>5): BOOT</td> <td></td> </tr> <tr> <td>2): ERROR</td> <td>6): Empty</td> <td></td> </tr> <tr> <td>3): USER</td> <td>7): Empty</td> <td></td> </tr> <tr> <td>4): BAT.</td> <td>8): MODE</td> <td></td> </tr> </table> <p>(The Q00JCPU, Q00UCPU, and Q01UCPU do not include 5).)</p>	1): RUN	5): BOOT		2): ERROR	6): Empty		3): USER	7): Empty		4): BAT.	8): MODE		S (Status change)	New	QnU			
1): RUN	5): BOOT																	
2): ERROR	6): Empty																	
3): USER	7): Empty																	
4): BAT.	8): MODE																	

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																		
SD202	LED off command	Bit pattern of LED that is turned off	<ul style="list-style-type: none"> <li>Specify the LEDs to be turned off using this register, and turn SM202 from OFF to ON to turn off the specified LEDs. USER and BOOT can be specified as the LEDs to be turned off.</li> <li>Specify the LEDs to be turned off in the following bit pattern. (Turned off at 1, not be turned off at 0.)</li> </ul> <p>(The Q00UJCPU, Q00UCPU, and Q01UCPU cannot specify the BOOT LED.)</p>	U	New	Qn(H) QnPH QnPRH QnU																		
SD203	Operating status of CPU	Operating status of CPU	<ul style="list-style-type: none"> <li>The CPU operating status is stored as indicated in the following figure:</li> </ul> <table border="1"> <tr> <td>1): Operating status of CPU</td> <td>0: RUN</td> </tr> <tr> <td></td> <td>1: STEP-RUN (For the QnACPU only)</td> </tr> <tr> <td></td> <td>2: STOP</td> </tr> <tr> <td></td> <td>3: PAUSE</td> </tr> <tr> <td>2): STOP/PAUSE cause</td> <td>0: Instruction in remote operation program from RUN/STOP/RESET switch ("RUN/STOP/RESET switch" for Basic model QCPU)</td> </tr> <tr> <td></td> <td>1: Remote contact</td> </tr> <tr> <td></td> <td>2: Remote operation from GX Developer/serial communication, etc.</td> </tr> <tr> <td></td> <td>3: Internal program instruction</td> </tr> <tr> <td>Note: Priority is earliest first</td> <td>4: Error</td> </tr> </table>	1): Operating status of CPU	0: RUN		1: STEP-RUN (For the QnACPU only)		2: STOP		3: PAUSE	2): STOP/PAUSE cause	0: Instruction in remote operation program from RUN/STOP/RESET switch ("RUN/STOP/RESET switch" for Basic model QCPU)		1: Remote contact		2: Remote operation from GX Developer/serial communication, etc.		3: Internal program instruction	Note: Priority is earliest first	4: Error	S (Every END processing)	D9015 format change	QCPU
1): Operating status of CPU	0: RUN																							
	1: STEP-RUN (For the QnACPU only)																							
	2: STOP																							
	3: PAUSE																							
2): STOP/PAUSE cause	0: Instruction in remote operation program from RUN/STOP/RESET switch ("RUN/STOP/RESET switch" for Basic model QCPU)																							
	1: Remote contact																							
	2: Remote operation from GX Developer/serial communication, etc.																							
	3: Internal program instruction																							
Note: Priority is earliest first	4: Error																							
SD204	LED display color	CPU-LED display color	<ul style="list-style-type: none"> <li>The LED display color of the LED status shown in SD201 1) to 8).</li> </ul> <ul style="list-style-type: none"> <li>1)RUN LED 0: OFF 1: Green</li> <li>2)ERROR LED 0: OFF 1: Red</li> <li>3)USER LED 0: OFF 1: Red</li> <li>4)BAT. LED 0: OFF 1: Yellow 2: Green</li> <li>5)BOOT LED 0: OFF 1: Green</li> <li>6)Empty</li> <li>7)Empty</li> <li>8)MODE LED 0: OFF 1: Green</li> </ul> <p>(The Q00UJCPU, Q00UCPU, and Q01UCPU do not include 5).)</p>	S (status change)	New	QnU																		

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU	Corresponding CPU																
					D9□□□																	
SD207	LED display priority ranking	Priorities 1 to 4	<ul style="list-style-type: none"> <li>When error is generated, the LED display (flicker) is made according to the error number setting priorities. (The Basic model QCPU supports only the annunciator (error item No. 7)).</li> <li>The Universal model QCPU sets execution/non-execution of LED display of the error corresponding to the each priority ranking when the error occurs.</li> <li>The setting areas for priorities are as follows:  <table border="1" style="margin-left: 40px;"> <tr> <td style="text-align: center;">b15 to b12</td> <td style="text-align: center;">b11 to b8</td> <td style="text-align: center;">b7 to b4</td> <td style="text-align: center;">b3 to b0</td> </tr> <tr> <td style="text-align: center;">Priority 4</td> <td style="text-align: center;">Priority 3</td> <td style="text-align: center;">Priority 2</td> <td style="text-align: center;">Priority 1</td> </tr> <tr> <td style="text-align: center;">Priority 8</td> <td style="text-align: center;">Priority 7</td> <td style="text-align: center;">Priority 6</td> <td style="text-align: center;">Priority 5</td> </tr> <tr> <td style="text-align: center;">Priority 11</td> <td style="text-align: center;">Priority 10</td> <td style="text-align: center;">Priority 9</td> <td></td> </tr> </table>                     (Priority 11 is valid when Redundant CPU is used.)                 </li> </ul> Default Value SD207 = 4321 <sub>H</sub> (0000 <sub>H</sub> for Basic model QCPU) SD208 = 8765 <sub>H</sub> (0700 <sub>H</sub> for Basic model QCPU) (0765 <sub>H</sub> for Redundant CPU) SD209 = 00A9 <sub>H</sub> (0000 <sub>H</sub> for Basic model QCPU) (0B09 <sub>H</sub> for Redundant CPU)	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Priority 4	Priority 3	Priority 2	Priority 1	Priority 8	Priority 7	Priority 6	Priority 5	Priority 11	Priority 10	Priority 9		U	D9038	Q00J/ Q00/Q01 <sup>*9</sup> Qn(H) QnPH QnPRH QnU
b15 to b12		b11 to b8		b7 to b4	b3 to b0																	
Priority 4		Priority 3		Priority 2	Priority 1																	
Priority 8	Priority 7	Priority 6	Priority 5																			
Priority 11	Priority 10	Priority 9																				
SD208	Priorities 5 to 8	D9039 format change																				
SD209	Priorities 9 to 11	New																				
SD210	Clock data	Clock data (year, month)	<ul style="list-style-type: none"> <li>The year (last two digits) and month are stored as BCD code as shown below:  <table border="1" style="margin-left: 40px;"> <tr> <td style="text-align: center;">b15 to b12</td> <td style="text-align: center;">b11 to b8</td> <td style="text-align: center;">b7 to b4</td> <td style="text-align: center;">b3 to b0</td> </tr> <tr> <td style="text-align: center;">Year</td> <td style="text-align: center;">Month</td> <td></td> <td></td> </tr> </table>                     Example:                      July, 1993                      9307<sub>H</sub> </li> </ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Year	Month			S (Request)/U	D9025	QCPU								
b15 to b12	b11 to b8	b7 to b4	b3 to b0																			
Year	Month																					
SD211	Clock data	Clock data (day, hour)	<ul style="list-style-type: none"> <li>The day and hour are stored as BCD code as shown below:  <table border="1" style="margin-left: 40px;"> <tr> <td style="text-align: center;">b15 to b12</td> <td style="text-align: center;">b11 to b8</td> <td style="text-align: center;">b7 to b4</td> <td style="text-align: center;">b3 to b0</td> </tr> <tr> <td style="text-align: center;">Day</td> <td style="text-align: center;">Hour</td> <td></td> <td></td> </tr> </table>                     Example:                      31st, 10 a.m.                      3110<sub>H</sub> </li> </ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Day	Hour			D9026										
b15 to b12	b11 to b8	b7 to b4	b3 to b0																			
Day	Hour																					
SD212	Clock data	Clock data (minute, second)	<ul style="list-style-type: none"> <li>The minutes and seconds (after the hour) are stored as BCD code as shown below:  <table border="1" style="margin-left: 40px;"> <tr> <td style="text-align: center;">b15 to b12</td> <td style="text-align: center;">b11 to b8</td> <td style="text-align: center;">b7 to b4</td> <td style="text-align: center;">b3 to b0</td> </tr> <tr> <td style="text-align: center;">Minute</td> <td style="text-align: center;">Second</td> <td></td> <td></td> </tr> </table>                     Example:                      35 min, 48 s                      3548<sub>H</sub> </li> </ul>	b15 to b12	b11 to b8	b7 to b4	b3 to b0	Minute	Second			D9027										
b15 to b12	b11 to b8	b7 to b4	b3 to b0																			
Minute	Second																					

\*9: Function version is B or later.

A

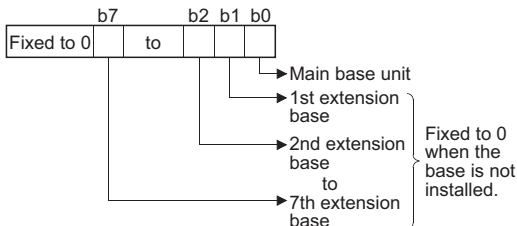
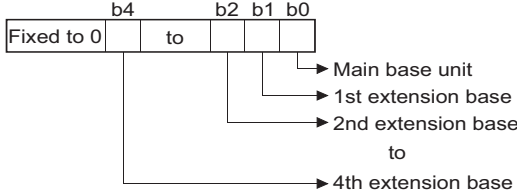
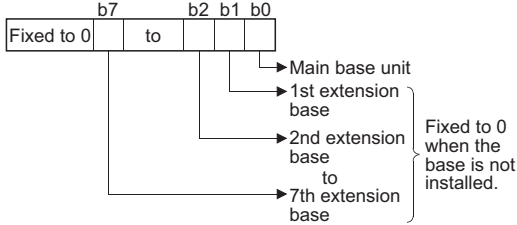
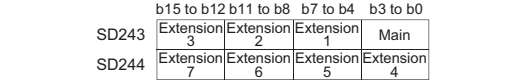
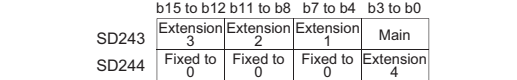
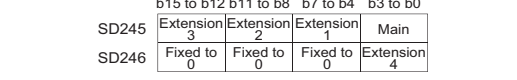
Appendix 4 SPECIAL REGISTER LIST

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																														
SD213	Clock data	Clock data (higher digits of year, day of week)	<ul style="list-style-type: none"> <li>The year (first two digits) and the day of the week are stored as BCD code as shown below.</li> </ul> <div style="display: flex; align-items: center;"> <div style="margin-right: 20px;"> <table border="1" style="border-collapse: collapse; text-align: center;"> <tr> <td>b15 to b12</td> <td>b11 to b8</td> <td>b7 to b4</td> <td>b3 to b0</td> </tr> <tr> <td style="height: 20px;"> </td> <td style="height: 20px;"> </td> <td style="height: 20px;"> </td> <td style="height: 20px;"> </td> </tr> </table> <p>Higher digits of year (19 or 20)</p> </div> <div> <p>Example: 1993, Friday 1905H</p> <table border="1" style="border-collapse: collapse; text-align: center;"> <tr> <th colspan="2">Day of the week</th> </tr> <tr><td>0</td><td>Sunday</td></tr> <tr><td>1</td><td>Monday</td></tr> <tr><td>2</td><td>Tuesday</td></tr> <tr><td>3</td><td>Wednesday</td></tr> <tr><td>4</td><td>Thursday</td></tr> <tr><td>5</td><td>Friday</td></tr> <tr><td>6</td><td>Saturday</td></tr> </table> </div> </div>	b15 to b12	b11 to b8	b7 to b4	b3 to b0					Day of the week		0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday	5	Friday	6	Saturday	S (Request)/U	D9028							
b15 to b12	b11 to b8	b7 to b4	b3 to b0																																	
Day of the week																																				
0	Sunday																																			
1	Monday																																			
2	Tuesday																																			
3	Wednesday																																			
4	Thursday																																			
5	Friday																																			
6	Saturday																																			
SD220	LED display data	LED display data	<ul style="list-style-type: none"> <li>LED display ASCII data (16 characters) stored here.</li> <li>(On the Basic model QCPU, the registers store the message (16 characters of ASCII data) at error occurrence (including annunciator ON).</li> </ul>	S (When changed)	New	QCPU																														
SD221																																				
SD222																																				
SD223																																				
SD224																																				
SD225																																				
SD226																																				
SD227																																				
							<table border="1" style="border-collapse: collapse; text-align: center; margin-left: 40px;"> <tr> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD220</td> <td>SD227</td> </tr> <tr> <td>15th character from the right</td> <td>16th character from the right</td> </tr> <tr> <td>SD221</td> <td>SD226</td> </tr> <tr> <td>13th character from the right</td> <td>14th character from the right</td> </tr> <tr> <td>SD222</td> <td>SD225</td> </tr> <tr> <td>11th character from the right</td> <td>12th character from the right</td> </tr> <tr> <td>SD223</td> <td>SD224</td> </tr> <tr> <td>9th character from the right</td> <td>10th character from the right</td> </tr> <tr> <td>SD224</td> <td>SD223</td> </tr> <tr> <td>7th character from the right</td> <td>8th character from the right</td> </tr> <tr> <td>SD225</td> <td>SD222</td> </tr> <tr> <td>5th character from the right</td> <td>6th character from the right</td> </tr> <tr> <td>SD226</td> <td>SD221</td> </tr> <tr> <td>3rd character from the right</td> <td>4th character from the right</td> </tr> <tr> <td>SD227</td> <td>SD220</td> </tr> <tr> <td>1st character from the right</td> <td>2nd character from the right</td> </tr> </table>	b15 to b8	b7 to b0	SD220	SD227	15th character from the right	16th character from the right	SD221	SD226	13th character from the right	14th character from the right	SD222	SD225	11th character from the right	12th character from the right	SD223	SD224	9th character from the right	10th character from the right	SD224	SD223	7th character from the right	8th character from the right	SD225	SD222	5th character from the right	6th character from the right	SD226	SD221	3rd character from the right
b15 to b8	b7 to b0																																			
SD220	SD227																																			
15th character from the right	16th character from the right																																			
SD221	SD226																																			
13th character from the right	14th character from the right																																			
SD222	SD225																																			
11th character from the right	12th character from the right																																			
SD223	SD224																																			
9th character from the right	10th character from the right																																			
SD224	SD223																																			
7th character from the right	8th character from the right																																			
SD225	SD222																																			
5th character from the right	6th character from the right																																			
SD226	SD221																																			
3rd character from the right	4th character from the right																																			
SD227	SD220																																			
1st character from the right	2nd character from the right																																			
		<ul style="list-style-type: none"> <li>The LED display device data at the time of CHK is not stored in the Basic model QCPU and the Universal model QCPU.</li> </ul>																																		
SD235	Module to which online module change is being performed	The header I/O number of the module to which online module change is being performed /10H	<ul style="list-style-type: none"> <li>10H is added to the value of the header I/O number of which the online module change is being performed.</li> </ul>	S (During online module change)	New	QnPH QnPRH																														
SD240	Base mode	0: Automatic mode 1: Detail mode	<ul style="list-style-type: none"> <li>Stores the base mode.</li> </ul>	S (Initial)	New	QCPU																														
SD241	Extension stage number	0: Main base only 1 to 7: Extension stage number	<ul style="list-style-type: none"> <li>Stores the maximum number of the extension bases being installed.</li> </ul>	S (Initial)	New																															



TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD242	A/Q base differentiation	Base type differentiation 0: QA**B is installed (A mode) 1: Q**B is installed (Q mode)		S (Initial)	New	Qn(H) QnPH QnPRH
	Installed Q base presence/absence	Base type differentiation 0: Base not installed 1: Q**B is installed		S (Initial)	New	Q00J/Q00/Q01
	Installed Q base presence/absence	Base type differentiation 0: Base not installed 1: Q**B is installed	 <ul style="list-style-type: none"> <li>The bits from the third extension stage to the seventh extension stage are fixed to "0" in the Q00UJCPU.</li> <li>The bits from the fifth extension stage to the seventh extension stage are fixed to "0" in the Q00UCPU, Q01UCPU, and Q02UCPU.</li> </ul>	S (Initial)	New	QnU
SD243	No. of base slots	No. of base slots		S (Initial)	New	Qn(H) QnPH QnPRH QnU
SD244			<ul style="list-style-type: none"> <li>As shown above, each area stores the number of slots being installed.</li> <li>The bits from the third extension stage to the seventh extension stage are fixed to "0" in the Q00UJCPU.</li> <li>The bits from the fifth extension stage to the seventh extension stage are fixed to "0" in the Q00UCPU, Q01UCPU, and Q02UCPU.</li> </ul>			
SD243	No. of base slots (Operation status)	No. of base slots		S (Initial)	New	Q00J/Q00/Q01
SD244			<ul style="list-style-type: none"> <li>As shown above, each area stores the number of slots being installed. (Number of set slots when parameter setting has been made)</li> </ul>			
SD245	No. of base slots (Mounting status)	No. of base slots		S (Initial)	New	Q00J/Q00/Q01*9
SD246			<ul style="list-style-type: none"> <li>As shown above, each area stores the number of module-mounted slots of the base unit (actual number of slots of the installed base unit).</li> </ul>			
SD250	Loaded maximum I/O	Loaded maximum I/O No.	<ul style="list-style-type: none"> <li>When SM250 goes from OFF to ON, the upper 2 digits of the final I/O number plus 1 of the modules loaded are stored as BIN values.</li> </ul>	S (Request END)	New	Qn(H) QnPH QnPRH
			<ul style="list-style-type: none"> <li>The upper 2 digits of the final I/O number plus 1 of the modules loaded are stored as BIN values.</li> </ul>	S (Initial)	New	Q00J/Q00/Q01 QnU

\*9: Function version is B or later.

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9 □ □ □	Corresponding CPU	
SD254	MELSECNET/10. MELSECNET/H information	Number of modules installed	• Indicates the number of mounted MELSECNET/10 modules or MELSECNET/H modules.	S (Initial)	New	QCPU	
SD255		Information from 1st module	I/O No.				• Indicates I/O number of mounted MELSECNET/10 module or MELSECNET/H module
SD256			Network No.				• Indicates network No. of mounted MELSECNET/10 module or MELSECNET/H module
SD257			Group number			• Indicates group No. of mounted MELSECNET/10 module or MELSECNET/H module	
SD258			Station No.			• Indicates station No. of mounted MELSECNET/10 module or MELSECNET/H module	
SD259			Standby information			• In the case of standby stations, the module number of the standby station is stored. (1 to 4)	
SD260 to SD264			Information from 2nd module			• Configuration is identical to that for the first module.	
SD265 to SD269		Information from 3rd module	• Configuration is identical to that for the first module.				
SD270 to SD274		Information from 4th module	• Configuration is identical to that for the first module.				
SD280	CC-Link error	Error detection status	<p>1) When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON).</p> <p>2) When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON).</p> <p>3) Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.</p> <p>The above module Nos. n are in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)</p>	S (Error)	New	Qn(H) QnPH QnPRH	

\*10: The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

\*11: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU
SD281	CC-Link error	Error detection status	<p>1) When Xn0 of the mounted CC-Link module turns ON, the bit of the corresponding station turns to 1 (ON).</p> <p>2) When either Xn1 or XnF of the mounted CC-Link module turns OFF, the bit of the corresponding station turns to 1 (ON).</p> <p>3) Turns to 1 (ON) when communication between the mounted CC-Link module and CPU module cannot be made.</p> <p>The above module Nos. n are in order of the head I/O numbers. (However, the one where parameter setting has not been made is not counted.)</p>	S (Error)	New	Qn(H) <sup>*14</sup> QnPH <sup>*14</sup> QnPRH <sup>*15</sup>
SD286	Device assignment	Points assigned to M (for extension)	<ul style="list-style-type: none"> <li>The number of points assigned to M is stored with 32 bits.</li> <li>Even if the points assigned to M are 32k points or less, the points are stored.</li> </ul>	S (Initial)	New	QnU <sup>*16</sup>
SD287		Points assigned to B (for extension)	<ul style="list-style-type: none"> <li>The number of points assigned to B is stored with 32 bits.</li> <li>Even if the points assigned to B are 32k points or less, the points are stored.</li> </ul>			
SD288		Number of points assigned for X	• Stores the number of points currently set for X devices			
SD289		Number of points assigned for Y	• Stores the number of points currently set for Y devices			
SD290	Device assignment (Same as parameter contents)	Number of points assigned for M	• Stores the number of points currently set for M devices	S (Initial)	New	QCPU
SD291		Number of points assigned for L	• Stores the number of points currently set for L devices			
SD292		Number of points assigned for B	• Stores the number of points currently set for B devices			
SD293		Number of points assigned for F	• Stores the number of points currently set for F devices			
SD294		Number of points assigned for SB	• Stores the number of points currently set for SB devices			
SD295		Number of points assigned for V	• Stores the number of points currently set for V devices			
SD296		Number of points assigned for S	• Stores the number of points currently set for S devices			
SD297	Number of points assigned for T	• Stores the number of points currently set for T device				
SD298	Device assignment (Same as parameter contents)	Number of points assigned for ST	• Stores the number of points currently set for ST devices	S (Initial)	New	QCPU
SD299		Number of points assigned for C	• Stores the number of points currently set for C devices			
SD300		Number of points assigned for D	• Stores the number of points currently set for D devices			
SD301		Number of points assigned for W	• Stores the number of points currently set for W devices			
SD302		Number of points assigned for SW	• Stores the number of points currently set for SW devices			
SD303		Number of points assigned for V	• Stores the number of points currently set for V devices			
SD304		Number of points assigned for S	• Stores the number of points currently set for S devices			

\*14: The module whose first 5 digits of serial No. is "08032" or later.  
 \*15: The module whose first 5 digits of serial No. is "09012" or later.  
 \*16: The module whose first 5 digits of serial No. is "10042" or later.

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.4 Special register

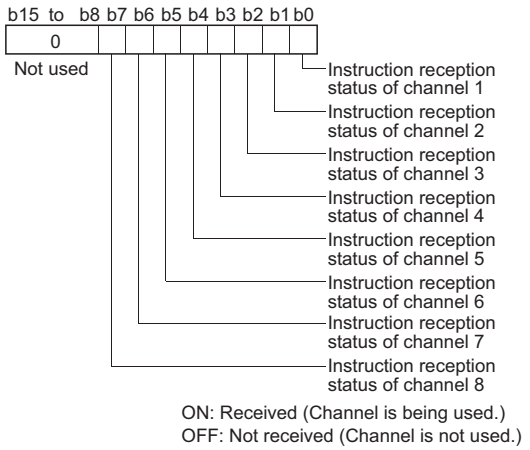
Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9 □ □ □	Corresponding CPU	
SD305	Device assignment (Index register)	16 bit modification Number of points assigned for Z	<ul style="list-style-type: none"> <li>Stores the number of points of index register (Z) to be modified in the range of 16 bits. (The assignment is set by the ZR device index modification setting parameter.)</li> </ul>	S (Initial)	New	QnU	
SD306	Device assignment (Same as parameter contents)	Number of points assigned for ZR (for extension)	<ul style="list-style-type: none"> <li>Stores the number of ZR device points (except the number of points of extended data register (D) and extended link register (W)). The number of assignment points of ZR device is stored into this SD only when 1k point or more is set to the extended data register (D) and extended link register (W).</li> </ul>	S (Initial)	New	QnU <sup>*17</sup>	
SD307							
SD308	Device assignment (assignment including the number of points set to the extended data register (D) and extended link register (W))	Number of points assigned for D (for inside + for extension)	<ul style="list-style-type: none"> <li>Stores the total number of points of the extended data register (D) and data register in internal device memory area (stores the value in 32-bit binary).</li> </ul>	S (Initial)	New	QnU <sup>*17</sup>	
SD309							
SD310							
SD311	Number of points assigned for W (for inside + for extension)	<ul style="list-style-type: none"> <li>Stores the total number of points of the extended link register (W) and link register in internal device memory area (stores the value in 32-bit binary).</li> </ul>	S (Initial)	New	QnU <sup>*17</sup>		
SD311							
SD315	Time reserved for communication processing	Time reserved for communication processing	<ul style="list-style-type: none"> <li>Reserves the designated time for communication processing with GX Developer or other units.</li> <li>The greater the value is designated, the shorter the response time for communication with other devices (GX Developer, serial communication units) becomes.</li> <li>If the designated value is out of the range above, it is processed that no setting is made.</li> <li>Setting range: 1 to 100 ms</li> <li>Note that the scan time becomes longer by the designated time.</li> </ul>	U	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH	
SD340	Ethernet information	No. of modules installed	<ul style="list-style-type: none"> <li>Indicates the number of mounted Ethernet module.</li> </ul>	S (Initial)	New	QCPU	
SD341		Information of 1st module	I/O No.				<ul style="list-style-type: none"> <li>Indicates I/O No. of mounted Ethernet module</li> </ul>
SD342			Network No.				<ul style="list-style-type: none"> <li>Indicates network No. of mounted Ethernet module</li> </ul>
SD343			Group No.				<ul style="list-style-type: none"> <li>Indicates group No. of mounted Ethernet module</li> </ul>
SD344			Station No.				<ul style="list-style-type: none"> <li>Indicates station No. of mounted Ethernet module</li> </ul>
SD345 to SD346			Empty				<ul style="list-style-type: none"> <li>Empty (With QCPU, the Ethernet module IP address of the 1st module is stored in buffer memory.)</li> </ul>
SD347			Empty				<ul style="list-style-type: none"> <li>Empty (With QCPU, the Ethernet module error code of the 1st module is read with the ERRRD instruction.)</li> </ul>
SD348 to SD354	Ethernet information	Information from 2nd module	<ul style="list-style-type: none"> <li>Configuration is identical to that for the first module.</li> </ul>	S (Initial)	New	Qn(H) QnPH QnPRH QnU <sup>*10</sup>	
SD355 to SD361		Information from 3rd module	<ul style="list-style-type: none"> <li>Configuration is identical to that for the first module.</li> </ul>				
SD362 to SD368		Information from 4th module	<ul style="list-style-type: none"> <li>Configuration is identical to that for the first module.</li> </ul>				

\*10: The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

\*11: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*17: The Universal model QCPU except the Q00UJCPU.

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU
SD380	Ethernet instruction reception status	Instruction reception status of 1st module	 <p>ON: Received (Channel is being used.) OFF: Not received (Channel is not used.)</p>	S (Instruction execution)	New	QnPRH
SD381	Ethernet instruction reception status	Instruction reception status of 2nd module	• Configuration is identical to that for the first module.			
SD382		Instruction reception status of 3rd module	• Configuration is identical to that for the first module.			
SD383		Instruction reception status of 4th module	• Configuration is identical to that for the first module.			

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.4 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU									
SD393	Multiple CPU system information	Number of multiple CPUs	<ul style="list-style-type: none"> <li>The number of CPU modules that comprise the multiple CPU system is stored. (1 to 3, Empty also included)</li> </ul>	S (Initial)	New	Q00/Q01 <sup>*9</sup> QnU									
SD394		CPU mounting information	<ul style="list-style-type: none"> <li>The CPU module types of No. 1 CPU to 3 and whether the CPU modules are mounted or not are stored.</li> </ul> <div style="text-align: center;"> <table border="1" style="margin: auto;"> <tr> <td style="text-align: center;">b15 to b12</td> <td style="text-align: center;">b11 to b8</td> <td style="text-align: center;">b7 to b4</td> <td style="text-align: center;">b3 to b0</td> </tr> <tr> <td style="text-align: center;">Empty (0)</td> <td style="text-align: center;">CPU No.3</td> <td style="text-align: center;">CPU No.2</td> <td style="text-align: center;">CPU No.1</td> </tr> </table>   </div>			b15 to b12	b11 to b8	b7 to b4	b3 to b0	Empty (0)	CPU No.3	CPU No.2	CPU No.1		Q00/Q01 <sup>*9</sup>
b15 to b12		b11 to b8	b7 to b4			b3 to b0									
Empty (0)		CPU No.3	CPU No.2			CPU No.1									
SD395		Multiple CPU number	<ul style="list-style-type: none"> <li>In a multiple CPU system configuration, the CPU number of the host CPU is stored.</li> <li>CPU No. 1: 1, CPU No. 2: 2, CPU No. 3: 3, CPU No. 4: 4</li> </ul>			S (Initial)	New	Q00/Q01 <sup>*9</sup> Qn(H) <sup>*9</sup> QnPH QnU							
SD396		No. 1 CPU operation status	The operation information of each CPU No. is stored. (The information on the number of multiple CPUs indicated in SD393 is stored.)			S (END processing error)	New	Q00/Q01 <sup>*9</sup> QnU <sup>*17</sup>							
SD397	No. 2 CPU operation status														
SD398	No. 3 CPU operation status														
SD399	No. 4 CPU operation status														
SD399	No. 4 CPU operation status	<div style="text-align: center;"> <table border="1" style="margin: auto;"> <tr> <td style="text-align: center;">b15 to b14</td> <td style="text-align: center;">b8 to b7</td> <td style="text-align: center;">b4 to b3</td> <td style="text-align: center;">b2 to b0</td> </tr> <tr> <td style="text-align: center;">Vacancy</td> <td style="text-align: center;">Classification</td> <td style="text-align: center;">Operation status</td> <td></td> </tr> </table>                       mounted                      0: Not mounted                      1: Mounted                 </div> <div style="display: flex; justify-content: space-around;"> <div>                     0: Normal                      1: Minor fault                      2: Medium fault                      3: Major fault                      Fh: Reset                 </div> <div>                     0: RUN                      2: STOP                      3: PAUSE                      4: Initial                      Fh: Reset                 </div> </div>		b15 to b14	b8 to b7				b4 to b3	b2 to b0	Vacancy	Classification	Operation status		QnU <sup>*11</sup>
b15 to b14	b8 to b7	b4 to b3	b2 to b0												
Vacancy	Classification	Operation status													

\*9: Function version is B or later.

\*11: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*17: The Universal model QCPU except the Q00UJCPU.

(3) System clocks/counters

TableApp.4.5 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD412	1 second counter	Number of counts in 1-second units	<ul style="list-style-type: none"> <li>Following programmable controller CPU module RUN, 1 is added each second</li> <li>Count repeats from 0 to 32767 to -32768 to 0</li> </ul>	S (Status change)	D9022	QCPU
SD414	2n second clock setting	2n second clock units	<ul style="list-style-type: none"> <li>Stores value n of 2n second clock (Default is 30)</li> <li>Setting can be made between 1 and 32767</li> </ul>	U	New	
SD415	2nms clock setting	2nms clock units	<ul style="list-style-type: none"> <li>Stores value n of 2nms clock (Default is 30)</li> <li>Setting can be made between 1 and 32767</li> </ul>	U	New	
SD420	Scan counter	Number of counts in each scan	<ul style="list-style-type: none"> <li>Incremented by 1 for each scan execution after the CPU module is set to RUN. (Not counted by the scan in an initial execution type program.)</li> <li>Count repeats from 0 to 32767 to -32768 to 0</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH QnU
			<ul style="list-style-type: none"> <li>Incremented by 1 for each scan execution after the CPU module is set to RUN.</li> <li>Count repeats from 0 to 32767 to -32768 to 0</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01
SD430	Low speed scan counter	Number of counts in each scan	<ul style="list-style-type: none"> <li>Incremented by 1 for each scan execution after the CPU module is set to RUN.</li> <li>Count repeats from 0 to 32767 to -32768 to 0</li> <li>Used only for low speed execution type programs</li> </ul>	S (Every END processing)	New	Qn(H) QnPH

## (4) Scan information

TableApp.4.6 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD500	Execution program No.	Program No. in execution	• Program number of program currently being executed is stored as BIN value.	S (Status change)	New	Qn(H) QnPH QnPRH QnU
SD510	Low speed execution type program No.	Low speed execution type program No. in execution	• Program number of low speed execution type program No. currently being executed is stored as BIN value. • Enabled only when SM510 is ON.	S (Every END processing)	New	Qn(H) QnPH
SD520	Current scan time	Current scan time (in 1 ms units)	• The current scan time is stored into SD520 and SD521. (Measurement is made in 100 $\mu$ s units. (For the Universal model QCPU, in 1 $\mu$ s units.)) SD520: Stores the ms place. (Storage range: 0 to 65535) SD521: Stores the $\mu$ s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999)) (Example) When the current scan time is 23.6ms, the following values are stored. SD520 = 23 SD521 = 600	S (Every END processing)	D9018 format change	QCPU
SD521		Current scan time (in 100 $\mu$ s units)		S (Every END processing)	New	
SD522	Initial scan time	Initial scan time (in 1 ms units)	• Stores the scan time of an initial execution type program into SD522 and SD523. (Measurement is made in 100 $\mu$ s units. (For the Universal model QCPU, in 1 $\mu$ s units.)) SD522: Stores the ms place. (Storage range: 0 to 65535) SD523: Stores the $\mu$ s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (First END processing)	New	
SD523		Initial scan time (in 100 $\mu$ s units)				
SD524	Minimum scan time	Minimum scan time (in 1 ms units)	• Stores the minimum value of the scan time except that of an initial execution type program into SD524 and SD525. (Measurement is made in 100 $\mu$ s units. (For the Universal model QCPU, in 1 $\mu$ s units.)) SD524: Stores the ms place. (Storage range: 0 to 65535) SD525: Stores the $\mu$ s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	D9017 format change	Qn(H) QnPH QnPRH QnU
SD525		Minimum scan time (in 100 $\mu$ s units)		S (Every END processing)	New	
SD526	Maximum scan time	Maximum scan time (in 1 ms units)	• Stores the maximum value of the scan time except that of an initial execution type program into SD526 and SD527. (Measurement is made in 100 $\mu$ s units. (For the Universal model QCPU, in 1 $\mu$ s units.)) SD526: Stores the ms place. (Storage range: 0 to 65535) SD527: Stores the $\mu$ s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	D9019 format change	
SD527		Maximum scan time (in 100 $\mu$ s units)			New	
SD528	Current scan time for low speed execution type programs	Current scan time (in 1 ms units)	• Stores the current scan time of a low speed execution type program into SD528 and SD529. (Measurement is made in 100 $\mu$ s units.) SD528: Stores the ms place. (Storage range: 0 to 65535) SD529: Stores the $\mu$ s place. (Storage range: 0 to 900)	S (Every END processing)	New	
SD529		Current scan time (in 100 $\mu$ s units)				
SD532	Minimum scan time for low speed execution type programs	Minimum scan time (in 1 ms units)	• Stores the minimum value of the scan time of a low speed execution type program into SD532 and SD533. (Measurement is made in 100 $\mu$ s units.) SD532: Stores the ms place. (Storage range: 0 to 65535) SD533: Stores the $\mu$ s place. (Storage range: 0 to 900)	S (Every END processing)	New	Qn(H) QnPH
SD533		Minimum scan time (in 100 $\mu$ s units)				
SD534	Maximum scan time for low speed execution type programs	Maximum scan time (in 1 ms units)	• Stores the maximum value of the scan time except that of the first scan of a low speed execution type program into SD534 and SD535. (Measurement is made in 100 $\mu$ s units.) SD534: Stores the ms place. (Storage range: 0 to 65535) SD535: Stores the $\mu$ s place. (Storage range: 0 to 900)	S (Every END processing)	New	
SD535		Maximum scan time (in 100 $\mu$ s units)				
SD540	END processing time	END processing time (in 1 ms units)	• Stores the time from the end of a scan execution type program to the start of the next scan into SD540 and SD541. (Measurement is made in 100 $\mu$ s units. (For the Universal model QCPU, in 1 $\mu$ s units.)) SD540: Stores the ms place. (Storage range: 0 to 65535) SD541: Stores the $\mu$ s place. (Storage range: 0 to 900) (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))	S (Every END processing)	New	Qn(H) QnPH QnPRH QnU
SD541		END processing time (in 100 $\mu$ s units)				

A

TableApp.4.6 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU																																																																							
SD524	Minimum scan time	Minimum scan time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the minimum value of the scan time into SD524 and SD525. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD524: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD525: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01																																																																							
SD525		Minimum scan time (in 100 $\mu$ s units)					SD526	Maximum scan time	Maximum scan time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the maximum value of the scan time into SD526 and SD527. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD526: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD527: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Every END processing)	New	SD527	Maximum scan time (in 100 $\mu$ s units)	SD540	END processing time	END processing time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the time from when the scan program ends until the next scan starts into SD540 and SD541. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD540: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD541: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Every END processing)	New	SD541	END processing time (in 100 $\mu$ s units)	SD542	Constant scan wait time	Constant scan wait time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the wait time for constant scan setting into SD542 and SD543. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD542: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD543: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> </ul>	S (Every END processing)	New	QCPU	SD543	Constant scan wait time (in 100 $\mu$ s units)	SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD544: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD545: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Cleared to 0 after the end of one low speed scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH	SD545	Cumulative execution time for low speed execution type programs (in 100 $\mu$ s units)	SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	SD547	Execution time for low speed execution type programs (in 100 $\mu$ s units)	SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH	SD549	Scan execution type program execution time (in 100 $\mu$ s units)	SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>
SD526	Maximum scan time	Maximum scan time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the maximum value of the scan time into SD526 and SD527. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD526: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD527: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Every END processing)	New																																																																								
SD527		Maximum scan time (in 100 $\mu$ s units)					SD540	END processing time	END processing time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the time from when the scan program ends until the next scan starts into SD540 and SD541. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD540: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD541: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Every END processing)	New	SD541	END processing time (in 100 $\mu$ s units)	SD542	Constant scan wait time	Constant scan wait time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the wait time for constant scan setting into SD542 and SD543. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD542: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD543: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> </ul>	S (Every END processing)	New	QCPU	SD543	Constant scan wait time (in 100 $\mu$ s units)	SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD544: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD545: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Cleared to 0 after the end of one low speed scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH	SD545	Cumulative execution time for low speed execution type programs (in 100 $\mu$ s units)	SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)		New	SD547	Execution time for low speed execution type programs (in 100 $\mu$ s units)	SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH	SD549	Scan execution type program execution time (in 100 $\mu$ s units)	SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New		SD552	Module service interval (in 100 $\mu$ s units)		
SD540	END processing time	END processing time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the time from when the scan program ends until the next scan starts into SD540 and SD541. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD540: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD541: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Every END processing)	New																																																																								
SD541		END processing time (in 100 $\mu$ s units)					SD542	Constant scan wait time	Constant scan wait time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the wait time for constant scan setting into SD542 and SD543. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD542: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD543: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> </ul>	S (Every END processing)	New	QCPU	SD543	Constant scan wait time (in 100 $\mu$ s units)	SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD544: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD545: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Cleared to 0 after the end of one low speed scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH	SD545	Cumulative execution time for low speed execution type programs (in 100 $\mu$ s units)	SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)		New	SD547	Execution time for low speed execution type programs (in 100 $\mu$ s units)	SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH	SD549	Scan execution type program execution time (in 100 $\mu$ s units)	SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	SD552		Module service interval (in 100 $\mu$ s units)										
SD542	Constant scan wait time	Constant scan wait time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the wait time for constant scan setting into SD542 and SD543. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD542: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD543: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> </ul>	S (Every END processing)	New	QCPU																																																																							
SD543		Constant scan wait time (in 100 $\mu$ s units)					SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD544: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD545: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Cleared to 0 after the end of one low speed scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH	SD545	Cumulative execution time for low speed execution type programs (in 100 $\mu$ s units)	SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New		SD547	Execution time for low speed execution type programs (in 100 $\mu$ s units)	SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH	SD549	Scan execution type program execution time (in 100 $\mu$ s units)	SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	SD552	Module service interval (in 100 $\mu$ s units)																				
SD544	Cumulative execution time for low speed execution type programs	Cumulative execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the cumulative execution time of a low speed execution type program into SD544 and SD545. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD544: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD545: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Cleared to 0 after the end of one low speed scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH																																																																							
SD545		Cumulative execution time for low speed execution type programs (in 100 $\mu$ s units)					SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New		SD547	Execution time for low speed execution type programs (in 100 $\mu$ s units)	SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH	SD549	Scan execution type program execution time (in 100 $\mu$ s units)	SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	SD552	Module service interval (in 100 $\mu$ s units)																													
SD546	Execution time for low speed execution type programs	Execution time for low speed execution type programs (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a low speed execution type program during one scan into SD546 and SD547. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD546: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD547: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New																																																																								
SD547		Execution time for low speed execution type programs (in 100 $\mu$ s units)				SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH	SD549	Scan execution type program execution time (in 100 $\mu$ s units)	SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	SD552	Module service interval (in 100 $\mu$ s units)																																							
SD548	Scan execution type program execution time	Scan execution type program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan execution type program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Qn(H) QnPH QnPRH																																																																							
SD549		Scan execution type program execution time (in 100 $\mu$ s units)					SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU	SD549	Scan program execution time (in 100 $\mu$ s units)	SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	SD552		Module service interval (in 100 $\mu$ s units)																																														
SD548	Scan program execution time	Scan program execution time (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the execution time of a scan program during one scan into SD548 and SD549. (Measurement is made in 100 <math>\mu</math>s units. (For the Universal model QCPU, in 1 <math>\mu</math>s units.))</li> <li>SD548: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD549: Stores the <math>\mu</math>s place. (Storage range: 0 to 900 (For the Universal model QCPU, storage range is 0 to 999))</li> <li>Stored every scan.</li> </ul>	S (Every END processing)	New	Q00J/Q00/Q01 QnU																																																																							
SD549		Scan program execution time (in 100 $\mu$ s units)					SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH	SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New	SD552	Module service interval (in 100 $\mu$ s units)																																																								
SD550	Service interval measurement module	Unit/module No.	<ul style="list-style-type: none"> <li>Sets I/O number for module that measures service interval.</li> </ul>	U	New	Qn(H) QnPH QnPRH																																																																							
SD551	Service interval time	Module service interval (in 1 ms units)	<ul style="list-style-type: none"> <li>Stores the service interval for the module specified in SD550 into SD551 and SD552 when SM551 is turned ON. (Measurement is made in 100 <math>\mu</math>s units.)</li> <li>SD551: Stores the ms place. (Storage range: 0 to 65535)</li> <li>SD552: Stores the <math>\mu</math>s place. (Storage range: 0 to 900)</li> </ul>	S (Request)	New																																																																								
SD552		Module service interval (in 100 $\mu$ s units)																																																																											



(5) Memory card

TableApp.4.7 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																
SD600	Memory card typs	Memory card typs	<ul style="list-style-type: none"> <li>Indicates the type of the memory card installed.</li> </ul> <p>(The bits for the drive 1 (RAM) type and drive 2 (ROM) type are fixed to "0" in the Q00UJCPU, Q00UCPU, and Q01UCPU.)</p>	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU																
SD602	Drive 1 (Memory card RAM) capacity	Drive 1 capacity	<ul style="list-style-type: none"> <li>Drive 1 capacity is stored in 1 k byte units.</li> <li>(Empty capacity after format is stored.)</li> </ul>	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU <sup>2</sup>																
SD603	Drive 2 (Memory card ROM) capacity	Drive 2 capacity	<ul style="list-style-type: none"> <li>Drive 2 capacity is stored in 1 k byte units.<sup>*1</sup></li> </ul>	S (Initial and card removal)	New	Qn(H) QnPH QnPRH QnU <sup>2</sup>																
SD604	Memory card use conditions	Memory card use conditions	<ul style="list-style-type: none"> <li>The use conditions for memory card are stored as bit patterns . (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr> <td>b0 : Boot operation (QBT)</td> <td>b8 : Not used</td> </tr> <tr> <td>b1 : Parameters (QPA)</td> <td>b9 : CPU fault history (QFD)</td> </tr> <tr> <td>b2 : Device comments (QCD)</td> <td>b10 : Not used</td> </tr> <tr> <td>b3 : Device initial value (QDI)</td> <td>b11 : Local device (QDL)</td> </tr> <tr> <td>b4 : File register R (QDR)</td> <td>b12 : Not used</td> </tr> <tr> <td>b5 : Sampling trace (QTD)</td> <td>b13 : Not used</td> </tr> <tr> <td>b6 : Not used</td> <td>b14 : Not used</td> </tr> <tr> <td>b7 : Not used</td> <td>b15 : Not used</td> </tr> </table>	b0 : Boot operation (QBT)	b8 : Not used	b1 : Parameters (QPA)	b9 : CPU fault history (QFD)	b2 : Device comments (QCD)	b10 : Not used	b3 : Device initial value (QDI)	b11 : Local device (QDL)	b4 : File register R (QDR)	b12 : Not used	b5 : Sampling trace (QTD)	b13 : Not used	b6 : Not used	b14 : Not used	b7 : Not used	b15 : Not used	S (Status change)	New	Qn(H) QnPH QnPRH
	b0 : Boot operation (QBT)	b8 : Not used																				
b1 : Parameters (QPA)	b9 : CPU fault history (QFD)																					
b2 : Device comments (QCD)	b10 : Not used																					
b3 : Device initial value (QDI)	b11 : Local device (QDL)																					
b4 : File register R (QDR)	b12 : Not used																					
b5 : Sampling trace (QTD)	b13 : Not used																					
b6 : Not used	b14 : Not used																					
b7 : Not used	b15 : Not used																					
	Memory card use conditions	Memory card use conditions	<ul style="list-style-type: none"> <li>The use conditions for memory card are stored as bit patterns . (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr> <td>b0 : Boot operation (QBT)<sup>*1</sup></td> <td>b8 : Not used</td> </tr> <tr> <td>b1 : Parameters (QPA)</td> <td>b9 : Not used</td> </tr> <tr> <td>b2 : Device comments (QCD)</td> <td>b10 : Not used</td> </tr> <tr> <td>b3 : Device initial value (QDI)<sup>*2</sup></td> <td>b11 : Local device (QDL)</td> </tr> <tr> <td>b4 : File register R (QDR)</td> <td>b12 : Not used</td> </tr> <tr> <td>b5 : Sampling trace (QTD)</td> <td>b13 : Not used</td> </tr> <tr> <td>b6 : Not used</td> <td>b14 : Not used</td> </tr> <tr> <td>b7 : Backup data (QBP)<sup>*3</sup></td> <td>b15 : Not used</td> </tr> </table> <p><sup>*1</sup>: Turned ON at boot start and OFF at boot completion.  <sup>*2</sup>: Turned ON when reflection of device initial value is started and OFF when reflection of device initial value is completed.  <sup>*3</sup>: The module whose first 5 digits of serial No. is "10102" or later.</p>	b0 : Boot operation (QBT) <sup>*1</sup>	b8 : Not used	b1 : Parameters (QPA)	b9 : Not used	b2 : Device comments (QCD)	b10 : Not used	b3 : Device initial value (QDI) <sup>*2</sup>	b11 : Local device (QDL)	b4 : File register R (QDR)	b12 : Not used	b5 : Sampling trace (QTD)	b13 : Not used	b6 : Not used	b14 : Not used	b7 : Backup data (QBP) <sup>*3</sup>	b15 : Not used	S (Status change)	New	QnU <sup>2</sup>
b0 : Boot operation (QBT) <sup>*1</sup>	b8 : Not used																					
b1 : Parameters (QPA)	b9 : Not used																					
b2 : Device comments (QCD)	b10 : Not used																					
b3 : Device initial value (QDI) <sup>*2</sup>	b11 : Local device (QDL)																					
b4 : File register R (QDR)	b12 : Not used																					
b5 : Sampling trace (QTD)	b13 : Not used																					
b6 : Not used	b14 : Not used																					
b7 : Backup data (QBP) <sup>*3</sup>	b15 : Not used																					

\*1: When the Q2MEM-8MBA is used, value stored in the special register SD603 differs depending on the combination of the serial number of the High Performance model QCPU and the manufacture control number of the ATA card. For details, refer to QCPU User's Manual (Hardware Design, Maintenance and Inspection).  
\*2: The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.7 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																
SD620	Drive 3/4 type	Drive 3/4 type	<ul style="list-style-type: none"> <li>Indicates the drive 3/4 type.</li> </ul> <p>(The bits for the drive 3 (standard RAM) type is fixed to "0" in the Q00UJCPU.)</p>	S (Initial)	New	Qn(H) QnPH QnPRH QnU																
			<ul style="list-style-type: none"> <li>Indicates the drive 3/4 type.</li> </ul>	S (Initial)	New	Q00J/Q00/Q01																
SD622	Drive 3 (Standard RAM) capacity	Drive 3 capacity	<ul style="list-style-type: none"> <li>Drive 3 capacity is stored in 1 k byte units. (Empty capacity after format is stored.)</li> </ul>	S (Initial)	New	Qn(H) QnPH QnPRH QnU																
			<ul style="list-style-type: none"> <li>Drive 3 capacity is stored in 1k byte units.</li> </ul>	S (Initial)	New	Q00J/Q00/Q01																
SD623	Drive 4 (Standard ROM) capacity	Drive 4 capacity	<ul style="list-style-type: none"> <li>Drive 4 capacity is stored in 1 k byte units. (Empty capacity after format is stored.)</li> </ul>	S (Initial)	New	Qn(H) QnPH QnPRH QnU																
			<ul style="list-style-type: none"> <li>Drive 4 capacity is stored in 1k byte units.</li> </ul>	S (Initial)	New	Q00J/Q00/Q01																
SD624	Drive 3/4 use conditions	Drive 3/4 use conditions	<ul style="list-style-type: none"> <li>The conditions for usage for drive 3/4 are stored as bit patterns. (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr> <td>b0 : Boot operation (QBT)</td> <td>b8 : Not used</td> </tr> <tr> <td>b1 : Parameters (QPA)</td> <td>b9 : CPU fault history (QFD)</td> </tr> <tr> <td>b2 : Device comments (QCD)</td> <td>b10 : SFC trace (QTS)</td> </tr> <tr> <td>b3 : Device initial value (QDI)</td> <td>b11 : Local device (QDL)</td> </tr> <tr> <td>b4 : File register (QDR)</td> <td>b12 : Not used</td> </tr> <tr> <td>b5 : Sampling trace (QTD)</td> <td>b13 : Not used</td> </tr> <tr> <td>b6 : Not used</td> <td>b14 : Not used</td> </tr> <tr> <td>b7 : Not used</td> <td>b15 : Not used</td> </tr> </table>	b0 : Boot operation (QBT)	b8 : Not used	b1 : Parameters (QPA)	b9 : CPU fault history (QFD)	b2 : Device comments (QCD)	b10 : SFC trace (QTS)	b3 : Device initial value (QDI)	b11 : Local device (QDL)	b4 : File register (QDR)	b12 : Not used	b5 : Sampling trace (QTD)	b13 : Not used	b6 : Not used	b14 : Not used	b7 : Not used	b15 : Not used	S (Status change)	New	Qn(H) QnPH QnPRH
			b0 : Boot operation (QBT)	b8 : Not used																		
b1 : Parameters (QPA)	b9 : CPU fault history (QFD)																					
b2 : Device comments (QCD)	b10 : SFC trace (QTS)																					
b3 : Device initial value (QDI)	b11 : Local device (QDL)																					
b4 : File register (QDR)	b12 : Not used																					
b5 : Sampling trace (QTD)	b13 : Not used																					
b6 : Not used	b14 : Not used																					
b7 : Not used	b15 : Not used																					
<ul style="list-style-type: none"> <li>The conditions for usage for drive 3/4 are stored as bit patterns. (In use when ON)</li> <li>The significance of these bit patterns is indicated below:</li> </ul> <table border="1"> <tr> <td>b0 : Not used</td> <td>b8 : Module error log<sup>2</sup></td> </tr> <tr> <td>b1 : Parameters (QPA)</td> <td>b9 : Not used</td> </tr> <tr> <td>b2 : Device comments (QCD)</td> <td>b10 : Not used</td> </tr> <tr> <td>b3 : Device initial value (QDI)<sup>*1</sup></td> <td>b11 : Local device (QDL)</td> </tr> <tr> <td>b4 : File register (QDR)</td> <td>b12 : Not used</td> </tr> <tr> <td>b5 : Sampling trace (QTD)</td> <td>b13 : Not used</td> </tr> <tr> <td>b6 : Not used</td> <td>b14 : Not used</td> </tr> <tr> <td>b7 : Not used</td> <td>b15 : Not used</td> </tr> </table> <p>*1: Turned ON at boot start and OFF at boot completion. *2: The modules whose first 5 digits of serial No. is "11043" or later.</p>	b0 : Not used	b8 : Module error log <sup>2</sup>	b1 : Parameters (QPA)	b9 : Not used	b2 : Device comments (QCD)	b10 : Not used	b3 : Device initial value (QDI) <sup>*1</sup>	b11 : Local device (QDL)	b4 : File register (QDR)	b12 : Not used	b5 : Sampling trace (QTD)	b13 : Not used	b6 : Not used	b14 : Not used	b7 : Not used	b15 : Not used	S (Status change)	New	QnU			
b0 : Not used	b8 : Module error log <sup>2</sup>																					
b1 : Parameters (QPA)	b9 : Not used																					
b2 : Device comments (QCD)	b10 : Not used																					
b3 : Device initial value (QDI) <sup>*1</sup>	b11 : Local device (QDL)																					
b4 : File register (QDR)	b12 : Not used																					
b5 : Sampling trace (QTD)	b13 : Not used																					
b6 : Not used	b14 : Not used																					
b7 : Not used	b15 : Not used																					
SD624	Drive 3/4 use conditions	Drive 3/4 use conditions	<ul style="list-style-type: none"> <li>The conditions for usage for drive 3/4 are stored as bit patterns.</li> </ul>	S (Status change)	New	Q00J/Q00/Q01																
SD640	File register drive	Drive number:	<ul style="list-style-type: none"> <li>Stores drive number being used by file register</li> </ul>	S (Status change) *10	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU <sup>*3</sup>																

\*3: The Universal model QCPU except the Q00UJCPU.

\*10: On the Basic model QCPU, data is set at STOP to RUN or RSET instruction execution after parameter execution.

TableApp.4.7 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																						
SD641	File register file name	File register file name	<ul style="list-style-type: none"> <li>Stores file register file name (with extension) selected at parameters or by use of QDRSET instruction as ASCII code.</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD641</td> <td>2nd character</td> <td>1st character</td> </tr> <tr> <td>SD642</td> <td>4th character</td> <td>3rd character</td> </tr> <tr> <td>SD643</td> <td>6th character</td> <td>5th character</td> </tr> <tr> <td>SD644</td> <td>8th character</td> <td>7th character</td> </tr> <tr> <td>SD645</td> <td>1st character of extension</td> <td>2EH(.)</td> </tr> <tr> <td>SD646</td> <td>3rd character of extension</td> <td>2nd character of the extension</td> </tr> </table>		b15 to b8	b7 to b0	SD641	2nd character	1st character	SD642	4th character	3rd character	SD643	6th character	5th character	SD644	8th character	7th character	SD645	1st character of extension	2EH(.)	SD646	3rd character of extension	2nd character of the extension	S (Status change)	New	Qn(H) QnPH QnPRH QnU <sup>*3</sup>	
				b15 to b8	b7 to b0																							
SD641				2nd character	1st character																							
SD642				4th character	3rd character																							
SD643				6th character	5th character																							
SD644				8th character	7th character																							
SD645				1st character of extension	2EH(.)																							
SD646	3rd character of extension	2nd character of the extension																										
SD642																												
SD643																												
SD644																												
SD645																												
SD646																												
SD646	File register file name	File register file name	<ul style="list-style-type: none"> <li>Stores file register file name (MAIN.QDR) selected at parameters as ASCII code.</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD641</td> <td>2nd character (A)</td> <td>1st character (M)</td> </tr> <tr> <td>SD642</td> <td>4th character (N)</td> <td>3rd character (I)</td> </tr> <tr> <td>SD643</td> <td>6th character ( )</td> <td>5th character ( )</td> </tr> <tr> <td>SD644</td> <td>8th character ( )</td> <td>7th character ( )</td> </tr> <tr> <td>SD645</td> <td>1st character of the extension (Q)</td> <td>2EH(.)</td> </tr> <tr> <td>SD646</td> <td>3rd character of the extension (R)</td> <td>2nd character of the extension (D)</td> </tr> </table>		b15 to b8	b7 to b0	SD641	2nd character (A)	1st character (M)	SD642	4th character (N)	3rd character (I)	SD643	6th character ( )	5th character ( )	SD644	8th character ( )	7th character ( )	SD645	1st character of the extension (Q)	2EH(.)	SD646	3rd character of the extension (R)	2nd character of the extension (D)	S (Initial)	New	Q00J/Q00/Q01	
				b15 to b8	b7 to b0																							
SD641				2nd character (A)	1st character (M)																							
SD642				4th character (N)	3rd character (I)																							
SD643				6th character ( )	5th character ( )																							
SD644				8th character ( )	7th character ( )																							
SD645				1st character of the extension (Q)	2EH(.)																							
SD646	3rd character of the extension (R)	2nd character of the extension (D)																										
SD647	File register capacity	File register capacity	<ul style="list-style-type: none"> <li>Stores the data capacity of the currently selected file register in 1 k word units.</li> </ul>	S (Status change)	New	Qn(H) QnPH QnPRH QnU <sup>*3</sup>																						
				S (Initial)			Q00J/Q00/Q01																					
SD648	File register block number	File register block number	<ul style="list-style-type: none"> <li>Stores the currently selected file register block number.</li> </ul>	S (Status change) *10	D9035	Q00J/Q00/Q01 Qn(H) QnPH QnPRH QnU <sup>*3</sup>																						
SD650	Comment drive	Comment drive number	<ul style="list-style-type: none"> <li>Stores the comment drive number selected at the parameters or by the QCDSET instruction.</li> </ul>	S (Status change)	New																							
SD651	Comment file name	Comment file name	<ul style="list-style-type: none"> <li>Stores the comment file name (with extension) selected at the parameters or by the QCDSET instruction in ASCII code.</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD651</td> <td>2nd character</td> <td>1st character</td> </tr> <tr> <td>SD652</td> <td>4th character</td> <td>3rd character</td> </tr> <tr> <td>SD653</td> <td>6th character</td> <td>5th character</td> </tr> <tr> <td>SD654</td> <td>8th character</td> <td>7th character</td> </tr> <tr> <td>SD655</td> <td>1st character of the extension</td> <td>2EH(.)</td> </tr> <tr> <td>SD656</td> <td>3rd character of the extension</td> <td>2nd character of the extension</td> </tr> </table>		b15 to b8	b7 to b0	SD651	2nd character	1st character	SD652	4th character	3rd character	SD653	6th character	5th character	SD654	8th character	7th character	SD655	1st character of the extension	2EH(.)	SD656	3rd character of the extension	2nd character of the extension	S (Status change)	New	Qn(H) QnPH QnPRH QnU	
				b15 to b8	b7 to b0																							
SD651				2nd character	1st character																							
SD652				4th character	3rd character																							
SD653				6th character	5th character																							
SD654				8th character	7th character																							
SD655	1st character of the extension	2EH(.)																										
SD656	3rd character of the extension	2nd character of the extension																										
SD652																												
SD653																												
SD654																												
SD655																												
SD656																												
SD660	Boot operation designation file	Boot designation file drive number	<ul style="list-style-type: none"> <li>Stores the drive number where the boot designation file (*.QBT) is being stored.</li> </ul>	S (Initial)	New																							
SD661		File name of boot designation file	File name of boot designation file	<ul style="list-style-type: none"> <li>Stores the file name of the boot designation file (*.QBT).</li> </ul> <table border="1"> <tr> <td></td> <td>b15 to b8</td> <td>b7 to b0</td> </tr> <tr> <td>SD661</td> <td>2nd character</td> <td>1st character</td> </tr> <tr> <td>SD662</td> <td>4th character</td> <td>3rd character</td> </tr> <tr> <td>SD663</td> <td>6th character</td> <td>5th character</td> </tr> <tr> <td>SD664</td> <td>8th character</td> <td>7th character</td> </tr> <tr> <td>SD665</td> <td>1st character of the extension</td> <td>2EH(.)</td> </tr> <tr> <td>SD666</td> <td>3rd character of the extension</td> <td>2nd character of the extension</td> </tr> </table>		b15 to b8	b7 to b0	SD661	2nd character	1st character	SD662	4th character	3rd character	SD663	6th character	5th character	SD664	8th character	7th character	SD665	1st character of the extension	2EH(.)	SD666	3rd character of the extension	2nd character of the extension	S (Initial)	New	Qn(H) QnPH QnPRH QnU <sup>*4</sup>
					b15 to b8	b7 to b0																						
SD661					2nd character	1st character																						
SD662					4th character	3rd character																						
SD663					6th character	5th character																						
SD664					8th character	7th character																						
SD665	1st character of the extension	2EH(.)																										
SD666	3rd character of the extension	2nd character of the extension																										
SD662																												
SD663																												
SD664																												
SD665																												
SD666																												

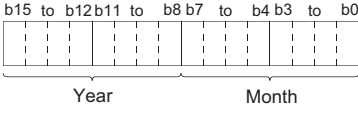
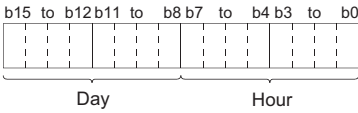
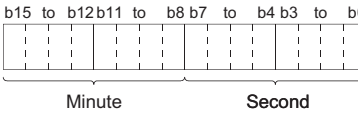
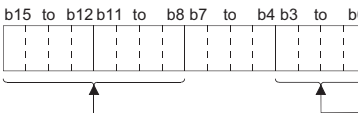
\*3: The Universal model QCPU except the Q00UJCPU.

\*4: The Universal model QCPU except the Q00UJCPU, Q00UCPU, and Q01UCPU.

\*10: On the Basic model QCPU, data is set at STOP to RUN or RSET instruction execution after parameter execution.

TableApp.4.7 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																		
SD670	Parameter enable drive information	Parameter enable drive No.	<ul style="list-style-type: none"> <li>Stores information of parameter storage destination drive which is enabled.                             <ul style="list-style-type: none"> <li>0: Drive 0 (Program memory)</li> <li>1: Drive 1 (SRAM card)</li> <li>2: Drive 2 (Flash card/ATA card)</li> <li>4: Drive 4 (Standard ROM)</li> </ul> </li> <li>(Only drive 0 and drive 4 are valid in the Q00UJCPU, Q00UCPU, and Q01UCPU.)</li> </ul>	S (Initial)	New																			
SD671	Status of latch data backup function	Status display	<p>Indicates the status of the latch data backup function.</p> <table border="1"> <thead> <tr> <th>Status</th> <th>Presence/absence of backup data</th> <th>Restore operation at turning power supply ON from OFF</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>No backup data</td> <td>Restoring not executed</td> </tr> <tr> <td>1</td> <td>Restore ready completion</td> <td>Restoring executed when turning power supply ON from OFF the following time</td> </tr> <tr> <td>2</td> <td>Restore execution completion</td> <td>Restoring not executed</td> </tr> <tr> <td>3</td> <td>Backup execution wait</td> <td>Restoring not executed</td> </tr> <tr> <td>4</td> <td>Restore repeated execution ready completion</td> <td>Restoring executed when turning power supply ON from OFF</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>"2 Restore ready completion" is a status immediately after restoring data.</li> <li>"3 Backup execution wait" is a status after turning power supply ON from OFF at "2 Restore ready completion".</li> </ul>	Status	Presence/absence of backup data	Restore operation at turning power supply ON from OFF	0	No backup data	Restoring not executed	1	Restore ready completion	Restoring executed when turning power supply ON from OFF the following time	2	Restore execution completion	Restoring not executed	3	Backup execution wait	Restoring not executed	4	Restore repeated execution ready completion	Restoring executed when turning power supply ON from OFF	S (Status change)	New	
Status	Presence/absence of backup data	Restore operation at turning power supply ON from OFF																						
0	No backup data	Restoring not executed																						
1	Restore ready completion	Restoring executed when turning power supply ON from OFF the following time																						
2	Restore execution completion	Restoring not executed																						
3	Backup execution wait	Restoring not executed																						
4	Restore repeated execution ready completion	Restoring executed when turning power supply ON from OFF																						
SD672	Backup information	Backup time (Year and month)	<ul style="list-style-type: none"> <li>Stores the last 2 digits of year and month when backup is performed in 2-digit BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:                      July, 1993                      9307H</p> <p>Year                      Month</p>	S (At write)	New	QnU																		
SD673		Backup time (Day and hour)	<ul style="list-style-type: none"> <li>Stores the day and hour when backup is performed in 2-digit BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:                      31st, 10 a.m.                      3110H</p> <p>Day                              Hour</p>																					
SD674		Backup time (Minute and second)	<ul style="list-style-type: none"> <li>Stores the minute and second when backup is performed in 2-digit BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:                      35 min., 48 sec.                      3548H</p> <p>Minute                              Second</p>																					
SD675		Backup time (Year and day of week)	<ul style="list-style-type: none"> <li>Stores the first 2 digits of year and day of week when backup is performed in BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:                      1993, Friday                      1905H</p> <p>Higher digits of year (0 to 99)</p> <table border="1"> <thead> <tr> <th>Day of the week</th> </tr> </thead> <tbody> <tr><td>0</td><td>Sunday</td></tr> <tr><td>1</td><td>Monday</td></tr> <tr><td>2</td><td>Tuesday</td></tr> <tr><td>3</td><td>Wednesday</td></tr> <tr><td>4</td><td>Thursday</td></tr> <tr><td>5</td><td>Friday</td></tr> <tr><td>6</td><td>Saturday</td></tr> </tbody> </table>				Day of the week	0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday	5	Friday	6	Saturday			
Day of the week																								
0	Sunday																							
1	Monday																							
2	Tuesday																							
3	Wednesday																							
4	Thursday																							
5	Friday																							
6	Saturday																							

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU											
SD676	Backup data restration information	Restore time (Year and month)	<ul style="list-style-type: none"> <li>Stores the last 2 digits of year and month when data is restored in 2-digit BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:   </p>	S (Initial)	New	QnU											
SD677		Restore time (Day and time)	<ul style="list-style-type: none"> <li>Stores the day and time when data is restored in 2-digit BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:   </p>														
SD678		Restore time (Minute and second)	<ul style="list-style-type: none"> <li>Stores the minute and second when data is restored in 2-digit BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:   </p>														
SD679		Restore time (Year and day of week)	<ul style="list-style-type: none"> <li>Stores the first 2 digits of year and day of week when data is restored in BCD code.</li> </ul> <p>b15 to b12 b11 to b8 b7 to b4 b3 to b0 Example:     Higher digits of year (0 to 99)   <table border="1" data-bbox="885 918 1005 1097"> <thead> <tr> <th>Day of the week</th> </tr> </thead> <tbody> <tr><td>0</td><td>Sunday</td></tr> <tr><td>1</td><td>Monday</td></tr> <tr><td>2</td><td>Tuesday</td></tr> <tr><td>3</td><td>Wednesday</td></tr> <tr><td>4</td><td>Thursday</td></tr> <tr><td>5</td><td>Friday</td></tr> <tr><td>6</td><td>Saturday</td></tr> </tbody> </table> </p>				Day of the week	0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday
Day of the week																	
0	Sunday																
1	Monday																
2	Tuesday																
3	Wednesday																
4	Thursday																
5	Friday																
6	Saturday																
SD681	Program memory write (transfer) status	Write (transfer) status display (percentage)	Displays the status of writing (transferring) the program memory (flash ROM) in percentage. (0 to 100%) "0" is set when the write direction is set.	S (At write)	New												
SD682	Program memory write count index	Write count index up to present	<ul style="list-style-type: none"> <li>Stores the index value for the number of write operations to the program memory (flash ROM) up to the present in BIN 32-bit value. When the index value exceeds 100 thousand times, "FLASH ROM ERROR" (error code: 1610) occurs. (The index value is calculated even when exceeding 100 thousand times.)</li> </ul> <p>Note) The write count does not equal to the index value. (Since a flash ROM write life is prolonged by the system, 1 is added to the write count index when writing is performed twice or so.)</p>	S (At write)	New												
SD683																	
SD686	Standard ROM write (transfer) status	Write (transfer) status display (per- centage)	Displays the status of writing (transferring) the standard ROM (flash ROM) in percentage. (0 to 100%) "0" is set when the write direction is set.	S (At write)	New												
SD687	Standard ROM write count index	Write count index up to present	<ul style="list-style-type: none"> <li>Stores the index value for the number of write operations to the standard ROM (flash ROM) up to the present in BIN 32-bit value. When the index value exceeds 100 thousand times, "FLASH ROM ERROR" (error code: 1610) occurs. (The index value is calculated even when exceeding 100 thousand times.)</li> </ul> <p>Note) The write count does not equal to the index value. (Since a flash ROM write life is prolonged by the system, 1 is added to the write count index when the total write capacity after the previous count up reaches about 1M byte.)</p>	S (At write)	New												
SD688																	
SD689	Backup error factor	Backup error factor	Stores the factor of the error that occurred in the backup. 0H : No error 100H: Memory card not inserted 200H: Size of backup target data exceeded 300H: Memory card write inhibit setting 400H: Memory card write error 500H: Backup target data read error (from program memory) 503H: Backup target data read error (from standard RAM) 504H: Backup target data read error (from standard ROM) 510H: Backup target data read error (from system data)	S (Backup error occurrence)	New	QnU <sup>*1</sup>											
SD690	Backup status	Backup status	Stores the current backup status. 0 : Before backup start 1 : Backup start prepared 2 : Backup start preparation completed 3 : Backup in execution 4 : Backup completed FF: Backup error	S (Status change)	New												

\*1: The module whose first 5 digits of serial No. is "10102" or later. (Except the Q00UJCPU, Q00UCPU, and Q01UCPU)

A

Appendix 4 SPECIAL REGISTER LIST

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD691	Backup execution status	Backup execution status display (Percentage)	<ul style="list-style-type: none"> <li>Displays the execution status of data backup to the memory card in percentage (0 to 100%).</li> <li>"0" is set when the backup starts.</li> </ul>	S (Status change)	New	QnU <sup>*1</sup>
SD692	Restoration error factor	Factor of error occurred in the restoration	Stores the factor of an error that occurred in the restoration. Each error factor is as follows: 800H: The CPU module model name is not matched. 801H: The file password is set only for the restoration destination data or is not matched. 810H: The verified backup data file is not matched or the backup data read failed.	S (Error occurrence)	New	
SD693	Restoration status	Current restoration status	Stores the current restoration execution status. Each error factor is as follows: 0 : Before restoration start 1 : Restoration in execution 2 : Restoration completed FF: Restoration error Sets "0" (Before restoring), however, when the restoration is completed only during the automatic restoration.	S (Status change)	New	
SD694	Restoration execution status	Restoration execution status display (Percentage)	<ul style="list-style-type: none"> <li>Displays the execution status of restoration to the CPU module in percentage (0 to 100%).</li> <li>"0" is set before the restoration.</li> <li>Sets "0" (Before restoring), however, when the restoration is completed only during the automatic restoration.</li> </ul>	S (Status change)	New	
SD695	Specification of writing to standard ROM instruction count	Specification of writing to standard ROM instruction count	<ul style="list-style-type: none"> <li>Specifies the maximum number of executions of the writing to standard ROM instruction (SP.DEVST) to write to the standard ROM per day.</li> <li>When the number of executions of the writing to standard ROM instruction exceeds the number of times set by SD695, "OPERATION ERROR" (error code: 4113) occurs.</li> <li>The setting range for SD695 is 1 to 32767. If 0 or value outside the range is set, "OPERATION ERROR" (error code: 4113) occurs at execution of the writing to standard ROM instruction.</li> </ul>	U	New	QnU
SD696	Available memory in memory card	Available memory in memory card	Stores the available memory in memory card. (Stores the value in 32-bit binary.)	S (Backup in operation)	New	QnU <sup>*1</sup>
SD697						
SD698	Backup data capacity	Backup data capacity	Stores the backup data capacity. (Stores the value in 32-bit binary.)			
SD699						

\*1: The module whose first 5 digits of serial No. is "10102" or later. (Except the Q00UJCPU, Q00UCPU, and Q01UCPU)

(6) Instruction-Related Registers

TableApp.4.8 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																																																																																																																																																																																																																																				
SD705	Mask pattern	Mask pattern	<ul style="list-style-type: none"> <li>During block operations, turning SM705 ON makes it possible to use the mask pattern being stored at SD705 (or at SD705 and SD706 if double words are being used) to operate on all data in the block with the masked values.</li> </ul>	U	New	Q00J/Q00/Q01 Qn(H) QnPH QnPRH																																																																																																																																																																																																																																				
SD706																																																																																																																																																																																																																																										
SD715	IMASK instruction mask pattern	Mask pattern	<ul style="list-style-type: none"> <li>Patterns masked by use of the IMASK instruction are stored in the following manner:</li> </ul> <table border="1" style="margin-left: 40px;"> <tr> <td></td> <td style="text-align: center;">b15</td> <td></td> <td style="text-align: center;">b1</td> <td style="text-align: center;">b0</td> </tr> <tr> <td>SD715</td> <td style="text-align: center;">I15</td> <td style="text-align: center;">to</td> <td style="text-align: center;">I1</td> <td style="text-align: center;">I0</td> </tr> <tr> <td>SD716</td> <td style="text-align: center;">I31</td> <td style="text-align: center;">to</td> <td style="text-align: center;">I17</td> <td style="text-align: center;">I16</td> </tr> <tr> <td>SD717</td> <td style="text-align: center;">I47</td> <td style="text-align: center;">to</td> <td style="text-align: center;">I33</td> <td style="text-align: center;">I32</td> </tr> </table>		b15		b1	b0	SD715	I15	to	I1	I0	SD716	I31	to	I17	I16	SD717	I47	to	I33	I32	S (During execution)	New	QCPU																																																																																																																																																																																																																
				b15		b1	b0																																																																																																																																																																																																																																			
SD715				I15	to	I1	I0																																																																																																																																																																																																																																			
SD716	I31	to	I17	I16																																																																																																																																																																																																																																						
SD717	I47	to	I33	I32																																																																																																																																																																																																																																						
SD716																																																																																																																																																																																																																																										
SD717																																																																																																																																																																																																																																										
SD718	Accumulator	Accumulator	<ul style="list-style-type: none"> <li>For use as replacement for accumulators used in A series programs.</li> </ul>	S/U	New																																																																																																																																																																																																																																					
SD719																																																																																																																																																																																																																																										
SD720	Program No. designation for PLOADP instruction	Program No. designation for PLOADP instruction	Stores the program number of the program to be loaded by the PLOADP instruction when designated. Designation range: 1 to 124	U	New	Qn(H) QnPH																																																																																																																																																																																																																																				
SD738	Message storage	Message storage	<ul style="list-style-type: none"> <li>Stores the message designated by the MSG instruction.</li> </ul> <table border="1" style="margin-left: 40px;"> <tr> <td></td> <td style="text-align: center;">b15</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b8</td> <td style="text-align: center;">b7</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b0</td> </tr> <tr> <td>SD738</td> <td style="text-align: center;">2nd character</td> <td></td> <td style="text-align: center;">1st character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD739</td> <td style="text-align: center;">4th character</td> <td></td> <td style="text-align: center;">3rd character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD740</td> <td style="text-align: center;">6th character</td> <td></td> <td style="text-align: center;">5th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD741</td> <td style="text-align: center;">8th character</td> <td></td> <td style="text-align: center;">7th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD742</td> <td style="text-align: center;">10th character</td> <td></td> <td style="text-align: center;">9th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD743</td> <td style="text-align: center;">12th character</td> <td></td> <td style="text-align: center;">11th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD744</td> <td style="text-align: center;">14th character</td> <td></td> <td style="text-align: center;">13th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD745</td> <td style="text-align: center;">16th character</td> <td></td> <td style="text-align: center;">15th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD746</td> <td style="text-align: center;">18th character</td> <td></td> <td style="text-align: center;">17th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD747</td> <td style="text-align: center;">20th character</td> <td></td> <td style="text-align: center;">19th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD748</td> <td style="text-align: center;">22nd character</td> <td></td> <td style="text-align: center;">21st character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD749</td> <td style="text-align: center;">24th character</td> <td></td> <td style="text-align: center;">23rd character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD750</td> <td style="text-align: center;">26th character</td> <td></td> <td style="text-align: center;">25th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD751</td> <td style="text-align: center;">28th character</td> <td></td> <td style="text-align: center;">27th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD752</td> <td style="text-align: center;">30th character</td> <td></td> <td style="text-align: center;">29th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD753</td> <td style="text-align: center;">32nd character</td> <td></td> <td style="text-align: center;">31st character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD754</td> <td style="text-align: center;">34th character</td> <td></td> <td style="text-align: center;">33rd character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD755</td> <td style="text-align: center;">36th character</td> <td></td> <td style="text-align: center;">35th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD756</td> <td style="text-align: center;">38th character</td> <td></td> <td style="text-align: center;">37th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD757</td> <td style="text-align: center;">40th character</td> <td></td> <td style="text-align: center;">39th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD758</td> <td style="text-align: center;">42nd character</td> <td></td> <td style="text-align: center;">41st character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD759</td> <td style="text-align: center;">44th character</td> <td></td> <td style="text-align: center;">43rd character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD760</td> <td style="text-align: center;">46th character</td> <td></td> <td style="text-align: center;">45th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD761</td> <td style="text-align: center;">48th character</td> <td></td> <td style="text-align: center;">47th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD762</td> <td style="text-align: center;">50th character</td> <td></td> <td style="text-align: center;">49th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD763</td> <td style="text-align: center;">52nd character</td> <td></td> <td style="text-align: center;">51st character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD764</td> <td style="text-align: center;">54th character</td> <td></td> <td style="text-align: center;">53rd character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD765</td> <td style="text-align: center;">56th character</td> <td></td> <td style="text-align: center;">55th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD766</td> <td style="text-align: center;">58th character</td> <td></td> <td style="text-align: center;">57th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD767</td> <td style="text-align: center;">60th character</td> <td></td> <td style="text-align: center;">59th character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD768</td> <td style="text-align: center;">62nd character</td> <td></td> <td style="text-align: center;">61st character</td> <td></td> <td></td> <td></td> </tr> <tr> <td>SD769</td> <td style="text-align: center;">64th character</td> <td></td> <td style="text-align: center;">63rd character</td> <td></td> <td></td> <td></td> </tr> </table>		b15	to	b8	b7	to	b0	SD738	2nd character		1st character				SD739	4th character		3rd character				SD740	6th character		5th character				SD741	8th character		7th character				SD742	10th character		9th character				SD743	12th character		11th character				SD744	14th character		13th character				SD745	16th character		15th character				SD746	18th character		17th character				SD747	20th character		19th character				SD748	22nd character		21st character				SD749	24th character		23rd character				SD750	26th character		25th character				SD751	28th character		27th character				SD752	30th character		29th character				SD753	32nd character		31st character				SD754	34th character		33rd character				SD755	36th character		35th character				SD756	38th character		37th character				SD757	40th character		39th character				SD758	42nd character		41st character				SD759	44th character		43rd character				SD760	46th character		45th character				SD761	48th character		47th character				SD762	50th character		49th character				SD763	52nd character		51st character				SD764	54th character		53rd character				SD765	56th character		55th character				SD766	58th character		57th character				SD767	60th character		59th character				SD768	62nd character		61st character				SD769	64th character		63rd character			
				b15	to	b8	b7	to	b0																																																																																																																																																																																																																																	
SD738				2nd character		1st character																																																																																																																																																																																																																																				
SD739				4th character		3rd character																																																																																																																																																																																																																																				
SD740				6th character		5th character																																																																																																																																																																																																																																				
SD741				8th character		7th character																																																																																																																																																																																																																																				
SD742				10th character		9th character																																																																																																																																																																																																																																				
SD743				12th character		11th character																																																																																																																																																																																																																																				
SD744				14th character		13th character																																																																																																																																																																																																																																				
SD745				16th character		15th character																																																																																																																																																																																																																																				
SD746				18th character		17th character																																																																																																																																																																																																																																				
SD747				20th character		19th character																																																																																																																																																																																																																																				
SD748				22nd character		21st character																																																																																																																																																																																																																																				
SD749				24th character		23rd character																																																																																																																																																																																																																																				
SD750				26th character		25th character																																																																																																																																																																																																																																				
SD751				28th character		27th character																																																																																																																																																																																																																																				
SD752				30th character		29th character																																																																																																																																																																																																																																				
SD753				32nd character		31st character																																																																																																																																																																																																																																				
SD754				34th character		33rd character																																																																																																																																																																																																																																				
SD755				36th character		35th character																																																																																																																																																																																																																																				
SD756				38th character		37th character																																																																																																																																																																																																																																				
SD757				40th character		39th character																																																																																																																																																																																																																																				
SD758				42nd character		41st character																																																																																																																																																																																																																																				
SD759				44th character		43rd character																																																																																																																																																																																																																																				
SD760				46th character		45th character																																																																																																																																																																																																																																				
SD761				48th character		47th character																																																																																																																																																																																																																																				
SD762				50th character		49th character																																																																																																																																																																																																																																				
SD763				52nd character		51st character																																																																																																																																																																																																																																				
SD764				54th character		53rd character																																																																																																																																																																																																																																				
SD765				56th character		55th character																																																																																																																																																																																																																																				
SD766				58th character		57th character																																																																																																																																																																																																																																				
SD767				60th character		59th character																																																																																																																																																																																																																																				
SD768	62nd character		61st character																																																																																																																																																																																																																																							
SD769	64th character		63rd character																																																																																																																																																																																																																																							

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.8 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU	Corresponding CPU																				
SD774 to SD775	PID limit setting (for complete derivative)	0: With limit 1: Without limit	<ul style="list-style-type: none"> <li>Specify the limit of each PID loop as shown below.</li> </ul> <table border="1"> <tr> <td>SD774</td> <td>Loop16</td> <td>to</td> <td>Loop2</td> <td>Loop1</td> </tr> <tr> <td>SD775</td> <td>Loop32</td> <td>to</td> <td>Loop18</td> <td>Loop17</td> </tr> </table>	SD774	Loop16	to	Loop2	Loop1	SD775	Loop32	to	Loop18	Loop17	U	New	Qn(H) QnPRH QnU										
SD774	Loop16	to	Loop2	Loop1																						
SD775	Loop32	to	Loop18	Loop17																						
SD774	PID limit setting (for complete derivative)	0: With limit 1: Without limit	<ul style="list-style-type: none"> <li>Specify the limit of each PID loop as shown below.</li> </ul> <table border="1"> <tr> <td>SD774</td> <td>b15 to b8</td> <td>to</td> <td>b7</td> <td>to</td> <td>b1</td> <td>b0</td> </tr> <tr> <td></td> <td></td> <td></td> <td>Loop8</td> <td></td> <td>Loop2</td> <td>Loop1</td> </tr> </table>	SD774	b15 to b8	to	b7	to	b1	b0				Loop8		Loop2	Loop1	U	New	Q00J/Q00/Q01 <sup>*9</sup>						
SD774	b15 to b8	to	b7	to	b1	b0																				
			Loop8		Loop2	Loop1																				
SD778	Refresh processing selection when the COM/CCOM instruction is executed	b0 to b14: 0: Do not refresh 1: Refresh b15 bit 0: Communication with CPU module is executed 1: Communication with CPU module is nonexecuted	<ul style="list-style-type: none"> <li>Selects whether or not the data is refreshed when the COM instruction is executed.</li> <li>Designation of SD778 is made valid when SM775 turns ON.</li> </ul> <table border="1"> <tr> <td>SD778</td> <td>b15</td> <td>b14</td> <td>to</td> <td>b5</td> <td>b4</td> <td>b3</td> <td>b2</td> <td>b1</td> <td>b0</td> </tr> <tr> <td></td> <td>0/1</td> <td></td> <td></td> <td>0</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> </tr> </table> <ul style="list-style-type: none"> <li>I/O refresh</li> <li>CC-Link refresh</li> <li>MELSECNET/H refresh</li> <li>Automatic refresh of intelligent function modules</li> <li>Automatic refresh of CPU shared memory (Fixed to "0" for Redundant CPU)</li> <li>Execution/non-execution of communication with CPU module</li> </ul> <ul style="list-style-type: none"> <li>Refresh between multiple CPUs by COM instruction is performed under the following occasion.                              Receiving operation from other device: b4 of SD778(refresh in the CPU shared memory) is turned to 1.                              Sending operation from host CPU : b15 of SD778(communication with peripheral device is executed/nonexecuted) is turned to 0.</li> </ul>	SD778	b15	b14	to	b5	b4	b3	b2	b1	b0		0/1			0	0/1	0/1	0/1	0/1	0/1	U	New	Q00J/Q00/Q01 <sup>*9</sup> Qn(H) <sup>*11</sup>
			SD778	b15	b14	to	b5	b4	b3	b2	b1	b0														
	0/1			0	0/1	0/1	0/1	0/1	0/1																	
<ul style="list-style-type: none"> <li>Selects whether or not the data is refreshed when the COM instruction is executed.</li> <li>Designation of SD778 is made valid when SM775 turns ON.</li> </ul> <table border="1"> <tr> <td>SD778</td> <td>b15</td> <td>b14</td> <td>to</td> <td>b6</td> <td>b5</td> <td>b4</td> <td>b3</td> <td>b2</td> <td>b1</td> <td>b0</td> </tr> <tr> <td></td> <td>0/1</td> <td></td> <td></td> <td>0</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> <td>0/1</td> </tr> </table> <ul style="list-style-type: none"> <li>I/O refresh</li> <li>CC-Link refresh</li> <li>CC-Link IE controller network or MELSECNET/H refresh</li> <li>Automatic refresh of intelligent function modules</li> <li>Reading input/output from group outside multiple CPU system</li> <li>Auto refresh using the multiple CPU high speed transmission area of multiple CPU system</li> <li>Execution/non-execution of communication with CPU module</li> </ul> <ul style="list-style-type: none"> <li>Refresh between multiple CPUs by COM instruction is performed under the following occasion.                              Receiving operation from other device: b4 of SD778(refresh in the CPU shared memory) is turned to 1.                              Sending operation from host CPU : b15 of SD778(communication with peripheral device is executed/nonexecuted) is turned to 0.</li> <li>When b2 (refresh of the CC-Link IE controller network and MELSECNET/H) of SD778 is 1, the CC-Link IE controller network and MELSECNET/H perform refresh.                              Therefore, if there are many refresh points, processing time for the COM instruction will be extended.</li> </ul>	SD778	b15	b14	to	b6	b5	b4	b3	b2	b1	b0		0/1			0	0/1	0/1	0/1	0/1	0/1	0/1	U	New	Qn(H) <sup>*13</sup> QnPH <sup>*12</sup> QnPRH	
SD778	b15	b14	to	b6	b5	b4	b3	b2	b1	b0																
	0/1			0	0/1	0/1	0/1	0/1	0/1	0/1																

\*9: Function version is B or later.

\*11: The module whose first 5 digits of serial No. is "04012" or later.

\*12: The module whose first 5 digits of serial No. is "07032" or later.

\*13: The module whose first 5 digits of serial No. is "09012" or later.



TableApp.4.8 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU																									
SD778	Refresh processing selection when the COM/CCOM instruction is executed	b0 to b14: 0: Do not refresh 1: Refresh b15 bit 0: communication with peripheral device is executed 1: communication with peripheral device is nonexecuted	<ul style="list-style-type: none"> <li>Selects whether or not the data is refreshed when the COM, CCOM instruction is executed.</li> <li>Designation of SD778 is made valid when SM775 turns ON.</li> </ul>	U	New	QnU																									
SD781 to SD793	Mask pattern of IMASK instruction	Mask pattern	<ul style="list-style-type: none"> <li>Stores the mask patterns masked by the IMASK instruction as follows:</li> </ul> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td></td> <td style="text-align: center;">b15</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b1</td> <td style="text-align: center;">b0</td> </tr> <tr> <td>SD781</td> <td style="text-align: center;">I63</td> <td></td> <td style="text-align: center;">I49</td> <td style="text-align: center;">I48</td> </tr> <tr> <td>SD782</td> <td style="text-align: center;">I79</td> <td></td> <td style="text-align: center;">I65</td> <td style="text-align: center;">I64</td> </tr> <tr> <td></td> <td colspan="4" style="text-align: center;">to</td> </tr> <tr> <td>SD793</td> <td style="text-align: center;">I255</td> <td></td> <td style="text-align: center;">I241</td> <td style="text-align: center;">I240</td> </tr> </table> <p>(The Q00UJCPU, Q00UCPU, and Q01UCPU cannot use the special registers SD786 to SD793.)</p>		b15	to	b1	b0	SD781	I63		I49	I48	SD782	I79		I65	I64		to				SD793	I255		I241	I240	S (During execution)	New	Qn(H) QnPH QnPRH QnU
	b15	to	b1	b0																											
SD781	I63		I49	I48																											
SD782	I79		I65	I64																											
	to																														
SD793	I255		I241	I240																											
SD781 to SD785	Mask pattern of IMASK instruction	Mask pattern	<ul style="list-style-type: none"> <li>Stores the mask patterns masked by the IMASK instruction as follows:</li> </ul> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td></td> <td style="text-align: center;">b15</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b1</td> <td style="text-align: center;">b0</td> </tr> <tr> <td>SD781</td> <td style="text-align: center;">I63</td> <td></td> <td style="text-align: center;">I49</td> <td style="text-align: center;">I48</td> </tr> <tr> <td>SD782</td> <td style="text-align: center;">I79</td> <td></td> <td style="text-align: center;">I65</td> <td style="text-align: center;">I64</td> </tr> <tr> <td></td> <td colspan="4" style="text-align: center;">to</td> </tr> <tr> <td>SD785</td> <td style="text-align: center;">I127</td> <td></td> <td style="text-align: center;">I113</td> <td style="text-align: center;">I112</td> </tr> </table>		b15	to	b1	b0	SD781	I63		I49	I48	SD782	I79		I65	I64		to				SD785	I127		I113	I112	S (During execution)	New	Q00J/Q00/Q01
	b15	to	b1	b0																											
SD781	I63		I49	I48																											
SD782	I79		I65	I64																											
	to																														
SD785	I127		I113	I112																											
SD794 to SD795	PID limit setting (for incomplete derivative)	0: With limit 1: Without limit	<ul style="list-style-type: none"> <li>Specify the limit of each PID loop as shown below.</li> </ul> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td></td> <td style="text-align: center;">b15</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b1</td> <td style="text-align: center;">b0</td> </tr> <tr> <td>SD794</td> <td style="text-align: center;">Loop16</td> <td></td> <td style="text-align: center;">Loop2</td> <td style="text-align: center;">Loop1</td> </tr> <tr> <td>SD795</td> <td style="text-align: center;">Loop32</td> <td></td> <td style="text-align: center;">Loop18</td> <td style="text-align: center;">Loop17</td> </tr> </table>		b15	to	b1	b0	SD794	Loop16		Loop2	Loop1	SD795	Loop32		Loop18	Loop17	U	New	Qn(H) <sup>*13</sup> QnPRH QnU										
	b15	to	b1	b0																											
SD794	Loop16		Loop2	Loop1																											
SD795	Loop32		Loop18	Loop17																											
SD794	PID limit setting (for incomplete derivative)	0: With limit 1: Without limit	<ul style="list-style-type: none"> <li>Specify the limit of each PID loop as shown below.</li> </ul> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td></td> <td style="text-align: center;">b15</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b8</td> <td style="text-align: center;">b7</td> <td style="text-align: center;">to</td> <td style="text-align: center;">b1</td> <td style="text-align: center;">b0</td> </tr> <tr> <td>SD794</td> <td style="text-align: center;">Loop8</td> <td></td> <td style="text-align: center;">Loop8</td> <td></td> <td></td> <td style="text-align: center;">Loop2</td> <td style="text-align: center;">Loop1</td> </tr> </table>		b15	to	b8	b7	to	b1	b0	SD794	Loop8		Loop8			Loop2	Loop1	U	New	Q00J/Q00/Q01 <sup>*9</sup>									
	b15	to	b8	b7	to	b1	b0																								
SD794	Loop8		Loop8			Loop2	Loop1																								

\*9: Function version is B or later.

\*13: The module whose first 5 digits of serial No. is "09012" or later.

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.8 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD796	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.1)	Maximum number of blocks range for dedicated instructions Range: 1 to 7 (Default: 2 Or when setting other than 1 to 7, the register operates as 7).	<ul style="list-style-type: none"> <li>Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU=CPU No.1). When the dedicated instruction of Multiple CPU transmission is executed to the CPU No.1, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM796 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instruction of Multiple CPU transmission.</li> </ul>	U (At 1 scan after RUN)	New	QnU*14*15
SD797	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.2)		<ul style="list-style-type: none"> <li>Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU=CPU No.2). When the dedicated instruction of Multiple CPU transmission is executed to the CPU No.2, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM797 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instruction of Multiple CPU transmission.</li> </ul>	U (At 1 scan after RUN)	New	
SD798	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.3)		<ul style="list-style-type: none"> <li>Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU=CPU No.3). When the dedicated instruction of Multiple CPU transmission is executed to the CPU No.3, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM798 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instruction of Multiple CPU transmission.</li> </ul>	U (At 1 scan after RUN)	New	
SD799	Maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction setting (for CPU No.4)		<ul style="list-style-type: none"> <li>Specifies the maximum number of blocks used for the multiple CPU high-speed transmission dedicated instruction (target CPU=CPU No.4). When the dedicated instruction of Multiple CPU transmission is executed to the CPU No.4, and the number of empty blocks of the dedicated instruction transmission area is less than the setting value of this register, SM799 is turned ON, which is used as the interlock signal for consecutive execution of the dedicated instruction of Multiple CPU transmission.</li> </ul>	U (At 1 scan after RUN)	New	

\*14: The Universal model QCPU except the Q00UJCPU, Q00UCPU, Q01UCPU, and Q02UCPU.

\*15: The range is from 1 to 9 for the Q03UDCPU, Q04UDCPU, and Q06UDHCP whose first 5 digits of serial number is "10012" or earlier.  
(Default: 2 Or when setting other than 1 to 9, the register operates as 9).

## (7) Debug

TableApp.4.9 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD840	Debug function usage	Debug function usage	<p>Stores the status of the debug function usage as shown below.</p> <p>0: Forced ON/OFF for external I/O 1: Executional conditioned device test 2 to 15: Absent (0 fix)</p> <p>(0: Not used, 1: Used)</p>	S (Status change)	New	QnU*1

\*1: The module whose first 5 digits of serial No. is "10042" or later.

## (8) Redundant CPU information (host system CPU information\*1)

TableApp.4.10 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD952	History of memory copy from control system to standby system	Latest status of memory copy from control system to standby system	<p>Stores the completion status of the memory copy from control system to standby system executed last.</p> <p>1) Stores the same value as stored into SD1596 at normal completion/ abnormal completion of the memory copy from control system to standby system.</p> <p>2) Backed up for a power failure, this special register holds the status of memory copy from control system to standby system executed last.</p> <p>3) Cleared to 0 by latch clear operation.</p>	S (Status change)	New	QnPRH

\*1: The host system CPU information is stored.

## (9) Remote password count

TableApp.4.11 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD979	Direct MELSOFT connection	Count of unlock processing failures	Stores the count of unlock processing failures. Range: 0 to FFFEh (FFFFh when the limit is exceeded)	S (Status change)	New	QnU*1
SD980 to SD995	Connection 1 to 16					
SD998	MELSOFT connection using TCP port					
SD999	FTP communication port					

\*1: This applies to the Built-in Ethernet port QCPU.

A

Appendix 4 SPECIAL REGISTER LIST

(10) A to Q conversion

ACPU special registers D9000 to D9255 correspond to Q special registers SD1000 to SD1255 after A to Q/QnA conversion.

(However, the Basic model QCPU and Redundant CPU do not support the A to Q conversion.)

These special registers are all set by the system, and cannot be set by the user program. To set data by the user program, correct the program for use of the QCPU special registers. However, some of SD1200 to SD1255 (corresponding to D9200 to 9255 before conversion) can be set by the user program if they could be set by the user program before conversion. For details on the ACPUs special registers, refer to the user's manual for the corresponding CPU, and MELSECNET or MELSECNET/B Data Link System Reference Manuals.

---

**POINT**

Check "Use special relay/special register from SM/SD1000" for "A-PLC" on the PLC system tab of PLC parameter in GX Developer when the converted special registers are used with the High Performance model QCPU, Process CPU, and Universal model QCPU.

When not using the converted special registers, uncheck "Use special relay/special registers from SM/SD1000" to save the time taken for processing special registers.

---

**Remark**

Supplemental explanation on "Special Register for Modification" column

- ① For the device numbers for which a special register for modification is specified, modify it to the special register for QCPU.
- ② For the device numbers for which  is specified, special register after conversion can be used.
- ③ Device numbers for which  is specified do not function for QCPU.

TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																								
D9000	SD1000	—	Fuse blown	Number of module with blown fuse	<ul style="list-style-type: none"> <li>When fuse blown modules are detected, the first I/O number of the lowest number of the detected modules is stored in hexadecimal. (Example: When fuses of Y50 to 6F output modules have blown, "50" is stored in hexadecimal)</li> <li>To monitor the number by peripheral devices, perform monitor operation given in hexadecimal.</li> <li>(Cleared when all contents of SD1100 to SD1107 are reset to 0.)</li> <li>Fuse blow check is executed also to the output modules of remote I/O stations.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>																																								
D9001	SD1001	—	Fuse blown	Number of module with blown fuse	<ul style="list-style-type: none"> <li>Stores the module numbers corresponding to setting switch numbers or base slot numbers when fuse blow occurred.</li> </ul> <table border="1"> <thead> <tr> <th colspan="2">AJ02 I/O module</th> <th colspan="2">Extension base unit</th> </tr> <tr> <th>Setting switch</th> <th>Stored data</th> <th>Base unit slot No.</th> <th>Stored data</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>4</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>5</td> </tr> <tr> <td>2</td> <td>2</td> <td>2</td> <td>6</td> </tr> <tr> <td>3</td> <td>3</td> <td>3</td> <td>7</td> </tr> <tr> <td>4</td> <td>4</td> <td></td> <td></td> </tr> <tr> <td>5</td> <td>5</td> <td></td> <td></td> </tr> <tr> <td>6</td> <td>6</td> <td></td> <td></td> </tr> <tr> <td>7</td> <td>7</td> <td></td> <td></td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>For the remote I/O station, the value of (module I/O No./10H) + 1 is stored.</li> </ul>	AJ02 I/O module		Extension base unit		Setting switch	Stored data	Base unit slot No.	Stored data	0	0	0	4	1	1	1	5	2	2	2	6	3	3	3	7	4	4			5	5			6	6			7	7			Qn(H) QnPH
AJ02 I/O module		Extension base unit																																												
Setting switch	Stored data	Base unit slot No.	Stored data																																											
0	0	0	4																																											
1	1	1	5																																											
2	2	2	6																																											
3	3	3	7																																											
4	4																																													
5	5																																													
6	6																																													
7	7																																													
D9002	SD1002	—	I/O module verify error	I/O module verify error module number	<ul style="list-style-type: none"> <li>If I/O modules, of which data are different from data entered, are detected when the power is turned on, the first I/O number of the lowest number unit among the detected units is stored in hexadecimal. (Storing method is the same as that of SD1000.) To monitor the number by peripheral devices, perform monitor operation given in hexadecimal.</li> <li>(Cleared when all contents of SD1116 to SD1123 are reset to 0.)</li> <li>I/O module verify check is executed also to the modules of remote I/O terminals.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>																																								
D9005	SD1005	—	AC DOWN counter	Number of times for AC DOWN	<ul style="list-style-type: none"> <li>When the AC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 20ms. (The value is stored in BIN code.) It is reset when the power supply is switched from OFF to ON.</li> <li>When the DC power supply module is used, 1 is added at occurrence of an instantaneous power failure of within 10ms. (The value is stored in BIN code.) It is reset when the power supply is switched from OFF to ON.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup> Qn(H) QnPH QnU <sup>*1</sup>																																								
D9008	SD1008	SD0	Self-diagnostic error	Self-diagnostic error number	<ul style="list-style-type: none"> <li>When error is found as a result of self-diagnosis, error number is stored in BIN code.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>																																								

\*1: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9009	SD1009	SD62	Annunciator detection	F number at which external failure has occurred	<ul style="list-style-type: none"> <li>When one of F0 to 2047 is turned on by OUT F or SET F instruction, the F number, which has been detected earliest among the F numbers which have turned on, is stored in BIN code.</li> <li>SD1009 can be cleared by RST F or LEDR instruction.</li> <li>If another F number has been detected, the clearing of SD1009 causes the next number to be stored in SD1009.</li> </ul>	Qn(H) QnPH QnU*1
D9010	SD1010	x	Error step	Step number at which operation error has occurred.	<ul style="list-style-type: none"> <li>When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code.</li> <li>Thereafter, each time operation error occurs, the contents of SD1010 are renewed.</li> </ul>	Qn(H) QnPH
D9011	SD1011	x	Error step	Step number at which operation error has occurred.	<ul style="list-style-type: none"> <li>When operation error has occurred during execution of application instruction, the step number, at which the error has occurred, is stored in BIN code. Since the step number is stored into SD1011 when SM1011 turns from OFF to ON, the data of SD1011 is not updated unless SM1011 is cleared by a user program.</li> </ul>	
D9014	SD1014	x	I/O control mode	I/O control mode number	<ul style="list-style-type: none"> <li>The I/O control mode set is returned in any of the following numbers: 0: Both input and output in direct mode 1: Input in refresh mode, output in direct mode 3: Both input and output in refresh mode</li> </ul>	
D9015	SD1015	SD203	Operating status of CPU	Operating status of CPU	<ul style="list-style-type: none"> <li>The operation status of CPU as shown below are stored in SD1015.</li> </ul> <p>*1: When the CPU module is in RUN mode and SM1040 is off, the CPU module remains in RUN mode if changed to PAUSE mode.</p>	Qn(H) QnPH QnU*1

\*1: The relevant modules are as follows:  
 • The Universal model QCPU whose serial number (first five digits) is "10102" or later.  
 • Q00UJCPU, Q00UCPU, Q01UCPU

TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU								
D9016	SD1016	x	Program number	0: Main program (ROM) 1: Main program (RAM) 2: Subprogram 1 (RAM) 3: Subprogram 2 (RAM) 4: Subprogram 3 (RAM) 5: Subprogram 1 (ROM) 6: Subprogram 2 (ROM) 7: Subprogram 3 (ROM) 8: Main program (E <sup>2</sup> PROM) 9: Subprogram 1 (E <sup>2</sup> PROM) A: Subprogram 2 (E <sup>2</sup> PROM) B: Subprogram 3 (E <sup>2</sup> PROM)	<ul style="list-style-type: none"> <li>Indicates which sequence program is run presently. One value of 0 to B is stored in BIN code.</li> </ul>	Qn(H) QnPH								
D9017	SD1017	SD524	Scan time	Minimum scan time (10 ms units)	<ul style="list-style-type: none"> <li>If scan time is smaller than the content of SD1017, the value is newly stored at each END. Namely, the minimum value of scan time is stored into SD1017 in BIN code.</li> </ul>	Qn(H) QnPH QnU <sup>1</sup>								
D9018	SD1018	SD520	Scan time	Scan time (10 ms units)	<ul style="list-style-type: none"> <li>At every END, the scan time is stored in BIN code and always rewritten.</li> </ul>									
D9019	SD1019	SD526	Scan time	Maximum scan time (10 ms units)	<ul style="list-style-type: none"> <li>If scan time is larger than the content of SD1019, the value is newly stored at each END. Namely, the maximum value of scan time is stored into SD1019 in BIN code.</li> </ul>									
D9020	SD1020	x	Constant scan	Constant scan time (User sets in 10 ms units)	<ul style="list-style-type: none"> <li>Sets the interval between consecutive program starts in multiples of 10 ms.                          0 : No setting                          1 to 200 : Set. Program is executed at intervals of (set value) x 10 ms.</li> </ul>	Qn(H) QnPH								
D9021	SD1021	-	Scan time	Scan time (1 ms units)	<ul style="list-style-type: none"> <li>At every END, the scan time is stored in BIN code and always rewritten.</li> </ul>	Qn(H) QnPH QnU <sup>1</sup>								
D9022	SD1022	SD412	1 second counter	Count in units of 1s.	<ul style="list-style-type: none"> <li>When the PC CPU starts running, it starts counting 1 every second.</li> <li>It starts counting up from 0 to 32767, then down to -32768 and then again up to 0. Counting repeats this routine.</li> </ul>									
D9025	SD1025	-	Clock data	Clock data (year, month)	<ul style="list-style-type: none"> <li>The year (last two digits) and month are stored as BCD code as shown below.</li> </ul> <div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-right: 10px;"> <table border="1" style="font-size: 8px;"> <tr><td>b15 to b12</td><td>b11 to b8</td><td>b7 to b4</td><td>b3 to b0</td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> </table> </div> <div style="margin-left: 10px;">                         Example:                          1987, July                          H8707                     </div> </div> <div style="display: flex; justify-content: space-around; width: 100%; margin-top: 5px;"> <span>Year</span> <span>Month</span> </div>		b15 to b12	b11 to b8	b7 to b4	b3 to b0				
b15 to b12	b11 to b8	b7 to b4	b3 to b0											
D9026	SD1026	-	Clock data	Clock data (day, hour)	<ul style="list-style-type: none"> <li>The day and hour are stored as BCD code as shown below.</li> </ul> <div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-right: 10px;"> <table border="1" style="font-size: 8px;"> <tr><td>b15 to b12</td><td>b11 to b8</td><td>b7 to b4</td><td>b3 to b0</td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> </table> </div> <div style="margin-left: 10px;">                         Example:                          31st, 10 a.m.                          H3110                     </div> </div> <div style="display: flex; justify-content: space-around; width: 100%; margin-top: 5px;"> <span>Day</span> <span>Hour</span> </div>		b15 to b12	b11 to b8	b7 to b4	b3 to b0				
b15 to b12	b11 to b8	b7 to b4	b3 to b0											
D9027	SD1027	-	Clock data	Clock data (minute, second)	<ul style="list-style-type: none"> <li>The minute and second are stored as BCD code as shown below.</li> </ul> <div style="display: flex; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-right: 10px;"> <table border="1" style="font-size: 8px;"> <tr><td>b15 to b12</td><td>b11 to b8</td><td>b7 to b4</td><td>b3 to b0</td></tr> <tr><td> </td><td> </td><td> </td><td> </td></tr> </table> </div> <div style="margin-left: 10px;">                         Example:                          35 min, 48 sec.                          H3548                     </div> </div> <div style="display: flex; justify-content: space-around; width: 100%; margin-top: 5px;"> <span>Minute</span> <span>Second</span> </div>	b15 to b12	b11 to b8	b7 to b4	b3 to b0					
b15 to b12	b11 to b8	b7 to b4	b3 to b0											

\*1: The relevant modules are as follows:  
 • The Universal model QCPU whose serial number (first five digits) is "10102" or later.  
 • Q00UJCPU, Q00UCPU, Q01UCPU

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU
D9028	SD1028	—	Clock data	Clock data (day of week)	<ul style="list-style-type: none"> <li>The day of the week is stored as BCD code as shown below.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>
D9035	SD1035	SD648	Extension file register	Use block No.	<ul style="list-style-type: none"> <li>Stores the block No. of the extension file register being used in BCD code.</li> </ul>	
D9036	SD1036	×	Extension file register for designation of device number	Device number when individual devices from extension file register are directly accessed	<ul style="list-style-type: none"> <li>Designate the device number for the extension file register for direct read and write in 2 words at SD1036 and SD1037 in BIN data.</li> <li>Use consecutive numbers beginning with R0 of block No. 1 to designate device numbers.</li> </ul>	
D9037	SD1037	×				
D9038	SD1038	SD207	LED display priority ranking	Priorities 1 to 4	<ul style="list-style-type: none"> <li>Sets priority of ERROR LEDs which illuminate (or flicker) to indicate errors with error code numbers.</li> <li>Configuration of the priority setting areas is as shown below.</li> </ul>	Qn(H) QnPH
D9039	SD1039	SD208		Priorities 5 to 7		
D9044	SD1044	×	For sampling trace	Step or time during sampling trace	<ul style="list-style-type: none"> <li>Turned on/off with a peripheral device.</li> <li>When STRA or STRAR instruction is executed, the value stored in SD1044 is used as the sampling trace condition.</li> <li>At scanning-----0</li> <li>At time-----Time (10 msec unit)</li> <li>The value is stored into SD1044 in BIN code.</li> </ul>	
D9049	SD1049	×	Work area for SFC	Block number of extension file register	<ul style="list-style-type: none"> <li>Stores the block number of the expansion file register which is used as the work area for the execution of a SFC program in a binary value.</li> <li>Stores "0" if an empty area of 16K bytes or smaller, which cannot be expansion file register No. 1, is used or if SM320 is OFF.</li> </ul>	
D9050	SD1050	×	SFC program error number	Error code generated by SFC program	<ul style="list-style-type: none"> <li>Stores error code of errors occurred in the SFC program in BIN code.</li> <li>0 : No error</li> <li>80: SFC program parameter error</li> <li>81: SFC code error</li> <li>82: Number of steps of simultaneous execution exceeded</li> <li>83: Block start error</li> <li>84: SFC program operation error</li> </ul>	
D9051	SD1051	×	Error block	Block number where error occurred	<ul style="list-style-type: none"> <li>Stores the block number in which an error occurred in the SFC program in BIN code.</li> <li>In the case of error 83 the starting block number is stored.</li> </ul>	
D9052	SD1052	×	Error step	Step number where error occurred	<ul style="list-style-type: none"> <li>Stores the step number, where error code 84 occurred in an SFC program, in BIN value.</li> <li>Stores "0" when error code 80, 81 or 82 occurred.</li> <li>Stores the block stating step number when error code 83 occurs.</li> </ul>	

\*1: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU



TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU			
D9053	SD1053	x	Error transition	Transition condition number where error occurred	<ul style="list-style-type: none"> <li>Stores the transition condition number, where error code 84 occurred in an SFC program, in BIN value.</li> <li>Stores "0" when error code 80, 81, 82 or 83 occurred.</li> </ul>	Qn(H) QnPH			
D9054	SD1054	x	Error sequence step	Sequence step number where error occurred	<ul style="list-style-type: none"> <li>Stores the sequence step number of transfer condition and operation output in which error 84 occurred in the SFC program in BIN code.</li> </ul>				
D9055	SD1055	SD812	Status latch execution step number	Status latch step	<ul style="list-style-type: none"> <li>Stores the step number when status latch is executed.</li> <li>Stores the step number in a binary value if status latch is executed in a main sequence program.</li> <li>Stores the block number and the step number if status latch is executed in a SFC program.</li> </ul> <div style="text-align: center;"> <table border="1" style="margin: auto;"> <tr> <td style="padding: 2px;">Block No. (BIN)</td> <td style="padding: 2px;">Step No. (BIN)</td> </tr> <tr> <td style="text-align: center;">← Upper 8 bits →</td> <td style="text-align: center;">← Lower 8 bits →</td> </tr> </table> </div>		Block No. (BIN)	Step No. (BIN)	← Upper 8 bits →
Block No. (BIN)	Step No. (BIN)								
← Upper 8 bits →	← Lower 8 bits →								
D9072	SD1072	x	PLC communication check	Data check of serial communication module	<ul style="list-style-type: none"> <li>In the self-loopback test of the serial communication module, the serial communication module writes/reads data automatically to make communication checks.</li> </ul>	Qn(H) QnPH			
D9085	SD1085	x	Register for setting time check value	1 s to 65535 s	<ul style="list-style-type: none"> <li>Sets the time check time of the data link instructions (ZNRD, ZNWR) for the MELSECNET/10.</li> <li>Setting range : 1 s to 65535 s (1 to 65535)</li> <li>Setting unit : 1 s</li> <li>Default value : 10 s (If 0 has been set, default 10 s is applied)</li> </ul>	Qn(H) QnPH			
D9090	SD1090	x	Number of special functions modules over	Number of special functions modules over	<ul style="list-style-type: none"> <li>For details, refer to the manual of each microcomputer program package.</li> </ul>	Qn(H) QnPH QnU <sup>*1</sup>			
D9091	SD1091	x	Detailed error code	Self-diagnosis detailed error code	<ul style="list-style-type: none"> <li>Stores the detail code of cause of an instruction error.</li> </ul>				
D9094	SD1094	SD251	Head I/O number of I/O module to be replaced	Head I/O number of I/O module to be replaced	<ul style="list-style-type: none"> <li>Stores the first two digits of the head I/O number of the I/O module, which will be dismantled/mounted online (with power on), in BIN value.</li> <li>Example) Input module X2F0 → H2F</li> </ul>	Qn(H) QnPH			
D9095	SD1095	SD200	DIP switch information	DIP switch information	<ul style="list-style-type: none"> <li>The DIP switch information of the CPU module is stored in the following format.</li> <li>0: OFF</li> <li>1: ON</li> </ul> <div style="text-align: center;"> </div>	Qn(H) QnPH			

\*1: The relevant modules are as follows:

- The Universal model QCPU whose serial number (first five digits) is "10102" or later.
- Q00UJCPU, Q00UCPU, Q01UCPU

A

Appendix 4 SPECIAL REGISTER LIST

TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																																																																																																																														
D9100	SD1100	-	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown	<ul style="list-style-type: none"> <li>Output module numbers (in units of 16 points), of which fuses have blown, are entered in bit pattern. (Preset output module numbers when parameter setting has been performed.)</li> </ul> <div style="text-align: center;"> <table border="1" style="margin: auto;"> <tr> <td></td> <td>b15</td><td>b14</td><td>b13</td><td>b12</td><td>b11</td><td>b10</td><td>b9</td><td>b8</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td> </tr> <tr> <td>SD1100</td> <td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td></td> <td></td><td></td><td></td><td>(YCB)</td><td></td><td></td><td></td><td>(Y80)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> <tr> <td>SD1101</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td>SD1107</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td></td> <td></td><td></td><td></td><td>(Y7)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>(Y7)</td><td></td><td></td><td></td> </tr> <tr> <td></td> <td></td><td></td><td></td><td>(B0)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>(30)</td><td></td><td></td><td></td> </tr> </table> <p style="text-align: center;">↑ Indicates fuse blow.</p> </div> <ul style="list-style-type: none"> <li>Fuse blow check is executed also to the output module of remote I/O station. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)</li> </ul>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1100	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0					(YCB)				(Y80)									SD1101	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SD1107	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0					(Y7)									(Y7)								(B0)									(30)				Qn(H) QnPH QnU*1																																																							
	b15					b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																																																																																																
SD1100	0					0	0	1	0	0	0	1	0	0	0	0	0	0	0	0																																																																																																																																																																
								(YCB)				(Y80)																																																																																																																																																																								
SD1101	0					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																																																																																
SD1107	0					0	0	0	1	0	0	0	0	0	0	0	1	0	0	0																																																																																																																																																																
								(Y7)									(Y7)																																																																																																																																																																			
				(B0)									(30)																																																																																																																																																																							
D9108	SD1108	-	Step transfer monitoring timer setting	Timer setting valve and the F number at time out	<ul style="list-style-type: none"> <li>Set the value of the step transition monitoring timer and the annunciator number (F number) that will be turned ON when the monitoring timer times out.</li> </ul> <div style="text-align: center;"> <table border="1" style="margin: auto;"> <tr> <td>b15</td><td>to</td><td>b8</td><td>b7</td><td>to</td><td>b0</td> </tr> <tr> <td colspan="2">↑</td><td colspan="2">↑</td><td colspan="2">↑</td> </tr> <tr> <td colspan="4">F number setting (02 to 255)</td> <td colspan="2">Timer time limit setting (1 to 255 s;(1 s units))</td> </tr> </table> </div> <ul style="list-style-type: none"> <li>By turning ON any of SM1108 to SM1114, the monitoring timer starts. If the transition condition following a step which corresponds to the timer is not established within set time, set annunciator (F) is turned on.)</li> </ul>	b15	to	b8	b7	to	b0	↑		↑		↑		F number setting (02 to 255)				Timer time limit setting (1 to 255 s;(1 s units))		Qn(H) QnPH																																																																																																																																																												
b15	to					b8	b7	to	b0																																																																																																																																																																											
↑						↑		↑																																																																																																																																																																												
F number setting (02 to 255)						Timer time limit setting (1 to 255 s;(1 s units))																																																																																																																																																																														
D9110	SD1110					-	I/O module verification error	Bit pattern, in units of 16 points, indicating the modules with verification errors.	<ul style="list-style-type: none"> <li>When I/O modules, of which data are different from those entered at power-ON, have been detected, the I/O module numbers (in units of 16 points) are entered in bit pattern. (Preset I/O module numbers set in parameters when parameter setting has been performed.)</li> </ul> <div style="text-align: center;"> <table border="1" style="margin: auto;"> <tr> <td></td> <td>b15</td><td>b14</td><td>b13</td><td>b12</td><td>b11</td><td>b10</td><td>b9</td><td>b8</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td> </tr> <tr> <td>SD1116</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td> </tr> <tr> <td></td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>(XY)</td> </tr> <tr> <td></td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>(0)</td> </tr> <tr> <td>SD1117</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td></td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>(XY)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> <tr> <td></td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>(B0)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> <tr> <td>SD1123</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td> </tr> <tr> <td></td> <td></td><td></td><td></td><td></td><td></td><td>(XY)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> <tr> <td></td> <td></td><td></td><td></td><td></td><td></td><td>(B0)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </table> <p style="text-align: center;">↑ Indicates an I/O module verify error.</p> </div> <ul style="list-style-type: none"> <li>I/O module verify check is executed also to remote I/O station modules. (If normal status is restored, clear is not performed. Therefore, it is required to perform clear by user program.)</li> </ul>		b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	SD1116	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1																	(XY)																	(0)	SD1117	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0									(XY)																	(B0)									SD1123	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0							(XY)																	(B0)											Qn(H) QnPH QnU*1
	b15									b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0																																																																																																																																																												
SD1116	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	1																																																																																																																																																																
																				(XY)																																																																																																																																																																
																				(0)																																																																																																																																																																
SD1117	0	0	0	0	0					0	0	1	0	0	0	0	0	0	0	0																																																																																																																																																																
								(XY)																																																																																																																																																																												
								(B0)																																																																																																																																																																												
SD1123	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0																																																																																																																																																																				
						(XY)																																																																																																																																																																														
						(B0)																																																																																																																																																																														
D9124	SD1124	SD63	Number of annunciator detections	Number of annunciator detections	<ul style="list-style-type: none"> <li>When one of F0 to 255 (F0 to 2047 for AuA and AnU) is turned on by SET F instruction 1 is added to the contents of SD63. When RST F or LEDR instruction is executed, 1 is subtracted from the contents of SD63.</li> <li>Quantity, which has been turned on by SET F instruction is stored into SD63 in BIN code. The value of SD63 is maximum 16.</li> </ul>																																																																																																																																																																															

\*1: The relevant modules are as follows:  
 • The Universal model QCPU whose serial number (first five digits) is "10102" or later.  
 • Q00UJCPU, Q00UCPU, Q01UCPU

TableApp.4.13 Special register

ACPU Special Register	Special Register after Conversion	Special Register for Modification	Name	Meaning	Details	Corresponding CPU																																																																																																																																																																																																															
D9125	SD1125	SD64	Annunciator detection number	Annunciator detection number	<ul style="list-style-type: none"> <li>When any of F0 to 2047 is turned on by SET F instruction, the annunciator numbers (F numbers) that are turned on in order are registered into SD1125 to SD1132.</li> <li>The F number turned off by RST F instruction is erased from any of SD1125 to SD1132, and the F numbers stored after the erased F number are shifted to the preceding registers.</li> </ul> By executing LEDR instruction, the contents of SD1125 to SD1132 are shifted upward by one. When there are 8 annunciator detections, the 9th one is not stored into SD1125 to SD1132 even if detected.	Qn(H) QnPH QnU <sup>*1</sup>																																																																																																																																																																																																															
D9126	SD1126	SD65																																																																																																																																																																																																																			
D9127	SD1127	SD66																																																																																																																																																																																																																			
D9128	SD1128	SD67																																																																																																																																																																																																																			
D9129	SD1129	SD68																																																																																																																																																																																																																			
D9130	SD1130	SD69																																																																																																																																																																																																																			
D9131	SD1131	SD70																																																																																																																																																																																																																			
D9132	SD1132	SD71																																																																																																																																																																																																																			
<div style="text-align: center;"> <table border="0"> <tr> <td></td> <td>SET</td><td>SET</td><td>SET</td><td>RST</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>SET</td><td>LEDR</td> </tr> <tr> <td></td> <td>F50</td><td>F25</td><td>F99</td><td>F25</td><td>F15</td><td>F70</td><td>F65</td><td>F38</td><td>F110</td><td>F151</td><td>F210</td><td></td><td></td><td></td><td></td><td></td> </tr> </table> </div> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <tr> <td>SD1009</td> <td>0</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>99</td> </tr> <tr> <td>SD1124</td> <td>0</td><td>1</td><td>2</td><td>3</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>8</td><td>8</td><td>8</td><td>8</td><td>8</td> </tr> <tr> <td>SD1125</td> <td>0</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>50</td><td>99</td> </tr> <tr> <td>SD1126</td> <td>0</td><td>0</td><td>25</td><td>25</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>99</td><td>15</td> </tr> <tr> <td>SD1127</td> <td>0</td><td>0</td><td>0</td><td>99</td><td>0</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>15</td><td>70</td> </tr> <tr> <td>SD1128</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>70</td><td>65</td> </tr> <tr> <td>SD1129</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>65</td><td>65</td><td>65</td><td>65</td><td>65</td><td>65</td><td>65</td><td>65</td><td>38</td> </tr> <tr> <td>SD1130</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>38</td><td>38</td><td>38</td><td>38</td><td>38</td><td>38</td><td>38</td><td>110</td> </tr> <tr> <td>SD1131</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>110</td><td>110</td><td>110</td><td>110</td><td>110</td><td>110</td><td>151</td> </tr> <tr> <td>SD1132</td> <td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>151</td><td>151</td><td>151</td><td>151</td><td>151</td><td>210</td> </tr> </table>								SET	SET	SET	RST	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	LEDR		F50	F25	F99	F25	F15	F70	F65	F38	F110	F151	F210						SD1009	0	50	50	50	50	50	50	50	50	50	50	50	50	50	50	99	SD1124	0	1	2	3	2	3	4	5	6	7	8	8	8	8	8	8	SD1125	0	50	50	50	50	50	50	50	50	50	50	50	50	50	50	99	SD1126	0	0	25	25	99	99	99	99	99	99	99	99	99	99	99	15	SD1127	0	0	0	99	0	15	15	15	15	15	15	15	15	15	15	70	SD1128	0	0	0	0	0	0	70	70	70	70	70	70	70	70	70	65	SD1129	0	0	0	0	0	0	0	65	65	65	65	65	65	65	65	38	SD1130	0	0	0	0	0	0	0	0	38	38	38	38	38	38	38	110	SD1131	0	0	0	0	0	0	0	0	0	110	110	110	110	110	110	151	SD1132	0	0	0	0	0	0	0	0	0	0	151	151	151	151	151	210			
	SET	SET					SET	RST	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	LEDR																																																																																																																																																																																																	
	F50	F25	F99	F25	F15	F70	F65	F38	F110	F151	F210																																																																																																																																																																																																										
SD1009	0	50	50	50	50	50	50	50	50	50	50	50	50	50	50	99																																																																																																																																																																																																					
SD1124	0	1	2	3	2	3	4	5	6	7	8	8	8	8	8	8																																																																																																																																																																																																					
SD1125	0	50	50	50	50	50	50	50	50	50	50	50	50	50	50	99																																																																																																																																																																																																					
SD1126	0	0	25	25	99	99	99	99	99	99	99	99	99	99	99	15																																																																																																																																																																																																					
SD1127	0	0	0	99	0	15	15	15	15	15	15	15	15	15	15	70																																																																																																																																																																																																					
SD1128	0	0	0	0	0	0	70	70	70	70	70	70	70	70	70	65																																																																																																																																																																																																					
SD1129	0	0	0	0	0	0	0	65	65	65	65	65	65	65	65	38																																																																																																																																																																																																					
SD1130	0	0	0	0	0	0	0	0	38	38	38	38	38	38	38	110																																																																																																																																																																																																					
SD1131	0	0	0	0	0	0	0	0	0	110	110	110	110	110	110	151																																																																																																																																																																																																					
SD1132	0	0	0	0	0	0	0	0	0	0	151	151	151	151	151	210																																																																																																																																																																																																					

\*1: The relevant modules are as follows:  
 • The Universal model QCPU whose serial number (first five digits) is "10102" or later.  
 • Q00UJCPU, Q00UCPU, Q01UCPU



(11) QCPU with built-in Ethernet port

TableApp.4.14 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU															
SD1270	Operation result	Stores operation result.	Stores the operation result of the time setting function. 0: Not executed 1: Success FFFFH: Failure																		
SD1271	Time setting function	Stores time acquired with time setting function.	Stores years (last two digits of the Christian Era) and months by two digits of BCD code.  <div style="display: flex; align-items: center;"> <div style="text-align: center; margin-right: 10px;">             b15 to b12 b11 to b8 b7 to b4 b3 to b0 ----- Year                      Month           </div> <div style="margin-left: 20px;">             Example:              July, 1993              9307H           </div> </div>																		
SD1272			Stores dates and hours acquired with time setting function by two digits of BCD code.  <div style="display: flex; align-items: center;"> <div style="text-align: center; margin-right: 10px;">             b15 to b12 b11 to b8 b7 to b4 b3 to b0 ----- Day                                      Hour           </div> <div style="margin-left: 20px;">             Example:              31st, 10 a.m.              3110H           </div> </div>																		
SD1273			Stores minutes and seconds acquired with time setting function by two digits of BCD code.  <div style="display: flex; align-items: center;"> <div style="text-align: center; margin-right: 10px;">             b15 to b12 b11 to b8 b7 to b4 b3 to b0 ----- Minute                                      Second           </div> <div style="margin-left: 20px;">             Example:              35 min., 48 sec.              3548H           </div> </div>	S (status change)																	
SD1274			Stores years (first two digits of the Christian Era) and days acquired with time setting function.  <div style="display: flex; align-items: center;"> <div style="text-align: center; margin-right: 10px;">             b15 to b12 b11 to b8 b7 to b4 b3 to b0 ----- Higher digits of year (0 to 99)           </div> <div style="margin-left: 20px;">             Example:              1993, Friday              1905H           </div> </div> <table border="1" style="margin-left: 20px; margin-top: 10px;"> <thead> <tr> <th colspan="2">Day of the week</th> </tr> </thead> <tbody> <tr><td>0</td><td>Sunday</td></tr> <tr><td>1</td><td>Monday</td></tr> <tr><td>2</td><td>Tuesday</td></tr> <tr><td>3</td><td>Wednesday</td></tr> <tr><td>4</td><td>Thursday</td></tr> <tr><td>5</td><td>Friday</td></tr> <tr><td>6</td><td>Saturday</td></tr> </tbody> </table>	Day of the week		0	Sunday	1	Monday	2	Tuesday	3	Wednesday	4	Thursday	5	Friday	6	Saturday	New	QnU*1
Day of the week																					
0	Sunday																				
1	Monday																				
2	Tuesday																				
3	Wednesday																				
4	Thursday																				
5	Friday																				
6	Saturday																				
SD1275	Required response time	Stores time taken from transmission to SNTP server to clock time setup at CPU. Range: 0 to FFFEH (Unit: ms) FFFFH when the above limit is exceeded.																			
SD1276	Forced connection invalidation	Specifies forced connection invalidation.	Specify this when a connection is to be invalidated forcibly on the user program. If invalidation is specified for a connection, it stops communication and does not respond. (When a remote password is used and frequent unlock processing errors have occurred on a connection, this is useful for temporarily inhibiting access to the connection.)  <div style="display: flex; align-items: center;"> <div style="text-align: center; margin-right: 10px;">             b15b14 to b1 b0 ----- SD1276           </div> <div style="margin-left: 20px;">             Connection 1              Connection 2              ...              Connection 15              Connection 16           </div> </div>																		
SD1277			0: Valid (default) 1: Invalid This register is to be invalidated if a socket communication is used as an open system.  <div style="display: flex; align-items: center;"> <div style="text-align: center; margin-right: 10px;">             b15b14 b13 b12 to b4 b3 b2 b1 b0 ----- SD1277           </div> <div style="margin-left: 20px;">             MELSOFT communication port (UDP/IP)              MELSOFT communication port (TCP/IP)              FTP communication port              Directconnection to MELSOFT           </div> </div>	U																	

\*1: This applies to the Built-in Ethernet port QCPU.

TableApp.4.15 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9 □ □ □	Corresponding CPU
SD1282	Open completion signal	Stores open completion status	<p>Open completion status of connections (whose open system is socket communication) using socket communication functions is stored. All bits corresponding to connections using any communications other than the socket communication are fixed to "0".</p> <p>0 : Open processing is not completed. 1 : Open processing is completed.</p>	S (Status change)	New	QnU <sup>2</sup>
SD1284	Open request signal	Stores open request status	<p>Open request status of connections using socket communication functions is stored. All bits corresponding to connections using any communications other than the socket communication are fixed to "0".</p> <p>0 : No open requests 1 : In open request</p>	S (Status change)	New	QnU <sup>2</sup>
SD1286	Reception status signal	Stores reception status	<p>Reception status of connections using socket communication functions is stored. All bits corresponding to connections using any communications other than the socket communication are fixed to "0".</p> <p>For TCP (Normal reception mode) 0 : Data have not been received. 1 : Data have been received.</p> <p>For TCP (Fixed length reception mode) 0 : Data have not been received , or received data size has not been reached to valid buffer size. 1 : Received data size has been reached to valid buffer size.</p> <p>For UDP 0 : Data have not been received. 1 : Data have been received.</p>	S (Status change)	New	QnU <sup>2</sup>
SD1288	Built-in Ethernet port connection status	Stores connection status of built-in Ethernet port	<p>Connection status of built-in Ethernet port is stored.</p> <p>0 : Not connected with or disconnected from hubs or devices. 1 : Connected to hubs or devices</p> <p>It may take several seconds for the QCPU to determine whether to connect or disconnect a built-in Ethernet port.</p>	S (Status change)	New	QnU <sup>2</sup>

\*2: The built-in Ethernet port QCPU whose serial number (first five digits) is "11012" or later is targeted.

A

Appendix 4 SPECIAL REGISTER LIST

(12) Fuse blown module

TableApp.4.16 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1300 SD1301 SD1302 SD1303 SD1304 SD1305 SD1306 SD1307 SD1308 SD1309 to SD1330 SD1331	Fuse blown module	Bit pattern in units of 16 points, indicating the modules whose fuses have blown 0 : No blown fuse 1 : Blown fuse present	<ul style="list-style-type: none"> <li>The numbers of output modules whose fuses have blown are input as a bit pattern (in units of 16 points). (If the module numbers are set by parameter, the parameter-set numbers are stored.)</li> <li>Also detects blown fuse condition at remote station output modules</li> </ul>	S (Error)	D9100 D9101 D9102 D9103 D9104 D9105 D9106 D9107 New New New	Qn(H) QnPH QnPRH QnU
<ul style="list-style-type: none"> <li>Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.</li> </ul>						

(13) I/O module verification

TableApp.4.17 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1400 SD1401 SD1402 SD1403 SD1404 SD1405 SD1406 SD1407 SD1408 SD1409 to SD1430 SD1431	I/O module verify error	Bit pattern, in units of 16 points, indicating the modules with verification errors. 0 : No I/O verification errors 1 : I/O verification error present	<ul style="list-style-type: none"> <li>When the I/O modules whose I/O module information differs from that registered at power-ON are detected, the numbers of those I/O modules are entered in bit pattern. (If the I/O numbers are set by parameter, the parameter-set numbers are stored.)</li> <li>Also detects I/O module information.</li> </ul>	S (Error)	D9116 D9117 D9118 D9119 D9120 D9121 D9122 D9123 New New New	Qn(H) QnPH QnPRH QnU
<ul style="list-style-type: none"> <li>Not cleared even if the blown fuse is replaced with a new one. This flag is cleared by error resetting operation.</li> </ul>						

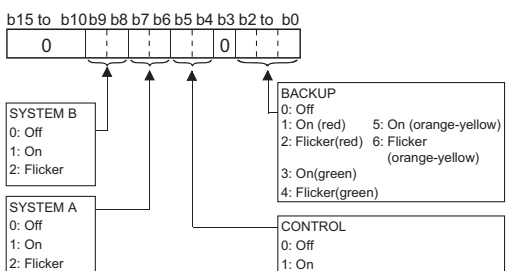
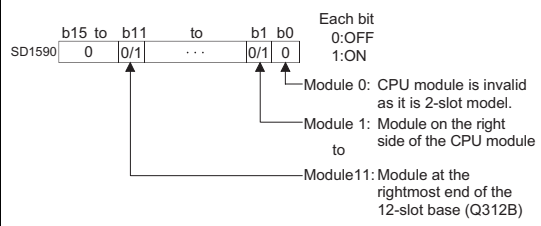
(14) Process control instructions

TableApp.4.18 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1500 SD1501	Basic period	Basic period tome	<ul style="list-style-type: none"> <li>Set the basic period (1 second units) use for the process control instruction using floating point data.</li> </ul> <p>Floating point data = <input type="text" value="SD1501"/> <input type="text" value="SD1500"/></p>	U	New	QnPH
SD1502	Process control instruction detail error code	Process control instruction detail error code	<ul style="list-style-type: none"> <li>Shows the detailed error contents for the error that occurred in the process control instruction.</li> </ul>	S (Error)	New	
SD1503	Process control instruction generated error location	Process control instruction generated error location	<ul style="list-style-type: none"> <li>Shows the error process block that occurred in the process control instruction.</li> </ul>	S (Error)	New	QnPH QnPRH
SD1506 SD1507	Dummy device	Dummy device	<ul style="list-style-type: none"> <li>Used to specify dummy devices by a process control instruction.</li> </ul>	U	New	
SD1508	Function availability selection for process control instruction	b0 Bumpless function availability setting for the S.PIDP instruction 0: Enabled 1: Disabled (Default: 0)	<ul style="list-style-type: none"> <li>Selects the availability (enabled/disabled) of the function for process control instructions.</li> </ul>	U	New	QnPH QnPRH

(15) For redundant systems (Host system CPU information \*1)  
SD1510 to SD1599 are only valid for redundant systems.  
They are all set to 0 for stand-alone systems.

TableApp.4.19 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU D9□□□	Corresponding CPU
SD1585	Redundant system LED status	4 LED states • BACKUP • CONTROL • SYSTEM A • SYSTEM B	The LED status for BACKUP, CONTROL, SYSTEM A, SYSTEM B is stored in the following format: 	S (status change)	New	QnPRH
SD1588	Reason(s) for system switching	Reason(s) for system switching that occurred in host station	Stores the reason(s) for system switching on the host system. The following values are stored corresponding to the methods for system switching: Initialized to 0 when the power supply is switched off and then on or the RESET switch is set to the RESET position and then to the neutral position. 0: Initial value (control system has not been switched) 1: Power off, Reset, H/W failure, WDT error, 2: CPU stop error (except WDT) 3: System switching request from network module 16: System switching dedicated instruction 17: System switching request from GX Developer	S (when condition occurs)	○	
SD1589	Reason(s) for system switching failure conditions	Reason(s) for system switching failure No.	• Stores the reason(s) for system switching failure. 0: System switching normal (default) 1: Tracking cable is not connected, tracking cable error, FPGA circuit failure. 2: H/W failure, power-OFF, Reset, WDT error on the standby system 3: H/W failure, power-OFF, Reset, WDT error on the Control system 4: Tracking data transfer initialization 5: Communication timeout 6: Serious error(except WDT error) on the Standby system 7: There is difference between both systems (detected as Backup mode only) 8: During memory copy from control system to standby system 9: During online program change 10: During detection of intelligent function module failure on the standby system 11: System switching being executed • Resets to "0" when host system is powered on. • Resets to "0" once system has been switched successfully.	S(when system is switched)	○	QnPRH
SD1590	Network module head address, which requested system switching	Network module head address, which requested system switching	• Stores head address of network module which a system switch request was initiated. • Turns off automatically by system, after network error is reset by user.  • Please refer to SD1690 which stores the corresponding head address of network module on other system.	S (Error/Status change)	New	QnPRH
SD1595	Memory copy target I/O number	Memory copy target I/O number	• Stores the memory copy target I/O No.(Standby system CPU module: 3D1H) of before SM1595 is turned from OFF to ON.	U	New	
SD1596	Memory copy status	Memory copy status	• Stores the execution result of Memory copy function. 0 : Memory copy successfully completed 4241H : Standby system power supply off 4242H : Tracking cable is disconnected or is damaged 4247H : Memory copy function is being executed 4248H : Unsupported memory copy destination I/O Number	S (Status change)	New	

\*1: The information of the host CPU module is stored.

A

Appendix 4 SPECIAL REGISTER LIST

(16) For redundant systems (Other system CPU information \*1)

SD1600 to SD1659 is only valid during the back up mode for redundant systems, and refresh cannot be done when in the separate mode.

SD1651 to SD1699 are valid in either the backup mode or separate mode.

When a stand-alone system SD1600 to SD1699 are all 0.

TableApp.4.20 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU SD□□*2	Corresponding CPU
SD1600	System error information	System error information	<ul style="list-style-type: none"> <li>If an error is detected by the error check for redundant system, the corresponding bit shown below turns ON. That bit turns OFF when the error is cleared after that.</li> </ul> <ul style="list-style-type: none"> <li>If any of b0, b1, b2 and b15 is ON, the other bits are all OFF.</li> <li>In the debug mode, b0, b1, b2 and b15 are all OFF.</li> </ul>	S(Every END)	-	
SD1601	System switching results	System switching results	<ul style="list-style-type: none"> <li>Stores the reasons for system switching.</li> <li>Stores the reasons for system switching into SD1601 of both systems when system switching occurred.</li> <li>Initialized to 0 at power OFF to ON/reset to unreset.</li> <li>The following shows values stored into this register. <ul style="list-style-type: none"> <li>0: Initial value (System switching has not occurred)</li> <li>1: Power-OFF, Reset, H/W failure, WDT error.(*)</li> <li>2: CPU stop error (except WDT)</li> <li>3: System switching request by network module</li> <li>16: System switching dedicated instruction</li> <li>17: System switching request from GX Developer</li> </ul> </li> <li>*: When the system is switched by the power OFF/reset of the control system, "1" is not stored into SD1601 of the new standby system.</li> </ul>	S(when system is switched)		QnPRH
SD1602	System switching dedicated instruction parameter	System switching dedicated instruction parameter	<ul style="list-style-type: none"> <li>Stores the parameters for system switching dedicated instruction SP.CONTSW. (The parameters (SD1602) for SP.CONTSW are stored in both systems A&amp;B)</li> <li>SD1602 is only valid when "16" is stored in SD1601.</li> <li>This SD1602 is updated once system switch instruction SP.CONTSW is activated.</li> </ul>	S(when system is switched)		
SD1610	Other system diagnostic error	Diagnostic error code	<ul style="list-style-type: none"> <li>The error value sorted in BIN code.</li> <li>Stores SD0 of the other system CPU module</li> </ul>	S(Every END)	SD0	
SD1611	Other system diagnostic error occurrence time	Diagnostic error occurrence time	<ul style="list-style-type: none"> <li>Stores the date and time when diagnostics error occurred corresponding to error code stored in SD1610.</li> <li>Data format is the same as SD1 to SD3.</li> <li>Also, stores the value to SD1 to SD3.</li> </ul>	S(Every END)	SD1 to SD3	
SD1612						
SD1613						
SD1614	Other system error information category	Error information category code	<ul style="list-style-type: none"> <li>Stores the category code corresponding to the error comment information/individual information code.</li> <li>Data format is the same as SD4.</li> <li>Also, stores the value to SD4.</li> </ul>	S(Every END)	SD4	
SD1615 to SD1625	Other system error common information	Error common information	<ul style="list-style-type: none"> <li>Stores the common information corresponding to the error code stored in this system CPU.</li> <li>Data composition is the same as SD5 to SD15.</li> <li>Also, stores the value to SD5 to SD15.</li> </ul>	S(Every END)	SD5 to SD15	
SD1626 to SD1636	Other system error individual information	Error individual information	<ul style="list-style-type: none"> <li>Stores the individual information corresponding to the error code stored in this system CPU.</li> <li>Data composition is the same as SD16 to SD26.</li> <li>Also, stores the value to SD16 to SD26.</li> </ul>	S(Every END)	SD16 to SD26	

\*2: Shows the special register (SD□□) for the host system CPU module.



TableApp.4.20 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU SD□□*2	Corresponding CPU
SD1649	Standby system error cancel command	Error code of error to be cleared	<ul style="list-style-type: none"> <li>Stores the error code of the error to be cleared by clearing a standby system error.</li> <li>Stores the error code of the error to be cleared into this register and turn SM1649 from OFF to ON to clear the standby system error.</li> <li>The value in the lowest digit (1 place) of the error code is ignored when stored into this register. (By storing 4100 in this register and resetting the error, errors 4100 to 4109 can be cleared.)</li> </ul>	S(Every END)		
SD1650	Other system operating information	Other system operating information	<p>Stores the operation information of the other system CPU module in the following format. "00FFH" 1 stored when a communication error occurs, or when in debug mode.</p> <p>Note : A communication error is caused by the following:.</p> <ul style="list-style-type: none"> <li>When the power supply is switched off, or when the other system is reset.</li> <li>H/W error occurs on either of system A or B.</li> <li>WDT error occurs.</li> <li>Tracking cable is not connected.</li> <li>Tracking cable is disconnected or damaged.</li> </ul>	S(Every END)	-	QnPRH
SD1690	Network module head address, which requested system switching on host (control) system	Network module head address, which requested system switching on host (control) system	<ul style="list-style-type: none"> <li>Stores head address of network module which a system switch request was initiated, using the following format.</li> <li>Turns off automatically by system, after network error is reset by user.</li> </ul> <p>Please refer to SD1590 which stores the corresponding head address of network module on host system.</p>	S(Every END)		

\*2 : Shows the special register (SD□□) for the host system CPU.

A

Appendix 4 SPECIAL REGISTER LIST

(17) For redundant systems (Trucking)

SD1700 to SD1779 is valid only for redundant systems.

These are all 0 for stand-alone systems.

TableApp.4.21 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corres- ponding ACPU D9□□□	Corresponding CPU
SD1700	Tracking error detection count	Tracking error detection count	<ul style="list-style-type: none"> <li>When the tracking error is detected, count is added by one.</li> <li>The counter repeats an increment and decrement of the value; 0 → 32767 → - 32768 → 0</li> </ul>	S(Error)		QnPRH
SD1710	Waiting time for online program change (standby system)	Waiting time for online program change (standby system)	<ul style="list-style-type: none"> <li>Set in seconds the waiting time of the standby system CPU module from when online program change to the control system CPU module is completed by the online program change for redundancy function until the online program change to the standby system CPU module starts.</li> <li>If no online program change request is issued to the standby system CPU module within the preset time after completion of the online program change to the control system CPU module, both system CPU modules judge it as the failure of the online program change for redundancy. In this case, both system CPU modules resume the consistency check between system A &amp; B suspended during the online program change. Also, the control system CPU module is set to accept a new request of online program change for redundancy.</li> <li>When both systems are powered on, 90 seconds are set to SD1710 as the default value.</li> <li>Set the value within the range 90 to 3600 seconds. When the setting is 0 to 89 seconds, it is regarded as 90 seconds for operation. If the setting is outside the allowed range, it is regarded other than 0 to 3600 seconds for operation.</li> <li>The waiting time for a start of online program change to the standby system CPU module is checked according to the SD1710 setting during online change of multiple blocks and online change of batch of files for redundancy.</li> </ul>	U/S (Initial)	New	QnPRH

(18) Redundant power supply module information

SD1780 to SD1789 are valid only for a redundant power supply system.

The bits are all 0 for a singular power supply system.

TableApp.4.22 Special register

Number	Name	Meaning	Explanation	Set by (When Set)	Corresponding ACPU	Corresponding CPU
SD1780	Power supply off detection status	Power supply off detection status	<ul style="list-style-type: none"> <li>Stores the status of the redundant power supply module with input power OFF in the following bit pattern.</li> <li>Stores 0 when the main base unit is not the redundant power main base unit (Q38RB).</li> </ul> <ul style="list-style-type: none"> <li>When configuring multiple CPU, the status is stored to 1st CPU module.</li> </ul>	S(Every END)	D9□□□ New	
SD1781	Power supply failure detection status	Power supply failure detection status	<ul style="list-style-type: none"> <li>Stores the failure detection status of the redundant power supply module in the following bit pattern. (The corresponding bit is cleared to 0 when the input power to the faulty redundant power supply module is switched OFF after detection of the redundant power supply module failure.)</li> <li>Stores 0 when the main base unit is not the redundant power main base unit (Q38RB).</li> </ul> <ul style="list-style-type: none"> <li>When configuring multiple CPU, the status is stored to 1st CPU module.</li> </ul>	S(Every END)	D9□□□ New	Qn(H) <sup>2</sup> QnPH <sup>2</sup> QnPRH QnU <sup>3</sup>
SD1782	Momentary power failure detection counter for power supply 1 <sup>*1</sup>	Momentary power failure detection count for power supply 1	<ul style="list-style-type: none"> <li>Counts the number of times of momentary power failure of the power supply 1/2.</li> <li>Monitors the status of the power supply 1/2 mounted on the redundant power main base unit (Q38RB) and counts the number of times of momentary power failure. Status of power supply 1/power supply 2 mounted on the redundant extension base unit is not monitored.</li> <li>When the CPU module starts, the counter of the power supply 1/2 is cleared to 0.</li> </ul>	S(Every END)	D9□□□ New	
SD1783	Momentary power failure detection counter for power supply 2 <sup>*1</sup>	Momentary power failure detection count for power supply 2	<ul style="list-style-type: none"> <li>If the input power to one of the redundant power supply modules is turned OFF, the corresponding counter is cleared to 0. The counter is incremented by 1 every time the momentary power failure of the power supply 1/2 is detected. (The counter repeats increment and decrement of the value; 0 → 32767 → -32768 → 0 (The system monitor of GX Developer shows the counter within the range between 0 and 65535.</li> <li>Stores 0 when the main base unit is not the redundant power main base unit (Q38RB).</li> <li>When configuring multiple CPU, the status is stored to 1st CPU module.</li> <li>The counter repeats increment and decrement of the value, 0 → 32767 → -32768 → 0 (The system monitor of GX Developer shows the counter within the range between 0 and 65535.</li> </ul>	S(Every END)	D9□□□ New	

\*1: The "power supply 1" indicates the redundant power supply module mounted on the POWER 1 slot of the redundant base unit (Q38RB/68RB/Q65WRB).

The "power supply 2" indicates the redundant power supply module mounted on the POWER 2 slot of the redundant base unit (Q38RB/68RB/Q65WRB).

\*2: The module whose first 5 digits of serial No. is "07032" or later.

However, for the multiple CPU system configuration, this applies to all CPU modules whose first 5 digits of serial No. are "07032" or later.

\*3: The module whose first 5 digits of serial No. is "10042" or later.

A

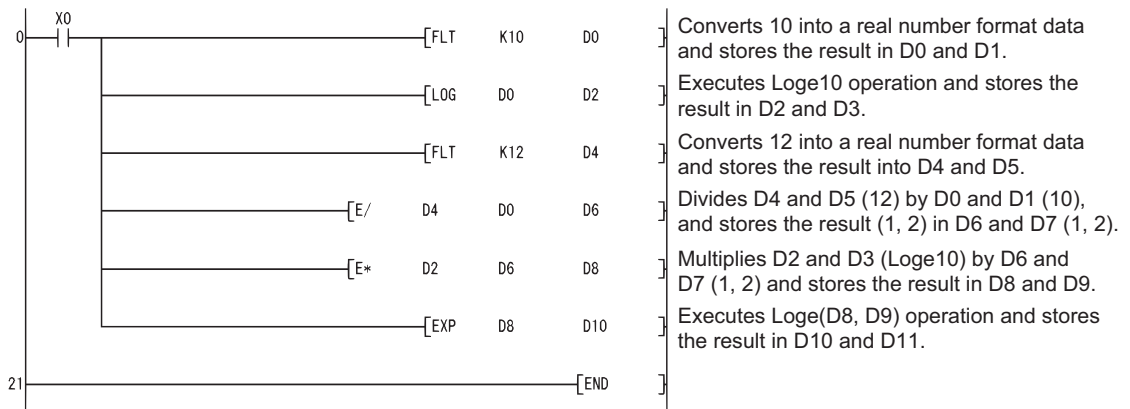
Appendix 4 SPECIAL REGISTER LIST

# Appendix 5 APPLICATION PROGRAM EXAMPLES

## Appendix 5.1 Concept of Programs which Perform Operations of $X^n$ , $\sqrt[n]{X}$

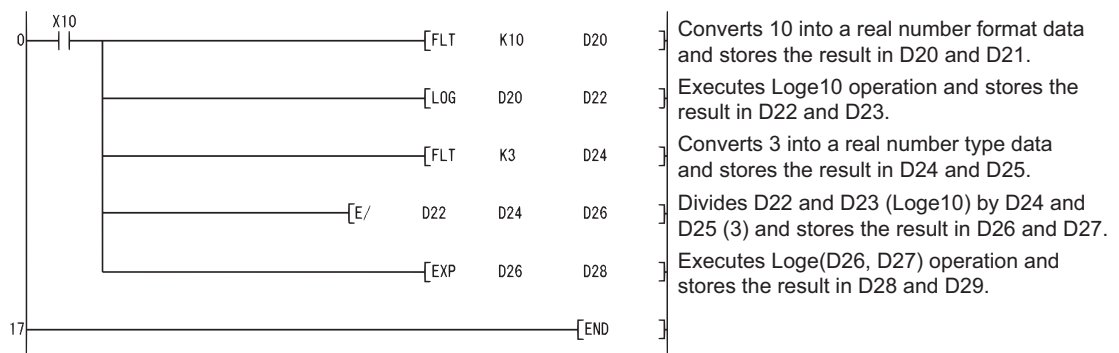
- (1) Concept of programs which perform operations of  $X^n$   
 $X^n$  can be operated using  $e^{(n \log_e X)}$ .

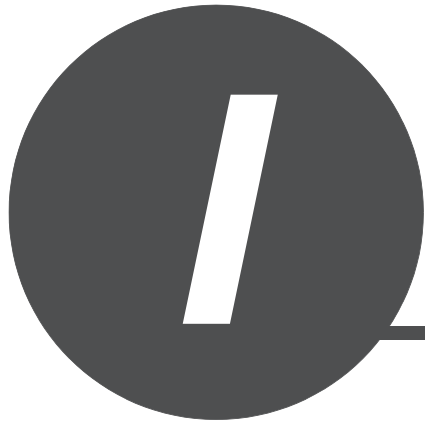
For example, the operation of  $10^{1.2}$  is  $e^{(1.2 \times \log_e 10)}$ , which is represented in the form of a sequence program as shown below.



- (2) Concept of program which performs operation of  $\sqrt[n]{X}$   
 $\sqrt[n]{X}$  can be operated using  $e^{(\frac{1}{n} \log_e X)}$ .

For example, the operation of  $\sqrt[3]{10}$  is  $e^{(\frac{1}{3} \times \log_e 10)}$ , which is represented in the form of a sequence program as shown below.





# INDEX

---



## [Symbols]

- (BIN 16-bit subtraction operations).....	6-22
\$+ (Linking character strings) .....	6-65,6-67
\$=, \$<>, \$>, \$<=, \$<, \$>= (Character string data comparisons).....	6-11
\$MOV (Character string transfers) .....	6-112
* (BIN 16-bit multiplication operations) .....	6-30
+ (BIN 16-bit addition operations).....	6-22
/ (BIN 16-bit division operations) .....	6-30
<(BIN 16-bit data comparisons).....	6-2
<=(BIN 16-bit data comparisons).....	6-2
<>(BIN 16-bit data comparisons).....	6-2
=(BIN 16-bit data comparisons).....	6-2
>(BIN 16-bit data comparisons).....	6-2
>=(BIN 16-bit data comparisons).....	6-2

## [Numerics]

16-bit data block transfers (FMOV) .....	6-120,6-122
16-bit data checks (SUM) .....	7-69
16-bit data exchange (XCH) .....	6-124
16-bit data exclusive NOR operation (WXNR)....	7-27
16-bit data searches (SER) .....	7-66
16-bit dead band controls (BAND).....	7-324
16-bit exclusive OR operations (WXOR) .....	7-19
16-bit negation transfers (CML).....	6-114
16-bit transfers (MOV) .....	6-106
1-bit shift to left of n-bit data (BSFL).....	7-49,7-51
1-bit shift to right of n-bit data (BSFR) .....	7-49,7-51
1-word shift to left of n-word data (DSFL)....	7-54,7-56
1-word shift to right of n-word data (DSFR) .....	7-54,7-56
32-bit data checks (DSUM) .....	7-69
32-bit data exchanges (DXCH).....	6-124
32-bit data exclusive NOR operation (DXNR) ....	7-27
32-bit data searches (DSER).....	7-66
32-bit dead band controls (DBAND) .....	7-324
32-bit exclusive OR operations (DXOR).....	7-19
32-bit negation transfers (DCML) .....	6-114
32-bit transfers (DMOV) .....	6-106
4-bit dissociation of 16-bit data (DIS) .....	7-77
4-bit linking of 16-bit data (UNI).....	7-79
7-segment decode (SEG).....	7-75

## [A]

A contact operation start (LD).....	5-2
A contact parallel connection (OR).....	5-2
A contact series connection (AND).....	5-2
ACOS (COS <sup>-1</sup> operation on floating-point data (Single precision)).....	7-267
ACOSD (COS <sup>-1</sup> operation on floating-point data (Double precision)) .....	7-269
Addition	
Addition of floating decimal point (Double precision) (ED+).....	6-50,6-52
Addition of floating decimal point (Single precision) (E+) .....	6-46,6-48
BCD 4-digit addition (B+) .....	6-34
BCD 8-digit addition (DB+).....	6-38

BIN 16-bit addition operations (+) .....	6-22
BIN 32-bit addition operations (D+).....	6-26
Block addition (BK+).....	6-59,6-62
Addition and subtraction of floating decimal point data (Double precision) (ED+, ED-) .....	6-50,6-52
Addition and subtraction of floating decimal point data (Single precision) (E+, E-).....	6-46,6-48
ADRSET (Indirect address read).....	7-395
ANB (Ladder block series connections).....	5-10
AND (=, <>, >, <=, <, >=) (BIN 16-bit data comparisons) .....	6-2
AND (A contact series connection).....	5-2
AND (D=, D<>, D>, D<=, D<, D>=) (BIN 32-bit data comparisons) .....	6-4
AND (E=, E<>, E>, E<=, E<, E>=) (Floating decimal point data comparisons(Single precision)).....	6-6
AND (ED=, ED<>, ED>, ED<=, ED<, ED>=) (Floating decimal point data comparisons(Double precision)) .....	6-8
And inverse (ANI) .....	5-3
AND(\$=, \$<>, \$>, \$<=, \$<, \$>=) (Character string data comparisons) .....	6-11
ANDF (Pulse series connections / trailing edge leading edge).....	5-5,5-7
ANDP (Pulse series connections / leading edge leading edge).....	5-5,5-7
ANDPI, ANDFI .....	5-8
ANI (B contact series connection) .....	5-2
Annunciator output (OUT F) .....	5-28
Application instructions .....	2-29
Arithmetic operation instructions.....	2-16
ASC (Conversion from hexadecimal BIN to ASCII) .....	7-228
ASIN (SIN <sup>-1</sup> operation on floating-point data (Single precision)) .....	7-262
ASIND (SIN <sup>-1</sup> operation on floating-point data (Double precision)) .....	7-265
ATAN (TAN <sup>-1</sup> operation on floating-point data (Single precision)) .....	7-271
ATAND (TAN <sup>-1</sup> operation on floating-point data .....	7-273

## [B]

B- (BCD 4-digit subtraction).....	6-34
B contact operation start (LDI).....	5-2
B* (BCD 4-digit multiplication) .....	6-42
B+ (BCD 4-digit addition) .....	6-34
B/ (BCD 4-digit division).....	6-42
BACOS (BCD type COS <sup>-1</sup> operation) .....	7-317
BAND (16-bit dead band controls).....	7-324
Basic instructions .....	2-10
BASIN (BCD type SIN <sup>-1</sup> operation).....	7-315
BATAN (BCD type TAN <sup>-1</sup> operation) .....	7-313
Batch recovery of index register (ZPOP) .....	7-400
Batch reset of bit devices (BKRST) .....	7-64
Batch save of index register (ZPUSH).....	7-400
BCD (BIN data to 4-digit) .....	6-73
BCD 4-digit addition and subtraction operations (B+, B-) .....	6-34

BCD 4-digit multiplication and division operations (B*, B/)	6-42	BK+ (Block addition)	6-59,6-62
BCD 4-digit square roots (BSQR)	7-306	BKAND (Block logical products)	7-9
BCD 8-digit addition and subtraction operations (DB+, DB-)	6-38	BKBCD (Conversion from block BIN 16-bit data to BCD 4-digit data)	6-98
BCD 8-digit multiplication and division operations (DB*, DB/)	6-44	BKBIN (Conversion from block BCD 4-digit data to block BIN 16-bit data)	6-100
BCD 8-digit square roots (BDSQR)	7-306	BKCMP □ (BIN block data comparisons)	6-15,6-18
BCD conversion		BKOR (Block logical sum operations)	7-17
Conversion from BIN data to 4-digit BCD (BCD)	6-73	BKRST (Batch reset of bit devices)	7-64
Conversion from BIN data to 8-digit BCD (DBCD)	6-73	BKXNR (Block exclusive NOR operations)	7-33
BCD type COS operations (BCOS)	7-311	BKXOR (Block exclusive OR operations)	7-25
BCD type COS <sup>-1</sup> operations (BACOS)	7-317	Block 16-bit exchanges (BXCH)	6-126
BCD type SIN operation (BSIN)	7-309	Block 16-bit transfers (BMOV)	6-117
BCD type SIN <sup>-1</sup> operation (BASIN)	7-315	Block addition (BK+)	6-59,6-62
BCD type TAN operation (BTAN)	7-313	Block exclusive NOR operations (BKXNR)	7-33
BCD type TAN <sup>-1</sup> operations (BATAN)	7-319	Block exclusive OR operations (BKXOR)	7-25
BCDDA (Conversion from BCD 4-digit to decimal ASCII)	7-189	Block logical products (BKAND)	7-9
BCOS (BCD type COS operations)	7-311	Block logical sum operations (BKOR)	7-17
BDSQR (BCD 8-digit square roots)	7-306	Block subtraction (BK-)	6-59,6-62
BIN (BCD 4-digit data to BIN data)	6-75	BMOV (Block 16-bit data transfers)	6-117
BIN 16-bit addition and subtraction operations (+, -)	6-22	BREAK (Forced end of FOR to NEXT instruction loop)	7-108
BIN 16-bit data comparisons (=, <>, >, <=, <, >=)	6-2	BRST (Bit reset for word devices)	7-59
BIN 16-bit data sort operations (SORT)	7-95	BSET (Bit set for word devices)	7-59
BIN 16-bit multiplication and division operations (*, /)	6-30	BSFL (1-bit shift to left of n-bit data)	7-49,7-51
BIN 16-bit to BIN 32-bit (DBL)	6-88	BSFR (1-bit shift to right of n-bit data)	7-49,7-51
BIN 16-bit to Gray code (GRY)	6-90	BSIN (BCD type SIN operation)	7-309
BIN 32-bit addition and subtraction operations (D+, D-)	6-26	BSQR (BCD 4-digit square roots)	7-306
BIN 32-bit block data comparisons (DBKCMPP □, DBKCMPP □ P)	6-18	BTAN (BCD type TAN operation)	7-313
BIN 32-bit data block addition and subtraction operations (DBK+(P),DBK-(P))	6-62	BTOW (Data linking in byte units)	7-85
BIN 32-bit data comparisons (D=, D<>, D>, D<=, D<, D>=)	6-4	Buffer memory access instructions	2-41
BIN 32-bit data sort operations (DSORT)	7-95	BXCH (Block 16-bit data exchanges)	6-126
BIN 32-bit data to BIN 16-bit data (WORD)	6-89		
BIN 32-bit data to Gray code (DGRY)	6-90	[C]	
BIN 32-bit multiplication and division operations (D*, D/)	6-32	Calculation of averages for 16-bit or 32-bit data (MEAN(P),DMEAN(P))	7-103
BIN block data comparisons (BKCMP □) ...	6-15,6-18	Calculation of totals for 16-bit data (WSUM)	7-99
BINDA (Conversion from BIN 16-bit data to decimal ASCII)	7-183	Calculation of totals for 32-bit data (DWSUM)	7-101,7-103
BINHA (Conversion from BIN 16-bit data to hexadecimal ASCII)	7-186	CALL (Subroutine program calls)	7-110
Bit data	3-3	Cautions on programming	3-27
Bit device output reverse (FF)	5-40	Changing check format of CHK instruction (CHKCIR, CHKEND)	7-179
Bit device shifts (SET)	5-44	Character string data	3-11
Bit processing instructions	2-34	Character string data comparisons	6-11
Bit reset for word devices (BRST)	7-59	Character string length detection (LEN)	7-204
Bit set for word devices (BSET)	7-59	Character string processing instructions	2-43
Bit tests (TEST/DTEST)	7-61	Character string search (INSTR)	7-239,7-241,7-243
BK- (Block subtraction)	6-59,6-62	Character string transfers (\$MOV)	6-112
		CHKCIR (Changing check format of CHK instruction)	7-179
		CHKEND (Changing check format of CHK instruction)	7-179
		CHKST, CHK (Special format failure checks)	7-175
		CJ (Pointer branch instruction)	6-129
		Clock comparison (TM=,TM>,TM<,TM=)	7-361
		Clock data addition operation (DATE+)	7-348
		Clock data subtraction operation (DATE-)	7-350
		Clock instructions	2-52

CML (16-bit negation transfers).....	6-114	Conversion from ASCII to hexadecimal BIN (HEX)	7-230
COM (Refresh instruction).....	7-134,7-137,7-141	Conversion from BCD 4-digit to decimal ASCII (BCDDA).....	7-189
Common logarithm operation on floating-point data (Double precision) (LOG10D(P)).....	7-302	Conversion from BCD 8-digit to decimal ASCII (DBCDDA).....	7-189
Common logarithm operation on floating-point data (Single precision) (LOG10(P)).....	7-300	Conversion from BIN 16-bit to character string (STR).....	7-206
Comparison operation instruction table.....	2-10	Conversion from BIN 16-bit to decimal ASCII (BINDA).....	7-183
Comparison operation instructions.....	6-2	Conversion from BIN 16-bit to floating decimal point (Double precision) (FLTD).....	6-81
Comparisons (BIN 16-bit data).....	6-2	Conversion from BIN 16-bit to floating decimal point (Single precision) (FLT).....	6-78
Comparisons (BIN 32-bit data).....	6-4	Conversion from BIN 16-bit to hexadecimal ASCII (BINHA).....	7-186
Comparisons (Character string data).....	6-11	Conversion from BIN 32-bit to character string (DSTR).....	7-206
Complement of 2 of BIN 16-bit data (NEG).....	6-94	Conversion from BIN 32-bit to decimal ASCII (DBINDA).....	7-183
Complement of 2 of BIN 32-bit data (DNEG).....	6-94	Conversion from BIN 32-bit to floating decimal point (Double precision) (DFLTD).....	6-81
COMRD (Reading device comment data).....	7-201	Conversion from BIN 32-bit to floating decimal point (Single precision) (DFLT).....	6-78
Conditions for execution of instructions.....	3-33	Conversion from BIN 32-bit to hexadecimal ASCII (DBINHA).....	7-186
Connection instructions		Conversion from block BCD 4-digit data to block BIN 16-bit data (BKBIN).....	6-100
Association instruction table.....	2-7	Conversion from block BIN 16-bit data to BCD 4-digit data (BKBCD).....	6-98
Ladder block parallel connection (ORB).....	5-10	Conversion from character string to BIN 16-bit (VAL).....	7-212
Ladder block series connection (ANB).....	5-10	Conversion from character string to BIN 32-bit (DVAL).....	7-212
Linking character strings (\$+).....	6-65	Conversion from character string to floating decimal point (EVAL).....	7-224
Contact instruction.....	2-6	Conversion from decimal ASCII to BCD 4-digit (DABCD).....	7-198
Contact instructions		Conversion from decimal ASCII to BCD 8-digit (DDABCD).....	7-198
Operation start (LD, LDI).....	5-2	Conversion from decimal ASCII to BIN 16-bit (DABIN).....	7-192
Parallel connection (OR, ORI).....	5-2	Conversion from decimal ASCII to BIN 32-bit (DDABIN).....	7-192
Pulse operation start (LDF, LDP).....	5-5,5-7	Conversion from floating decimal point to character string (ESTR).....	7-217
Pulse parallel connection (ORF, ORP).....	5-5,5-7	Conversion from floating-point angle to radian (Double precision) (RADD).....	7-277
Pulse serial connection (ANF, ANP).....	5-5,5-7	Conversion from floating-point angle to radian (Single precision) (RAD).....	7-275
Series connection (AND, ANI).....	5-2	Conversion from floating-point radian to angle (Double precision) (DEGD).....	7-281,7-283,7-285
Conversion		Conversion from floating-point radian to angle (Single precision) (DEG).....	7-279
BCD 4-digit to BIN (BIN).....	6-75	Conversion from hexadecimal ASCII to BIN 16-bit (HABIN).....	7-195
BCD 8-digit to BIN (DBIN).....	6-75	Conversion from hexadecimal ASCII to BIN 32-bit (DHABIN).....	7-195
BIN 16-bit to BIN 32-bit (DBL).....	6-88		
BIN 16-bit to floating decimal point (Double precision) (FLTD).....	6-81		
BIN 16-bit to floating decimal point (Single precision) (FLT).....	6-78		
BIN 16-bit to Gray code (GRY).....	6-90		
BIN 32-bit to BIN 16-bit (WORD).....	6-89		
BIN 32-bit to floating decimal point (Double precision) (DFLTD).....	6-81		
BIN 32-bit to floating decimal point (Single precision) (DFLT).....	6-78		
BIN 32-bit to Gray code (DGRY).....	6-90		
BIN to BCD 4-digit (BCD).....	6-73		
BIN to BCD 8-digit (DBCD).....	6-73		
Double precision to Single precision (EDCON).....	6-104		
Floating decimal point data to BIN 16-bit (Double precision) (INTD).....	6-86		
Floating decimal point data to BIN 16-bit (Single precision) (INT).....	6-83		
Floating decimal point data to BIN 32-bit (Single precision) (DINT).....	6-83		
Floating decimal point data to BIN32-bit (Double precision) (DINTD).....	6-86		
Gray code to BIN 16-bit (GBIN).....	6-92		
Gray code to BIN 32-bit (DGBIN).....	6-92		
Single precision to Double precision (ECON).....	6-102		



Conversion from hexadecimal BIN to ASCII (ASC)	7-228	Data table operation instructions	2-40
Conversion of Gray code to BIN 16-bit (GBIN)	6-92	DATE- (Clock data subtraction operation)	7-350
Conversion of Gray code to BIN 32-bit (DGBIN)	6-92	Date comparison (DT=,DT>,DT=)	7-356
Conversion to BIN		DATE+ (Clock data addition operation)	7-348
BCD 4-digit to BIN 16-bit (BIN)	6-75	DATERD (Reading clock data)	7-344
BCD 8-digit to BIN 32-bit (DBIN)	6-75	DATEWR (Writing clock data)	7-346
Floating decimal point data to BIN 16-bit (Double precision) (INTD)	6-86	DB- (BCD 8-digit subtraction)	6-38
Floating decimal point data to BIN 16-bit (Single precision) (INT)	6-83	DB* (BCD 8-digit multiplication)	6-44
Floating decimal point data to BIN 32-bit (Double precision) (DINTD)	6-86	DB+ (BCD 8-digit addition)	6-38
Floating decimal point data to BIN 32-bit (Single precision) (DINT)	6-83	DB/ (BCD 8-digit division)	6-44
Conversion to floating decimal point (Double precision) (FLTD, DFLTD)	6-81	DBAND (32-bit dead band controls)	7-324
Conversion to floating decimal point (Single precision) (FLT, DFLT)	6-78	DBCD (Conversion from BIN to BCD 8-digit)	6-73
COS (COS operation on floating-point data (Single precision))	7-254	DBCDDA (Conversion from BCD 8-digit to decimal ASCII)	7-189
COS operation on floating-point data (Double precision) (COSD)	7-256	DBIN (BCD 8-digit to BIN 16-bit conversion)	6-75
COS operation on floating-point data (Single precision) (COS)	7-254	DBINDA (Conversion from BIN 32-bit to decimal ASCII)	7-183
COS <sup>-1</sup> operation on floating-point data (Double precision) (ACOSD)	7-269	DBINHA (Conversion from BIN 32-bit to hexadecimal ASCII)	7-186
COS <sup>-1</sup> operation on floating-point data (Single precision) (ACOS)	7-267	DBK-	6-63
COSD (COS operation on floating-point data (Double precision))	7-256	DBK+	6-62
Count 1-phase input or down (UDCNT1)	6-143	DBL (BIN 16-bit to BIN 32-bit)	6-88
Count 2-phase input or down (UDCNT2)	6-146	DCML (32-bit negation transfers)	6-114
Counters (OUT C)	5-26	DDABCD (Conversion from decimal ASCII to BCD 8-digit)	7-198
[D]		DDABIN (Conversion from decimal ASCII to BIN 32-bit)	7-192
D- (BIN 32-bit subtraction operations)	6-26	DDEC (Decrementing 32-bit BIN)	6-71
D(P).DDR(Reading Devices to Another CPU)	10-17	Debugging and failure diagnosis instructions	2-42
D(P).DDWR(Writing Devices to Another CPU)	10-13	DEC (Decrementing 16-bit BIN)	6-69
D* (BIN 32-bit multiplication operations)	6-32	DECO (Decoding from 8 to 256 bits)	7-71
D+ (BIN 32-bit addition operations)	6-26	Decoding from 8 to 256 bits (DECO)	7-71
D/ (BIN 32-bit division operations)	6-32	Decrement	
D=, D<>, D>, D<=, D<, D>= (BIN 32-bit data comparisons)	6-4	BIN 16-bit (DEC)	6-69
DABCD (Conversion from decimal ASCII to BCD 4-digit)	7-198	BIN 32-bit (DDEC)	6-71
DABIN (Conversion from decimal ASCII to BIN 16-bit)	7-192	Decrementing 16-bit BIN (DEC)	6-69
DAND (Logical products with 32-bit data)	7-3	Decrementing 32-bit BIN (DDEC)	6-71
Data control instructions	2-49	DEG (Conversion from floating-point radian to angle (Single precision))	7-279
Data conversion instruction table	2-22	DEGD (Conversion from floating-point radian to angle (Double precision))	7-281,7-283,7-285
Data conversion instructions	6-73	Deleting data from data tables (FDEL)	7-157
Data dissociation in byte units (WTOB)	7-85	Deletion of character string (STRDEL(P))	7-243
Data link instructions	2-59	DELTA (Pulse conversion of direct output)	5-42
Data linking in byte units (BTOW)	7-85	Designating data	3-3
Data processing instructions	2-35	Designation of modification values in index modification (IXDEV, IXSET)	7-148
		Device range check	3-27
		DFLT (Conversion from BIN 32-bit to floating decimal point (Single precision))	6-78
		DFLTD (Conversion from BIN 32-bit to floating decimal point (Double precision))	6-81
		DFRO (Reading 2-word data from intelligent function modules)	7-160
		DGBIN (Conversion of Gray code to BIN 16-bit)	6-92
		DGRY (BIN 32-bit to Gray code)	6-90
		DHABIN (Conversion from hexadecimal ASCII to BIN 32-bit)	7-195

DI (Interrupt disable).....	6-133	[E]	
Digit designation .....	3-4	E-	(Subtraction of floating decimal point data (Single precision))..... 6-46,6-48
Digit designation of bit devices .....	3-4	E*	(Multiplication of floating decimal point data (Single precision))..... 6-54
DINC (Incrementing 32-bit BIN).....	6-71	E+	(Addition of floating decimal point data (Single precision))..... 6-46,6-48
DINT (Floating decimal point data to BIN 32-bit (Single precision)).....	6-83	E/	(Division of floating decimal point data (Single precision))..... 6-54
DINTD (Floating decimal point data to BIN 32-bit (Double precision)) .....	6-86	E=, E<>, E>, E<=, E<, E>=	(Floating decimal point data comparisons(Single precision)) ..... 6-6
Direct 1-byte read from file register (ZRRDB)....	7-391	ECALL	(Sub-routine calls between program files) .....
DIS (4-bit grouping of 16-bit data) .....	7-77		7-120
Display instructions.....	2-41	ECON	(Single precision to Double precision conversion)..... 6-102
Dissociation of random data (NDIS) .....	7-81	ED-	(Subtraction of floating decimal point data (Double precision)) ..... 6-50,6-52
Division		ED*	(Multiplication of floating decimal point data (Double precision)) ..... 6-56
BCD 4-digit (B/).....	6-42	ED+	(Addition of floating decimal point data (Double precision))..... 6-50,6-52
BCD 8-digit division (DB/) .....	6-44	ED/	(Division of floating decimal point data (Double precision))..... 6-56
BIN 16-bit (/).....	6-30	ED=,ED<>,ED>,ED<=,ED<,ED>=	(Floating decimal point data comparisons (Double precision)) ..... 6-8
Division of floating decimal point(Double precision) (ED/)	6-56	EDCON	(Double precision to Single precision conversion)..... 6-104
Division of floating decimal point(Single precision) (E/)	6-54	EDMOV	(Floating-point data transfer (Double precision))..... 6-110
DLIMIT (Upper and lower limit controls for BIN 32-bit) .....	7-321	EDNEG	(Floating-point sign inversion (Double precision))..... 6-97
DMAX (Maximum value search for 32-bit data)...	7-89	EFCALL	(Output OFF calls between program files) .....
DMEAN(P).....	7-103		7-125
DMIN (Minimum value search for 32-bit data)....	7-92	EGF	(Pulse operation results / leading edge)..... 5-18
DMOV (32-bit transfers) .....	6-106	EGP	(Pulse operation results / trailing edge)..... 5-18
DNEG (Complement of 2 of BIN 32-bit data) .....	6-94	EI	(Interrupt enable)..... 6-133
DOR (Logical sums of 32-bit data) .....	7-11	EMOD	(Floating decimal point to BCD)..... 7-245
Double precision to Single precision conversion (EDCON) .....	6-104	EMOV	(Floating-point data transfer (Single precision)) .....
Double word data .....	3-6		6-108
DRCL (Left rotation of 32-bit data) .....	7-44	ENCO	(Encoding from 256 to 8 bits) .....
DRCR (Right rotation of 32-bit data) .....	7-41		7-73
DROL (Left rotation of 32-bit data) .....	7-44	Encoding	from 256 to 8 bits (ENCO) .....
DROR (Right rotation of 32-bit data) .....	7-41	END	(End sequence program) .....
DSCL(P) .....	7-331		5-53
DSCL2(P) .....	7-335	End main routine program (FEND) .....	5-51
DSER (32-bit data searches).....	7-66	End sequence program (END) .....	5-53
DSFL (1-word shift to left of n-word data)....	7-54,7-56	ENEG	(Floating-point sign inversion(Single precision)) .....
DSFR (1-word shift to right of n-word data) .....	7-54,7-56		6-96
DSORT (BIN 32-bit data sort).....	7-95	EREXP	(From BCD format data to floating decimal point)..... 7-248
DSTR (Conversion from BIN 32-bit to character string) .....	7-206	Error display and annunciator reset instruction (LEDR) .....	7-172
DSUM (32-bit data checks) .....	7-69		
DTEST (Bit tests).....	7-61	ESTR	(Conversion from floating decimal point to character string)..... 7-217
DTO (Writing 2-word data to intelligent function modules).....	7-163	EVAL	(Conversion from character string to floating decimal point) .....
DUTY (Timing pulse generation) .....	7-388		7-217
DVAL (Conversion from character string to BIN 32-bit) .....	7-212	EXP	(Exponent operation on floating-point data (Single precision))..... 7-291
DWSUM (Calculation of totals for 32-bit data) .....	7-101,7-103	Expansion clock data addition operation (S.DATE+)	..... 7-366
DXCH (16-bit data exchanges).....	6-124		
DXNR (32-bit data exclusive NOR operation) .....	7-27		
DXOR (32-bit exclusive OR operations).....	7-19		
DZONE (Zone control for BIN 32-bit data) .....	7-327,7-330,7-334		



[J]	
JMP (Pointer branch).....	6-129
Jump to END (GOEND).....	6-132
[K]	
KEY (Numerical key input from keyboard).....	7-396
[L]	
Ladder block parallel connections (ORB).....	5-10
Ladder block series connections (ANB).....	5-10
LD (\$=, \$<>, \$>, \$<=, \$<, \$>=) (Character string data comparisons).....	6-11
LD (=, <>, >, <=, <, >=) (BIN 16-bit data comparisons).....	6-2
LD (A contact operation start).....	5-2
LD (D=, D<>, D>, D<=, D<, D>=) (BIN 32-bit data comparisons).....	6-4
LD (E=, E<>, E>, E<=, E<, E>=) (Floating decimal point data comparisons(Single precision)).....	6-6
LD (ED=, ED, ED>, ED<=, ED<, ED>=) (Floating decimal point data comparisons(Double precision)).....	6-8
LDF (Pulse operation start / trailing edge).....	5-5,5-7
LDI (B contact operation start).....	5-2
LDP (Pulse operation start / leading edge).....	5-5,5-7
LDPI, LDFI.....	5-7
Leading edge output (PLS).....	5-37
LEDR (Error display and annunciator reset instruction).....	7-172
LEFT (Extracting character string data from the left).....	7-232
Left rotation of 16-bit data (ROL, RCL).....	7-38
Left rotation of 32-bit data (DROL, DRCL).....	7-44
LEN (Character string length detection).....	7-204
LIMIT (Upper and lower limit controls for BIN 16-bit).....	7-321
Link refresh instructions.....	2-59
Linking character strings (\$+).....	6-65,6-67
Linking of random data (NUNI).....	7-81
Load (LD).....	5-2
Load + unload (PSWAPP).....	7-445
Load inverse (LDI).....	5-2
Load program from Memory Card (PLOADP) ...	7-440
LOG (Natural logarithm operation on floating-point data (Single precision)).....	7-296,7-302
LOGD (Natural logarithm operation on floating-point data (Double precision)).....	7-298
Logical operation instructions.....	2-29
Logical product.....	7-2
Logical products with 16-bit data (WAND).....	7-3
Logical products with 32-bit data (DAND).....	7-3
Logical sum.....	7-2
Logical sums of 16-bit data (WOR).....	7-11
Logical sums of 32-bit data (DOR).....	7-11
Low speed retentive timer (OUTH ST).....	5-22
Low speed timer (OUT T).....	5-22

[M]	
Master control instructions.....	5-47
Matrix input (MTR).....	6-166
MAX (Maximum value search for 16-bit data).....	7-89
Maximum value search for 16-bit data (MAX).....	7-89
Maximum value search for 32-bit data (DMAX)...	7-89
MC (Setting the master control).....	5-47
MCR (Resetting the master control).....	5-47
MEAN(P).....	7-103
MEF (Pulse operation results / trailing edge).....	5-17
MEP (Pulse operation results / leading edge).....	5-17
MIDR (Random selection from character strings).....	7-235
MIDW (Random replacement in character strings).....	7-235
MIN (Minimum value search for 16-bit data).....	7-92
Minimum value search for 16-bit data (MIN).....	7-92
Minimum value search for 32-bit data (DMIN).....	7-92
MOV (16-bit transfers).....	6-106
MPP (Operation results pop).....	5-12
MPS (Operation results push).....	5-12
MRD (Operation results read).....	5-12
MTR (Matrix input).....	6-166
Multiplication	
BCD 4-digit (B*).....	6-42
BCD 8-digit (DB*).....	6-44
BIN 16-bit (*).....	6-30
BIN 32-bit (D*).....	6-32
Multiplication of floating decimal point (Double precision) (ED*).....	6-56
Multiplication of floating decimal point (Single precision) (E*).....	6-54
Multiplication and division of floating decimal point (Double precision)(ED*, ED/).....	6-56
Multiplication and division of floating decimal point (Single precision)(E*, E/).....	6-54
[N]	
Natural logarithm operation on floating-point data (Double precision) (LOGD).....	7-298
Natural logarithm operation on floating-point data (Single precision) (LOG).....	7-296,7-302
n-bit shift to left of 16-bit data (SFL).....	7-46
n-bit shift to right of 16-bit data (SFR).....	7-46
n-bit shift to right or left of n-bit data (SFTBR(P), SFTBL(P)).....	7-51
n-bit shift to right or left of n-word data (SFTWR(P), SFTWL(P)).....	7-56
NEG (complement of 2 of BIN 16-bit data).....	6-94
Network refresh instruction (ZCOM).....	8-2
NEXT (FOR to NEXT).....	7-105
No operation (NOP, NOPLF, PAGE).....	5-57
NOP.....	5-57
NOP (No operation).....	5-57
NOPLF (No operation page change).....	5-57
Number of steps.....	3-34
Numerical key input (KEY).....	7-396
Numerical key input from keyboard (KEY).....	7-396
NUNI (Linking of random data).....	7-81

[O]	
Operation errors	3-27
Operation results inversion (INV)	5-15
Operation results pop (MPP)	5-12
Operation results push (MPS)	5-12
Operation results read (MRD)	5-12
Operation start (LD, LDI)	5-2
OR (\$=, \$<>, \$>, \$<=, \$<, \$>=) (Character string data comparisons)	6-11
OR (=, <>, >, <=, <, >=) (BIN 16-bit data comparisons)	6-2
OR (A contact parallel connection)	5-2
OR (D=, D<>, D>, D<=, D<, D>=) (BIN 32-bit data comparisons)	6-4
OR (E=, E<>, E>, E<=, E<, E>=) (Floating decimal point data comparisons (Single precision))	6-6
OR (ED=, ED<>, ED>, ED<=, ED<, ED>=) (Floating decimal point data comparisons (Double precision))	6-8
Or inverse (ORI)	5-2
ORB (Ladder block parallel connections)	5-10
ORF (Pulse parallel connection / trailing edge)	5-5,5-7
ORI (B contact parallel connection)	5-2
ORP (Pulse parallel connection / leading edge)	5-5,5-7
ORPI, ORFI	5-8
Other convenient instructions	2-6
Other instructions	5-55
Application instructions	2-29
Sequence instructions	2-6
OUT	
Annunciator output (OUT F)	5-28
Counters (OUT C)	5-26
High speed retentive timer (OUTH ST)	5-22
High speed timer (OUTH T)	5-22
Low speed retentive timer (OUT ST)	5-22
Low speed timer (OUT T)	5-22
Output (OUT)	5-20
Out instructions (OUT)	5-20
Output instruction table	2-8
Output instructions (OUT)	5-20
Output of sub-routine program OFF calls (FCALL)	7-116
Output OFF calls between program files (EFCALL)	7-125
Output reverse (FF)	5-40

[P]	
PAGE (No operation page change)	5-57
Page change (NOPLF)	5-57
Page change (PAGE n)	5-57
Parallel connection (OR, ORI)	5-2
Parallel connections (ORB)	5-10
PCHK (Program low speed execution registration instruction)	7-384
PLF (Trailing edge output)	5-37
PLOADP (Load program from Memory Card)	7-440

PLOW (Program low speed execution registration)	7-382
PLS (Leading edge output)	5-37
PLSY (Fixed cycle pulse output)	6-162
POFF (Program output OFF standby instruction)	7-378
Pointer branching instruction (CJ, SCJ, JMP)	6-129
Pop (MPP)	5-12
PR (Print ASCII code instruction)	7-166
PRC (Print comment instruction)	7-169
Print ASCII code instruction (PR)	7-166
Print comment instruction (PRC)	7-169
Program branch instruction table	2-27
Program control instructions	2-56
Program execution control instruction table	2-27
Program low speed execution registration instruction (PCHK)	7-384
Program low speed execution registration (PLOW)	7-382
Program output OFF standby instruction (POFF)	7-378
Program scan execution registration instruction (PSCAN)	7-380
Program standby instruction (PSTOP)	7-377
PSCAN (Program scan execution registration instruction)	7-380
PSTOP (Program standby instruction)	7-377
PSWAPP (Load + unload)	7-445
Pulse conversion	
(DELTA)	5-42
(EGF, EGP)	5-18
(MEF, MEP)	5-17
Pulse conversion of direct output (DELTA)	5-42
Pulse density measurement (SPD)	6-160
Pulse NOT operation start, pulse NOT series connection, pulse NOT parallel connection LDPI, LDFI, ANDPI, ANDFI, ORPI, ORFI)	5-7
Pulse operation results	
Operation result conversions (MEF, MEP)	5-17
Pulse conversions of edge relay operation results (EGF, EGP)	5-18
Pulse operation start (LDF, LDP)	5-5,5-7
Pulse parallel connection (ORF, ORP)	5-5,5-7
Pulse series connection (ANDF, ANDP)	5-5
Pulse width modulation (PWM)	6-164
PUNLOADP (Unload program from program memory)	7-443
Push (MPS)	5-12
PWM (Pulse width modulation)	6-164

[Q]	
QCDSET (File setting for comments)	7-342
QCPU dedicated instructions	2-60
QDRSET (Setting files for file register use)	7-339

[R]	
RAD (Conversion from floating-point angle to radian (Single precision))	7-275

RADD (Conversion from floating-point angle to radian (Double precision)) .....	7-277	ROTC (Rotary table shortest direction control) .....	6-154
RAMP (Ramp signal) .....	6-157	RSET (Switching file register numbers) .....	7-337
Ramp signal (RAMP) .....	6-157	RST	
Random number generation (RND/SRND) .....	7-304	Resetting devices (RST) .....	5-32
Random selection from and replacement in character strings (MIDR) .....	7-235	Resetting the annunciators (RST F) .....	5-35
Random selection replacement in character strings (MIDW) .....	7-235	RTREAD (Reading routing information) .....	8-6
RBMOV (High-speed block transfer of file register) .....	7-448	RTWRITE (Writing routing information) .....	8-8
RCL (Left rotation of 16-bit data) .....	7-38		
RCR (Right rotation of 16-bit data) .....	7-35	[S]	
Read (MRD) .....	5-12	S.DATE- (Expansion clock data subtraction operation) .....	7-366
Read data from standard ROM (S.DEVLD) .....	7-438	S.DATE+ (Expansion clock data addition operation) .....	7-366
Reading 1-word data from intelligent function modules (FROM) .....	7-160	S.DATERD (Reading expansion clock data) .....	7-366
Reading 2-word data from intelligent function modules (DFRO) .....	7-160	S.DEVLD (Read data from standard ROM) .....	7-438
Reading clock data (DATERD) .....	7-344	S.TO (Write to host CPU shared memory) .....	9-4
Reading data from designated file (SP.FREAD) .....	7-424	Scaling (Point-by-point coordinate data) (SCL(P), DSCL(P)) .....	7-330
Reading device comment data (COMRD) .....	7-201	Scaling (Point-by-point coordinate data) (SCL2(P), DSCL2(P)) .....	7-334
Reading expansion clock data (S.DATERD) .....	7-366	SCJ (Pointer branching instruction) .....	6-129
Reading from other CPU shared memory (FROM) .....	9-12	SCL(P) .....	7-330
Reading module information (UNIRD) .....	7-402	SCL2 .....	7-334
Reading newest data from data tables (FPOP) .....	7-155	SECOND (Time data conversion) .....	7-352
Reading oldest data from data tables (FIFR) .....	7-153	SEG (7-segment decode) .....	7-75
Reading routing information (RTREAD) .....	8-6	Sequence instructions .....	2-6
Real number data .....	3-8	Sequence program stop (STOP) .....	5-55
Recovery from interrupt programs (IRET) .....	6-139	SER (16-bit data searches) .....	7-66
Refresh instruction (COM) .....	7-134	Series connection (AND, ANI) .....	5-2
Related programming manuals .....	1-2	Series connections (ANB) .....	5-10
Resetting devices (RST) .....	5-32,5-35	SET	
Resetting the annunciators (RST F) .....	5-35	Setting devices (SET) .....	5-30
Resetting the master control (MCR) .....	5-47	Setting the annunciators (SET F) .....	5-35
Resetting watchdog timer (WDT) .....	7-386	Setting devices (SET) .....	5-30,5-35
RET (Return from sub-routine programs) .....	7-115	Setting files for file register use (QDRSET) .....	7-339
Return from sub-routine programs (RET) .....	7-115	Setting the annunciators (SET F) .....	5-35
Revercing		Setting the master control (MC) .....	5-47
Bit device output reverse (FF) .....	5-40	SFL (n-bit shift to left of 16-bit data) .....	7-46
Floating-point sign inversion (Double precision) (EDNEG) .....	6-97	SFR (n-bit shift to right of 16-bit data) .....	7-46
Floating-point sign inversion (Single precision) (ENEG) .....	6-96	SFT (Bit device shifts) .....	5-44
Operation results inversion (INV) .....	5-15	SFTBL(P) .....	7-52
RFS (I/O refresh) .....	6-141	SFTBR(P) .....	7-51
RIGHT (Extracting character string data from the right) .....	7-232	SFTWL(P) .....	7-57
Right rotation of 16-bit data (ROR, RCR) .....	7-35	SFTWR(P) .....	7-56
Right rotation of 32-bit data (DROR, DRCL) .....	7-41	Shift instruction .....	5-44,7-46
RND (Random number generation and series update) .....	7-304	(Application instructions) .....	2-29
ROL (Left rotation of 16-bit data) .....	7-38	Shift instruction table	
ROR (Right rotation of 16-bit data) .....	7-35	(Sequence instructions) .....	2-6
Rotary table shortest direction control (ROTC) .....	6-154	SIN (SIN operation on floating-point data (Single precision)) .....	7-250
Rotation instructions .....	2-32	SIN operation on floating-point data (Double precision) (SIND) .....	7-252
		SIN operation on floating-point data (Single precision) (SIN) .....	7-250
		SIN <sup>-1</sup> operation on floating-point data (Double precision) (ASIND) .....	7-265
		SIN <sup>-1</sup> operation on floating-point data (Single precision) (ASIN) .....	7-262

SIND (SIN operation on floating-point data (Double precision)).....	7-252
Single precision to Double precision conversion (ECON).....	6-102
SORT (BIN 16-bit data sort).....	7-95
SP.CONTSW (System switching instruction).....	11-2
SP.DEVST (Writing data to standard ROM).....	7-436
SP.FREAD (Reading data from designated file).....	7-424
SP.FWRITE (Writing data to designated file)....	7-413
SPD (Pulse density measurement).....	6-160
Special format failure checks (CHKST, CHK) ...	7-175
Special function instructions.....	2-46
Special timer (STMR).....	6-151
SQR (Square root operation for floating-point data (Single precision)).....	7-287
SQRD (Square root operation for floating-point data (Double precision)).....	7-289
Square root operation for floating-point data (Double precision) (SQRD).....	7-289
Square root operation for floating-point data (Single precision) (SQR).....	7-287
SRND (Random number generation and series updates).....	7-304
STMR (Special function timer).....	6-151
STOP (Sequence program stop).....	5-55
STR (Conversion from BIN 16-bit to character string).....	7-206
Structure creation instructions.....	2-38
Subroutine program calls (CALL).....	7-110
Subroutine calls (XCALL).....	7-129
Subroutine calls between program files (ECALL).....	7-120
Subroutine program output OFF calls (FCALL).....	7-116
Subset processing.....	3-25
Subtraction	
BCD 4-digit subtraction (B-).....	6-34
BCD 8-digit subtraction (DB-).....	6-38
BIN 16-bit subtraction operations (-).....	6-22
BIN 32-bit subtraction operations (D-).....	6-26
Block subtraction (BK-).....	6-59,6-62
Subtraction of floating decimal point data (Double precision) (ED-).....	6-50,6-52
Subtraction of floating decimal point data (Single precision) (E-).....	6-46,6-48
SUM (16-bit data checks).....	7-69
SWAP (Upper and lower byte exchanges).....	6-128
Switching file register numbers (RSET).....	7-337
Switching instructions.....	2-51
System Switching (SP.CONTSW).....	11-2

## [T]

TAN (TAN operation on floating-point data (Single precision)).....	7-258
TAN operation on floating-point data (Double precision)(TAND).....	7-260
TAN operation on floating-point data (Single precision)(TAN).....	7-258

TAN <sup>-1</sup> operation on floating-point data (Double precision)(ATAND).....	7-273
TAN <sup>-1</sup> operation on floating-point data (Single precision)(ATAN).....	7-271
TAND (TAN operation on floating-point data (Double precision)).....	7-260
Teaching timer (TTMR).....	6-149
Termination instruction table.....	2-9
TEST (Bit tests).....	7-61
TIMCHK (Time check instruction).....	7-390
Time check instruction (TIMCHK).....	7-390
Time data conversion (HOUR).....	7-354,7-356,7-361
Time data conversion (SECOND).....	7-352
Timer (OUT T).....	5-22
Timing pulse generation (DUTY).....	7-388
TO (Writing 1-word data to intelligent function modules).....	7-163
TRACE (Trace set).....	7-411
TRACER (Trace reset).....	7-411
TTMR (Teaching timer).....	6-149
Types of Instructions.....	2-2

## [U]

UDCNT1 (Counter 1-phase input up or down) ..	6-143
UDCNT2 (Counter 2-phase input up or down) ..	6-146
UNI (4-bit linking of 16-bit data).....	7-79
UNIRD (Reading module information).....	7-402
Unload program from program memory (PUNLOADP).....	7-443
Up / Down counter	
Count 1-phase input or dawn (UDCNT1).....	6-143
Count 2-phase input or down (UDCNT2).....	6-146
Upper and lower byte exchanges (SWAP).....	6-128
Upper and lower limit controls for BIN 32-bit (DLIMIT).....	7-321

## [V]

VAL (Conversion from character string to BIN 16-bit).....	7-212
-----------------------------------------------------------	-------

## [W]

WAND (Logical products with 16-bit data).....	7-3
WDT (Resetting watchdog timer).....	7-386
WOR (Logical sums of 16-bit data).....	7-11
WORD (Conversion from BIN 32-bit to BIN 16-bit).....	6-89
Word data.....	3-4
Word device bit designation.....	3-3
Writing 1-word data to intelligent function modules (TO).....	7-163
Writing 2-word data to intelligent function modules (DTO).....	7-163
Writing clock data (DATEWR).....	7-346
Writing data to designated file (SP.FWRITE)....	7-413
Writing data to standard ROM (SP.DEVST).....	7-436
Writing data to the data tables (FIFW).....	7-151
Writing routing information (RTWRITE).....	8-8
Writing to the CPU shared memory of host CPU...	9-2

S.TO.....	9-4
TO.....	9-7
WSUM (Calculation of totals for 16-bit data) .....	7-99
WTOB (Data dissociation in byte units).....	7-85
WXNR (16-bit data exclusive NOR operation).....	7-27
WXNR (16-bit data non-exclusive logical sum operations).....	7-30
WXOR (16-bit exclusive OR operations) .....	7-19,7-22
[X]	
XCALL (Subroutine program call).....	7-129
XCH (32-bit data exchange) .....	6-124
[Z]	
ZCOM (Network refresh instruction).....	8-2
ZONE (Zone control for BIN 16-bit) .....	7-327,7-330,7-334
Zone control for BIN 16-bit (ZONE) .....	7-327,7-330,7-334
Zone control for BIN 32-bit data (DZONE) .....	7-327,7-330,7-334
ZPOP (Batch recovery of index register).....	7-400
ZPUSH (Batch save of index register).....	7-400
ZRRDB (Direct 1-byte read from file register)....	7-391
ZRWRB (File register direct 1-byte write).....	7-393



# **Warranty**

Please confirm the following product warranty details before using this product.

## **1. Gratis Warranty Term and Gratis Warranty Range**

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.

However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

The gratis warranty term of the product shall be for one year after the date of purchase or delivery to a designated place.

Note that after manufacture and shipment from Mitsubishi, the maximum distribution period shall be six (6) months, and the longest gratis warranty term after manufacturing shall be eighteen (18) months. The gratis warranty term of repair parts shall not exceed the gratis warranty term before repairs.

[Gratis Warranty Range]

- (1) The range shall be limited to normal use within the usage state, usage methods and usage environment, etc., which follow the conditions and precautions, etc., given in the instruction manual, user's manual and caution labels on the product.
- (2) Even within the gratis warranty term, repairs shall be charged for in the following cases.
  1. Failure occurring from inappropriate storage or handling, carelessness or negligence by the user. Failure caused by the user's hardware or software design.
  2. Failure caused by unapproved modifications, etc., to the product by the user.
  3. When the Mitsubishi product is assembled into a user's device, Failure that could have been avoided if functions or structures, judged as necessary in the legal safety measures the user's device is subject to or as necessary by industry standards, had been provided.
  4. Failure that could have been avoided if consumable parts (battery, backlight, fuse, etc.) designated in the instruction manual had been correctly serviced or replaced.
  5. Failure caused by external irresistible forces such as fires or abnormal voltages, and Failure caused by force majeure such as earthquakes, lightning, wind and water damage.
  6. Failure caused by reasons unpredictable by scientific technology standards at time of shipment from Mitsubishi.
  7. Any other failure found not to be the responsibility of Mitsubishi or that admitted not to be so by the user.

## **2. Onerous repair term after discontinuation of production**

- (1) Mitsubishi shall accept onerous product repairs for seven (7) years after production of the product is discontinued.

Discontinuation of production shall be notified with Mitsubishi Technical Bulletins, etc.
- (2) Product supply (including repair parts) is not available after production is discontinued.

## **3. Overseas service**

Overseas, repairs shall be accepted by Mitsubishi's local overseas FA Center. Note that the repair conditions at each FA Center may differ.

## **4. Exclusion of loss in opportunity and secondary loss from warranty liability**

Regardless of the gratis warranty term, Mitsubishi shall not be liable for compensation of damages caused by any cause found not to be the responsibility of Mitsubishi, loss in opportunity, lost profits incurred to the user by Failures of Mitsubishi products, special damages and secondary damages whether foreseeable or not, compensation for accidents, and compensation for damages to products other than Mitsubishi products, replacement by the user, maintenance of on-site equipment, start-up test run and other tasks.

## **5. Changes in product specifications**

The specifications given in the catalogs, manuals or technical documents are subject to change without prior notice.

## **6. Product application**

- (1) In using the Mitsubishi MELSEC programmable controller, the usage conditions shall be that the application will not lead to a major accident even if any problem or fault should occur in the programmable controller device, and that backup and fail-safe functions are systematically provided outside of the device for any problem or fault.
- (2) The Mitsubishi programmable controller has been designed and manufactured for applications in general industries, etc. Thus, applications in which the public could be affected such as in nuclear power plants and other power plants operated by respective power companies, and applications in which a special quality assurance system is required, such as for Railway companies or Public service purposes shall be excluded from the programmable controller applications.

In addition, applications in which human life or property that could be greatly affected, such as in aircraft, medical applications, incineration and fuel devices, manned transportation, equipment for recreation and amusement, and safety devices, shall also be excluded from the programmable controller range of applications.

However, in certain cases, some applications may be possible, providing the user consults their local Mitsubishi representative outlining the special requirements of the project, and providing that all parties concerned agree to the special circumstances, solely at the users discretion.

Microsoft, Windows, Windows NT are registered trademarks of Microsoft Corporation in the United States and other countries.

Pentium and Celeron are trademarks of Intel Corporation in the United States and other countries.

Ethernet is a trademark of Xerox Co., Ltd. in the United States.

CompactFlash is a trademark of SanDisk Corporation.

VxWorks, Tornado, WindPower, WindSh and WindView are registered trademarks of Wind River Systems, Inc.

Other company names and product names used in this document are trademarks or registered trademarks of respective owners.



# QCPU Programming Manual

## Common Instruction 2/2

MODEL	QCPU-P-KY-E
MODEL CODE	13JW10
SH(NA)-080809ENG(2/2)-C(0907)KWIX	

 **MITSUBISHI ELECTRIC CORPORATION**

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN  
NAGOYA WORKS : 1-14, YADA-MINAMI 5-CHOME, HIGASHI-KU, NAGOYA, JAPAN

When exported from Japan, this manual does not require application to the Ministry of Economy, Trade and Industry for service transaction permission.

Specifications subject to change without notice.