STARAN S APPLE Programming Manual

a new way of thinking

GOODYEAR AEROSPACE CORPORATION AKRON, OHIO 44315

GOODYEAR AEROSPACE CORPORATION

AKRON, OHIO 44315

STARAN S APPLE PROGRAMMING MANUAL

GER-15637

JUNE 1972

NOTICE

This document contains material generated by Goodyear Aerospace Corporation and is transmitted for the purpose of aiding the transaction of business between Goodyear Aerospace Corporation and the recipient. It is understood that the material contained herein will not be used, copied, or disclosed to others, without specific written consent of Goodyear Aerospace Corporation.

APPLE UPDATING

The Associative Processor Programming Language (APPLE) continues to be improved and expanded. Interested parties should contact Goodyear Aerospace Corporation, Computer Division Marketing, Akron, Ohio 44315, Telephone: (216) 794-3631 for information regarding the latest update of APPLE.

LIST OF EFFECTIVE PAGES

Insert latest changed pages and dispose of superseded pages.

NOTE: On a changed page, the portion of the text affected by the latest change is indicated by a vertical line in the outer margin of the page. Changes to illustrations are indicated by miniature pointing hands. A zero in the change number column indicates an original page.

The total number of pages in this manual is 247, consisting of the following:

Page No.	Change Number
	<u></u>
Title	0
A	0
i - vi	0
1-1 - 1-3	0
2-i, 2-1 - 2-163.	0
3-1 - 3-29	0
Ai, Al - A5	0
Bi, Bl	0
Ci, Cl _ C7	0
Di, D1 - D5	0
Ei, El - E4	0
Fi, Fl	0
Gi, G1 - G7	0
X-1 - X-6	0

NOTE -

- This document supersedes GER-15532 and GER-15635.

А

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
1	INTRODUCTION	1 - 1
	General	1 - 1
	APPLE	1-2
	One - To-One	1-2
	One-To-Many	1-2
	In Line	1-2
	Subroutine Call Sequence	1-2
	Assembler Directives	1-2
	Comment Statements	1-2
	APPLE Features	1-2
	QUICK INDEX OF APPLE INSTRUCTION GROUPS	‱2-i
2	APPLE LANGUAGE STRUCTURE	2-1
	Source Statements	2-1
	Label Field	2 - 1
	Command Field	2-1
	Argument Field	2 - 1
	Comment Field	2-3
	Required Entries	2-3
	Summary	2-3
	Language Elements	2-4
	Character Set	2-4
	Symbols	2-4
	Symbol Table	2-4
	Constants	2-5
	Octal Constants	2-5 2-5
	Hexadecimal Constants	2-5
	Expressions	2-6
	Examples	2-6
	Location Counters	2-6
	Load Location Counter	2-6 2-6 2-6
	Addressing.	2-7
	Control Memory Address	2-7 2-7
	$\begin{array}{cccccccccccccccccccccccccccccccccccc$	2-8 2-8
	Example 1	2-8 2-8
	Assembler Directives	2-9

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
2 (cont)	Branch Instructions	2-17
	Register Instructions	2-35
	Associative Instructions	2-55
	Loads	2-55
	Stores	2-79
	Searches	2 - 108
	Moves	2 - 125
	Arithmetics	2 - 138
	Control and Test	2-154
	Pager Instructions	2-159
3	SUPERVISOR CALLS	3-1
	Introduction	3-1
	Slot Numbers	3 - 1
	Device Assignment Table (DAT)	3-2
	Instruction Description	3-3
	BUFFER Pseudo-op Format	3-3
	Supervisor Call (SVC) Format	3-3
	Buffer	3-3
	Supervisor Call (SVC)	3-3
	SPS Services or Calls	3-4
	Attach	3-5
	Format	3-5
	Device Codes	3-5
	STARAN Special Device Codes For ATTACH Function	3-6
	STARAN Control Memory	3-6
	Format Buffer Format For Device -1 Example	3-6 3-7 3-8
	STARAN Associative Memory	3-8
	Format Buffer Format For Device -2	3-8 3-9
	Example	3-11

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
3 (cont)	STARAN Registers	3 - 12
	Format Buffer Format For Device -3 Example	3 - 12 3 - 13 3 - 14
	Read	3 - 15
	Format	3 - 15
	Example	3-15
	Write	3 - 16
	Format	3-16
	Example	3 - 16
	Read/Write BUFFER Pseudo-op	3-17
	Format	3-17
	Example	3-19
	Restart Program	3-21
	Reset Peripheral Devices	3-22
	Free Device For New Task	3-23
	Exit to Supervisor	3-24
	Timer Start	3-25
	Int - Signal Sequential Processor Interrupt	3-26
	Isetup - Setup Interrupt	3-27
	Pager Control	3-28
	PI/O Control	3-29

APPENDIX	TITLE	PAGE
A	SUMMARY OF APPLE MNEMONICS AND INSTRUCTION FORMATS	A-i
В	ERROR CODES	B-i
C ′	TERMS AND SYMBOLS	C-i
D	HEXADECIMAL/DECIMAL TABLE	D-i
E	OCTAL/DECIMAL	E-i
F	POWERS OF TWO TABLE	F-i
G	PROGRAM EXAMPLES	G-i
	INDEX	X - 1

LIST OF FIGURES

FIGURE	TITLE	PAGE
Frontispiece	STARAN S Computer System	vi
2-1	APPLE Assembler Coding Form	2-2
3 - 1	Device Assignment Table (DAT)	3-2

LIST OF TABLES

TABLE	TITLE	PAGE
2-1	Registers	2-36
2-2	Register Combinations	2-36

FOREWORD

GENERAL

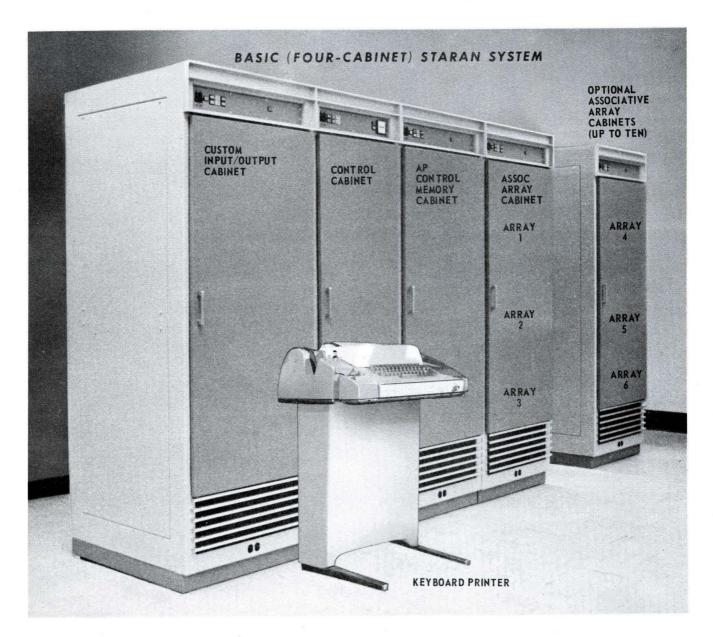
The APPLE Programming Manual is one of five standard manuals for STARAN S. As a composite group, the manuals provide the information necessary for programming, operating, and maintaining the standard STARAN S. The titles and publication numbers of the STARAN S manuals are as follows:

Title	Publication
STARAN S Reference Manual	GER-15636
STARAN S APPLE Programming Manual	GER-15637
STARAN S Operator's Guide	GER-15638
STARAN S Systems Programmer's Reference Manual	GER -15639
STARAN S Maintenance Manual	GER-15640

APPLE MANUAL

The APPLE Programming Manual is intended as a reference manual to guide the programmer in the use of the assembly language. The manual is written for the experienced programmer who has familiarized himself with the STARAN S Reference Manual, GER-15636.

CUSTOM INPUT/ OUTPUT Since the I/O cabinets are not standard units, but are customized for each particular installation, this manual includes no description of I/O mnemonics included in the APPLE language of a given installation.



STARAN S COMPUTER SYSTEM

CHAPTER 1

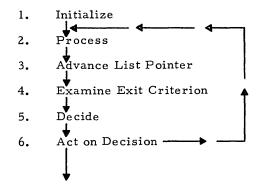
INTRODUCTION

GENERAL

The Goodyear Aerospace Corporation (GAC) Associative Processor, STARAN S*, is a new digital computer system differing significantly from conventional digital computers.

The Associative Processor (AP) is a general-purpose computer capable of performing search, arithmetic, logic, and store operations simultaneously on many independent sets of data. This capability, which is a feature unique to STARAN S, results in certain major differences between programming techniques for STARAN S and those for conventional machines.

As an example, consider the familiar "loop" programming concept. A loop is defined as a set of commands repeatedly and consecutively executed on different sets of data. Conventional programming of a loop involves the following steps:



To process a new set of data conventionally requires execution of the complete loop, including steps 3, 4, 5, and 6, as coding and execution time overhead.

In an AP, execution of the equivalent of a loop on associative items requires initialization and a single pass through the process step. There is no need to advance a list pointer to reference the next set of data to be processed, to determine when to exit from the loop, or to repeatedly execute the process step. The loop is one of many examples of program simplification and improved execution time possible with an AP.

*TM. Goodyear Aerospace Corporation, Akron, Ohio 44315

APPLE	Development of a new digital machine organization involves the design of a programming language suitable for the computer.
	APPLE is the acronym for the Associative Processor Programming LanguagE. APPLE is a machine-oriented symbolic language designed to expedite programming for the STARAN S system.
	APPLE mnemonics produce four basic types of assembler generated output:
	1) One-to-One Translation
	2) One-to-Many Translation
	3) Assembler Directives
	4) Comment Statements
ONE-TO-ONE	Most assembler level languages for conventional computers generate one machine language instruction per mnemonic. Many of the basic APPLE mnemonics fall into this category.
ONE-TO-MANY	Several APPLE mnemonics are in the one-to-many category. Many basic AP programming functions require more than one machine language instruction per mnemonic. Some of these mnemonics produce in-line machine instructions; others generate a subroutine call to a sequence of machine instructions.
In Line	The one-to-many mnemonics producing in-line machine instructions are equivalent to macro instructions of higher level assembly languages.
Subroutine Call Sequence	A library of subroutines is provided by APPLE and resides in Page 0 memory. The one-to-many mnemonics produce in-line subroutine call sequences similar to the linkages provided in FORTRAN to the SIN or TAN functions of a FORTRAN library.
ASSEMBLER DIRECTIVES	Assembler directive statements provide functions that assist the programmer in controlling the assignment of storage addresses, defining data and storage fields, and controlling the APPLE system itself. With a few exceptions, assembler directive statements do not generate machine language code.
COMMENT STATEMENTS	Comment statements may appear anywhere in the program and will be printed on the listing device. However, comment statements have no effect on the object code produced.

APPLE FEATURES APPLE is essentially a symbolic assembly language. All AP memories and registers may be referenced symbolically.

Constants can be expressed as decimal, octal, or hexadecimal numbers in source statements. Addresses can be expressed absolutely or symbolically.

A listing of the source program statements, the resulting machine language code, and a symbol table may be produced by APPLE for each program. When a source program is assembled, an extensive syntactical check is provided by APPLE. Detected errors are printed on the program listing in error codes (Appendix B) at the left-hand margin of the particular statement in error. A maximum of two error codes can be printed for each statement.

QUICK INDEX

APPLE INSTRUCTION GROUPS

ASSEMBLER DIRECTIVES BRANCH INSTRUCTIONS REGISTER INSTRUCTIONS ASSOCIATIVE INSTRUCTIONS _____ Loads_____ Stores _____ Searches _____ Moves _____ Arithmetics _____ CONTROL and TEST _____ PAGER INSTRUCTIONS _____

CHAPTER 2

APPLE LANGUAGE STRUCTURE

The source statement is the basic component of an APPLE program. STATEMENTS Source statements consist of the following four entries: Label, Command Argument, and Comment. APPLE accepts source statements in free format. Blanks act as field delimiters. The suggested coding form for source statements is shown in figure 2-1. The columns on the coding form correspond to those of a standard 80-column Hollerith coded card. One line of coding on the form corresponds to one source card. Columns 1 through 72, inclusive, constitute the active line. Columns 73 through 80 are ignored by APPLE except for listing purposes. The source statement may be continued past 72 columns by inserting a semicolon (;), which, when scanned, terminates the present active

line. APPLE then searches the next active line to complete the

source statement.

Label Field.

LABEL FIELD

SOURCE

The Label Field is usually an optional symbol created by the programmer to identify the statement line. The symbol may consist of nine characters or less, with the first character in column one. If the first column is blank, the Label entry is assumed omitted. The symbol in the Label Field can contain alphabetics (A-Z) or numerics (0-9); however, at least one of the characters must be an alphabetic. The Label Field entry may have the same configuration as predefined mnemonics without conflict, since APPLE distinguishes through context which usage is intended. Only one entry is permitted in the

COMMAND FIELD

The Command Field is a requirement. It may consist of several symbols separated by commas (,). The first symbol is the predefined mnemonic (Appendix A) for a particular command. Command modifiers may follow the command, depending upon the individual command. No embedded blanks are allowed in the Command Field.

ARGUMENT FIELD

Entries in the Argument Field properly specify the instruction. In general, the purpose of this field is to identify the source and destination locations to the command. Other entries, such as Control Digits, are also included in this field. The entries are separated by commas and no embedded blanks are allowed. APPLE assumes no Argument Field entries if 16 contiguous blanks follow the Command

2 - 1

L-1751(5-72)

*APPLE ASSEMBLER CODING FORM

PROGRAM _____

PAGE _____ 0F _____

PROGRAMMER

DATE _____

*THIS IS A SUGGESTED FORM TO FACILITATE READING SOURCE LISTINGS, THE "APPLE" ASSEMBLER WILL ACCEPT FREE FORM WITH A SPACE TERMINATING EACH FIELD.

	LABEL	COMMAND	ARGUMENT	COMMENTS	ID/SEQ	
	1 2 3 4 5 6 7 8 9 1	0 11121314151617161	9 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 36	0 <mark>40 414243444546 47 48 4950 51525354555657 585960 616263646566676869707172</mark>	7 3 747576 77 78 79 80	
1			+			
2	· · · · · · · · · · · · · · · · · · ·					
з						
4						
5			╺┽╌┸╌┚╍┶┺┖┥╌┨╌┨╌┨╸┪╸	<u></u>		
6						
7						
8						
9						
10		╾┸╶┠╼┸╼╿╼╿╴┹╶╢╼╸		<u> </u>		
11						
12		╶┧╾┦╾┨╼╞╶┨╶┱╸┨╌┑				
13						
14						
15				╶╶╦┉┶╦┶╌┹╺┶┥┥╌┚╌╌┙┥╌┚╶┚╼┸╌╿┥╶┚╾╹╴┛╸┛╴┚╶┥╴┙╸		
16		┊	┉┥╾╌╧┙╧╧╴╴╴╵╌╘╡╌╌╌┶╶╴╧╶╴╴			
17	<u></u>					
18	·····		· · · · · · · · · · · · · · · · · · ·			
19			· · · · · · · · · · · · · · · · · · ·	· · · · · · · · · · · · · · · · · · ·		
20						

Figure 2-1. APPLE Assembler Coding Form

ARGUMENTField. Symbols appearing in the Argument Field must be defined toFIELDthe program, either by being predefined by APPLE or by appearing(cont)in the Label Field of a source statement.

COMMENT FIELD Comments are descriptive items of information that may be included on the program listing. Comment entries consist of any information the programmer wishes to record. All valid characters, including blanks, can be used. The Comment Field begins one blank after the Argument Field, or if no Argument Field exists, comments begin after 16 contiguous blanks follow the Command Field. An asterisk (*) in column one indicates the entire source statement is a comment.

REQUIRED ENTRIES

Required entries for the various mnemonics are underlined in the Format description of each instruction discussion (i.e., $\underline{B} = a(r)\pm k, cd$).

SUMMARY

- APPLE interprets the fields from left to right: Label, Command, Argument, Comment.
- A blank column terminates any field except the Comment Field, which is terminated at column 80.
- One or more blanks at the beginning of a line indicates there is no Label Field entry.
- 4) The Label Field entry, when present, must begin in column 1.
- 5) The Command Field begins with the first nonblank column following the Label Field or in the first nonblank column following column 1, if the Label Field is omitted.
- 6) The Argument Field begins with the first nonblank column following the Command Field. An Argument Field is designated as being blank in either of two ways:
 - a. Sixteen or more blank columns follow the Command Field.
 - b. The end of the active line (column 72) is encountered and continuation is not indicated.
- 7) The Comment Field begins in the first nonblank column following the Argument Field, or when the Argument Field is omitted, at least 16 blank columns following the Command Field.

CHARACTER SET APPLE language statements are written using the following alphabetics, numerics, operators, and delimiters:

Alphabetics A through Z Numerics 0 through 9 Operators \$ + - * = Delimiters , () BLANK ' ;

Each character is represented by an 8-bit byte. Only 47 characters of the set of 256 code combinations defined as the Extended Binary Coded Decimal Interchange Code (EBCDIC) are included in APPLE's character set. Most of the terms used in APPLE source statements are expressed in the character set shown above; however, language features, such as comments, permit the use of any of the 256 EBCDIC codes.

SYMBOLS

Symbols are formed from combinations of characters. Symbols provide programmers with a convenient means of identifying program elements so that they can be referred to by other elements. Symbols must conform to the following rules:

- 1) Symbols consist of 1 to 9 alphanumeric characters.
- 2) At least one character in a symbol must be alphabetic.
- 3) No special characters or embedded blanks can appear in a symbol.
- A symbol may be defined only once. If duplicate symbols occur they will be flagged as errors.

Symbols provide the most commonly used means of addressing source statements, constants, and storage locations. Symbols are normally defined in the Label Field of a source statement. After a symbol has been defined, it can be referred to by Argument Field entries. The value of a symbol can be equated to an absolute value (see EQU, DF in the Assembler Directives discussion)

Symbol Table

APPLE compiles a table containing all the symbols that appear in the Label Field and the addresses at which they appear. References to symbols cause APPLE to interrogate the symbol table for the address associated with the symbol. CONSTANTS

A constant is a self-defining language element whose value is explicit. Self-defining terms are useful in constants requiring a value rather than the symbolic address of the location where that value is stored. Three constant notations are used in APPLE instructions: octal, decimal, and hexadecimal.

Octal Constants An octal constant consists of a signed octal number enclosed by single quotation marks and preceded by the letter O.

The constant is right-justified in its field. For example,

Constant	ant Binary Value		Hexadecimal Value					
O' 1234 '	001	010	011	100	0010	1001	1100	(29C)

The octal digits and their binary equivalents are as follows:

0	-	000	4	-	100
1	-	001	5	-	101
2	-	010	6	-	110
3	-	011	7	-	111

Decimal Constants

A decimal constant consists of an integer (no decimal point) that may be signed. For example, 100 or -5423.

Hexadecimal Constants A hexadecimal constant consists of a signed hexadecimal number enclosed by single quotation marks and preceded by the letter X. For example,

X'9C01F' X'C0FFEE'

The assembler generates four binary bits of storage for each hexadecimal digit. The hexadecimal digits and their binary equivalents are as follows:

X'FFFF'

0	-	0000	8 - 1000
1	-	0001	9 - 1001
2	-	0010	A - 1010
3	-	0011	B - 1011
4	-	0100	C - 1100
5	-	0101	D - 1101
6	-	0110	E_1110
7	-	0111	F - 1111

EXPRESSIONS

Argument Field entries consist of either single-term expressions or double-term expressions. Single-term expressions are symbols, constants, or Location Counter references (\$). Double-term expressions are two single terms connected with an arithmetic operator. The valid arithmetic operators are a plus sign (+) for addition and a minus sign (-) for subtraction. The first single-term expression of a double-term expression may be a symbol or constant, and the second single-term expression must be a constant.

Examples

ValidInvalidTAG+5TAG-LABELLABEL-235+TAG5+32TAG+5+23

LOCATION COUNTERS

APPLE maintains two internal Location Counters: a Load Location Counter and an Execution Location Counter. The Load Location Counter keeps track of the addresses associated with the instructions when the program is loaded. The Execution Counter keeps track of the addresses associated with the instructions when they are executed.

The Load Location Counter keeps track of the addresses associated with the instructions when they are loaded.

As each instruction or data area is assembled, the Load Location Counter is incremented by the length of the assembled item. Therefore, the Load Location Counter is the address of the next available storage location in Control Memory after the instruction is assembled. This address is the location where the instruction will reside after being loaded.

As each instruction or data area is assembled, the Execution Location Counter is incremented by the length of the assembled item. The Execution Location Counter differs from the Load Location Counter when Pager commands are encountered. (See Pager Instructions.) Each STRTSG that appears in an assembly reinitializes the Execution Location Counter. This address is the location where the instruction will reside when executed.

The special symbol, \$ (dollar sign), is predefined by APPLE as Location Counters. The \$ may be used to alter the Location Counters at assembly time (see ORG in Assembler Directives Discussion). The \$ may also be used in an absolute expression to refer to an address. In this context it is the Execution Location Counter that forms the address.

Load Location Counter

Execution Location Counter

Location Counter Symbol (\$)

ADDRESSING

Control Memory Address The Control Memory Address is a symbolic or absolute address in bulk core, page memory, or High Speed Data Buffer. A Control Memory Address expression is comprised of four terms in the form $a(r)\pm k,cd$. Note that required terms are underlined.

- a This entry is the only one required. This term may be a symbol or a constant.
- k This entry must be a constant. At assembly time $\pm k$ is added to the value of 'a' to form the address.

r - This entry must be one of the following registers:R0 through R7, DP.

At execution time the contents of this specified register is added to the value $a\pm k$. It is this result that defines the Control Memory Address. The contents of the register can be considered to be the base address, and the double-term expression $a\pm k$ can be considered to be the displacement.

cd - This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an a±k type of expression, where 'a' and 'k' are defined as above.

cd Values	Action
1	Decrement BL
2	Increment DP
3	Decrement BL and Increment DP
4	Decrement DP
5	Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

Associative Memory or Common Register Field Expression A field expression defines the most significant bit position and the number of contiguous bit positions (field length) occupied by a field. There are two ways of constructing a field expression: Associative Memory or Common Register Field Expression (cont)

• 1

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most significant bit position and the number of contiguous bits occupied by a field in either the Common register or Associative Memory. The optional constant modifier, i, modifies only the most significant bit position.

(b,i)±j

where b may be a constant or a symbol and represents the most significant bit position of a field. If b was defined as a field via a previous DF instruction, the most significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant modifying only the most significant bit position of the field.

• Example 1	AJAX	DF	10,3
		•	
		•	
		sc	AJAX-3,(100,3)

The field AJAX begins in bit column 10 and spans 3 bit columns (bit columns 10, 11, 12).

The expression AJAX-3 has modified the most significant bit position to a value of 7 and spans 3 bit columns (bit columns 7, 8, 9).

The expression (100, 3) defines a field beginning with bit column 100 and spans 3 bit columns (bit columns 100, 101, 102).

The field FIELD1 begins in bit column 0 and spans 5 bit columns (bit columns 0, 1, 2, 3, 4).

The expression (FIELD1, 17)+O' 17+ has modified the most significant bit position to a value of 15, and has also modified the number of bit columns to 17 (bit columns 15, 16, ..., 31).

The expression (X'80', O'21') defines a field beginning with bit column 128 and spans 17 bit columns (bit columns 128, 129,..., 144).

2-8

ASSEMBLER DIRECTIVES

Assembler directive statements provide auxiliary functions to APPLE and assist the programmer in checking, documenting, and organizing a program.

The assembler directives are:

Mnemonic	Instruction
START	Start APPLE
END	End APPLE
ORG	Initialize Location Counter
EQU	Equate
DF	Define a Field
DS	Define Storage
TOF	Top of Form
EVEN	Make Location Counter Even
DC	Define Constant
GEN	Generate Machine Instructions
NOP	No Operation
A or E	Character String Generator

	This instruction performs initializing functions for APPLE, and generates pertinent header information for all object programs. This instruction is required and should be the first source statement in all APPLE programs.				
Format	Label	Command	Argument	Comment	
	symbol	<u>START</u>			
• Label	Any valid s	symbol or blank.			
• Command	START				
• Argument	No entry required.				

END

START

End APPLE

Start APPLE

This instruction will process and assemble all previous source program statements. The END instruction is required and must be the last source statement of every assembly.

Format	Label	Command	Argument	Comment	
	symbol	END	a±k		
• Label	Any valid symbol or blank.				
• Command	END				
• Argument	An optional entry.				
• • a±k	'a' may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k. This term represents an address designating where program execution will start immediately after the object program is loaded.				
• • blank	not automa	tically begin exec CND - statement s	ution upon compl	ld, this program will etion of loading. In this mbler the end of the	

2-10

Initialize Location Counter

This instruction commands the assembler to assemble succeeding instructions beginning at the address specified in the Argument Field. The Load Location Counter and Execution Location Counter are loaded with the value of $a\pm k$.

Format	Label	Command	Argument	Comment
	symbol	ORG	<u>a</u> ±k	

• Label Any valid symbol or blank.

ORG

Command

• • a±k

ORG

Argument

One entry is required.

'a' may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k. Moreover 'a' may be one of the following special predefined symbols provided for ease of programming:

Definition
Page 0 Memory Starting Address
Page 1 Memory Starting Address
Page 2 Memory Starting Address
High-Speed Data Buffer Memory Starting Address
Direct Memory Access Memory Starting Address
Bulk Core Storage Memory Starting Address

BULKC+16

ORG

Example

• Note

In this example the first instruction following the ORG statement will be assigned the Bulk core address X'8020' (BULKC assigns the address X'8010' in the APPLE assembler).

EQU

Equate

EQU

This instruction permits the programmer to assign a value to a symbol. Whenever the symbol appears in a succeeding instruction, the equated value will be used to form the machine language code.

Format	Label	Command	Argument	Comment
	symbol	EQU	<u>a</u> ±k	

Any valid symbol. This entry is required.

Command

● ● a±k

• Label

'a' may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k. 'a' may also be one of the special predefined APPLE symbols such as register abbreviations (Table 2-1) PAGE0, PAGE1, PAGE2, HSDB, DMA, BULKC, X, Y, and M. However, if a special symbol is used it cannot be modified by k.

\mathtt{DF}

Define a Field

This instruction permits the programmer to assign a field definition value to a symbol for later use. Whenever the symbol appears in instructions, the defined field value will be used to form the machine language code.

Label	Command	Argument	Comment
symbol	DF	$\underline{a}_{1\pm k_{1},\underline{a}_{2\pm k_{2}}}$	

Any valid symbol. This entry is required.

Label

Format

Command

• Argument Two entries are required.

 \mathbf{DF}

• • a1^{±k}1, a2^{±k}2

'a' may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k. The value of the term $a_1 \pm k_1$ represents the most significant bit position of the field being defined. The value of the term $a_2 \pm k_2$ represents the number of contiguous bit positions (field length) occupied by the field being defined.

The sum of $a_1 \pm k_1$ or $a_2 \pm k_2$ must not exceed the total number of bits in an associative memory word (0 to 255). If the field being defined is a field in the Common register, the sum of $a_1 \pm k_1$ or $a_2 \pm k_2$ should not exceed the number of bits in the Common register (0 to 31).

Note

Define Storage

This assembler directive will allocate the next specified number of 32 bit words as a contiguous block of control memory.

Format	Label	Command	Argument	Comment
	symbol	<u>DS</u> ,a±k		
• Label Any valid s		ymbol or blank.		
• Command	DS			
• • a±k	'a' may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k. The value of the term $a\pm k$ specifies the number of contiguous words to be reserved. If this entry is omitted, a default value of one is assumed.			
• Argument	Blank			

TOF Top of Form

This assembler directive will issue a form feed to the assembly listing device. TOF may be placed anywhere in the program and has no effect on the object code produced.

Format	Label	Command	Argument	Comment	
		TOF			
• Label	Must be bla	ank.			
• Command	TOF				
• Argument	None required.				
• Comment	The commo	ent will be printed	at the top of the p	age after the form feed.	

DS

EVEN

Make Location Counter Even

If the Execution Location Counter is odd when this instruction is encountered, an NOP will be produced in the object code; otherwise, no object code will be produced. Therefore, after this instruction has been processed the Execution Location Counter will be even. (Ref. SPSW instruction.)

Format	Label	Command	Argument	Comment		
	symbol	EVEN				
• Label	Any valid symbol or blank.					
• Command	EVEN					
• Argument	None required.					

Define C

Define Constant

This instruction will generate a specified value for a specified number of 32 bit control memory words.

Format	Label	Command	Argument	Comment
	symbol	DC,a1 ^{±k} 1	$\underline{a}_2^{\pm k}_2$	

• Label Any valid symbol or blank.

DC

Command

• • a₁±k₁

DC

 a_1 may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k_1 . The value of the term $a_1 \pm k_1$ specifies the number of contiguous 32-bit words. If this entry is omitted, a value of one is assumed.

Argument

• • a₂±k₂

 a_2 may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant k₂. The value of the term $a_2\pm k_2$ is the value to be inserted in each of the 32-bit words.

Generate Machine Instructions

Any valid symbol or blank.

GEN

This instruction permits the programmer to generate machine codes for instructions not covered by APPLE. (See STARAN S Reference Manual for detailed machine language coding.) This instruction is also useful when generating words of data rather than instructions.

Label	Command	Argument	Comment
symbol	$\underline{\text{GEN}, k_1}, \dots, k_n$	$\underline{a}_1 \pm j_1, \dots, a_n \pm j_n$	

• Label

Format

GEN

Command

• • k_i

Argument

•• a_i±j_i

Note

 a_i may be either a symbol or a constant whose value may be optionally modified by plus or minus the constant j_i . These term(s) represent the value(s) to be inserted into each of the corresponding data field(s). There must be a one-to-one correspondence between the k_i and $a_i \pm j_i$ terms.

One or more constants that define the length of the consecutive data fields $a_i \pm j_i$ respectively. The sum of all the k's must be less than or equal to 32.

If the sum of the lengths of the data fields is less than 32, the information will be right-justified in the word.

NOP	No Operat	ion		
	This instr	uction performs	no operation whe	en it is executed
Format	Label	Command	Argument	Comment
	symbol	NOP		
• Label	Any valid	symbol or blank.	·	
• Command	NOP	•		
• Argument	No entry 1	equired.		

A or E

Character String Generator

These two assembler directives enable the programmer to generate messages for output.

Format	Label	Command	Argument	Comment		
	symbol $\underline{Axc_1c_2c_{i-1}c_ix}$					
or	Label	Command	Argument	Comment		
	symbol	$\underline{\operatorname{Exc}_{1}}^{c_{2}} \cdots ^{c_{i-1}} ^{c_{i}} \underline{x}$				
• Label	Any valid	symbol or blank.				
• Command	A charact	er string entry is requi	red.			
• • A	-	esents an assembler dir the seven bit ASCII code		ling the assembler to ne succeeding character		
• • E	E represents an assembler directive commanding the assembler to generate the eight bit EBCDIC code equivalent to the succeeding character string.					
• • x	x must be any non-alphanumeric character and serves as the "begin" and "end" marker of the character string $c_1c_2c_{i-1}c_i$. x cannot be a ';'.					
• • c ₁ c ₂ c _{i-1} c _i	The c _i may be any allowable ASCII or EBCDIC character (except the ';') depending on whether A or E is used respectively. One or more full thirty- two bit words are generated with the ASCII or EBCDIC code of the c _i packed on byte boundaries at four characters per word. If there are not enough characters to generate a full word, the remaining bytes will be padded with blank (or space) characters.					
• Argument	No entry is required.					
Note		no provision for continu ource cards.	ation of a chara	cter string onto		
	A ';' character can be used in the text of the character string if there are enough blank delimiters preceding it so that it would fall in the comment field when parsed according to the free format rules of an ordinary source statement.					

2-16

BRANCH INSTRUCTIONS

Branch instructions alter the execution sequence of a program if certain conditions exist.

The branch instructions are:

Mnemonic	Instruction
в	Unconditional Branch
BZ	Branch if Zero
BNZ	Branch if Not Zero
BBS	Branch if Bit Set
BBZ	Branch if Bit Zero
BRS	Branch if Response
BNR	Branch if No Response
BOV	Branch if Overflow
BNOV	Branch if No Overflow
BAL	Branch and Link
RPT	Repeat
LOOP	Loop

, бала С Unconditional Branch

This instruction will transfer control from the current program address to the address specified in the Argument Field.

Format	Label	Command	Argument	Comment			
	symbol	B	<u>a(r)±k, cd</u>				
• Label	Any valid s	ymbol or blank					
• Command	В						
• Argument	Core, Page	The Control Memory Address is a symbolic or absolute address in Bulk Core, Page Memory, or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form $a(r)\pm k$, cd.					
• • a		is required only her a symbol or a		n (r) is omitted. This term			
• • k	This option	al term must be	a constant and mod	lifies 'a'.			
• • r	This entry may be one of the following nine registers: R0 through R7, DP. The contents of this specified register is added to the value $a\pm k$ at execution time. It is this result that defines the Control Memory Address. The contents of the register can be considered the base address, and the $a\pm k$ expression can be considered the displacement.						
• • cd	This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an a±k type of expression where 'a' and k are as defined above.						
	cd Values 1 2 3 4 5 The Contro	Incre Decr Decr Decr	Action ement BL ment DP ement BL and incre ement DP ement BL and DP	ement DP ne base register option has			

been selected, and the register forming the base register is the DP register.

в

2-18

Branch if Zero

This instruction will transfer control from the current program address to the address specified in the argument field, if the command field register, r_1 , is zero.

Format	Label	Command	Argument	Comment
	symbol	<u>BZ, r</u> 1	<u>a(r</u> 2)±k, cd	

Label

ΒZ

Any valid symbol or blank.

ΒZ

• Command

• • ^r1

Register	Definition
FP1	Field Pointer 1 (8 bits)
FP2	Field Pointer 2 (8 bits)
FP3	Field Pointer 3 (8 bits)
FL1	Field Length Counter 1 (8 bits)
FL2	Field Length Counter 2 (8 bits)
FPE	Field Pointer E (8 bits)
BL	Block Length Counter (16 bits)
DP	Data Pointer Register (16 bits)

Argument

а

• k

r₂

The Control Memory Address is a symbolic or absolute address in Bulk Core, Page Memory, or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form $a(r_2)\pm k$, cd.

This entry is required only if the optional term (r_2) is omitted. This term may be either a symbol or a constant.

This optional term must be a constant and modifies 'a'.

This entry must be one of the following nine registers: R0 through R7, DP. The contents of the specified register is added to the value $a\pm k$ at execution time. The result defines the Control Memory Address. The contents of the register can be considered the base address, and the $a\pm k$ expression can be considered the displacement.

ΒZ

• • cd

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an $a\pm k$ type of expression, where 'a' and k are as defined above.

cd Values	Action
1	Decrement BL
2	Increment DP
3	Decrement Bl and increment DP
4	Decrement DP
5	Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

Branch if Not Zero

This instruction will transfer control from the current program address to the address specified in the argument field, if the command field register, r_1 , is not zero.

Format	Label	Command	Argument	Comment
	symbol	BNZ,r1	<u>a</u> (r ₂)±k, cd	

- Label Any valid symbol or blank.
- Command BNZ
 - rl Register Definition FPl Field Pointer 1 (8 bits) Field Pointer 2 (8 bits) FP2FP3 Field Pointer 3 (8 bits) FLI Field Length Counter 1 (8 bits) Field Length Counter 2 (8 bits) FL2 FPE Field Pointer E (8 bits) Block Length Counter (16 bits) BLData Pointer Register (16 bits) DP

Argument

BNZ

The Control Memory Address is a symbolic or absolute address in Bulk Core, Page Memory, or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form $a(r_2)\pm k$, cd.

- • a This entry is required only if the optional term (r_2) is omitted. This term may be either a symbol or a constant.
- •• k

This optional term must be a constant and modifies 'a'.

• • ^r2

This entry may be one of the following nine registers: R0 through R7, DP. The contents of the specified register is added to the value a±k at execution time. It is this result that defines the Control Memory Address. The contents of the register can be considered the base address, and the a±k expression can be considered the displacement.

• • cd

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an $a\pm k$ type of expression, where 'a' and k are defined as above.

BN Z

• • cd (cont) cd ValuesAction1Decrement BL2Increment DP3Decrement BL and increment DP4Decrement DP5Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

Branch if Bit Set

The execution of the branch in this instruction is contingent on the status of a selected bit in the Common register. Prior to execution of this instruction, an instruction must be executed to load the FP1 register with the address of the bit in the Common register to be tested for the contingency. If the selected Common register bit is one, this instruction will transfer control from the current program address to the address specified in the Argument Field.

Label	Command	Argument	Comment
symbol	BBS	$\underline{a}(r)\pm k$, cd	

• Label

Format

Any valid symbol or blank.

BBS

Command

Argument

a

• k

• r

The Control Memory Address is a symbolic or absolute address in Bulk Core, Page Memory, or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form $a(r)\pm k$, cd.

This entry is only required if the optional term (r) is omitted. This term may be either a symbol or a constant.

This optional term must be a constant and modifies 'a'.

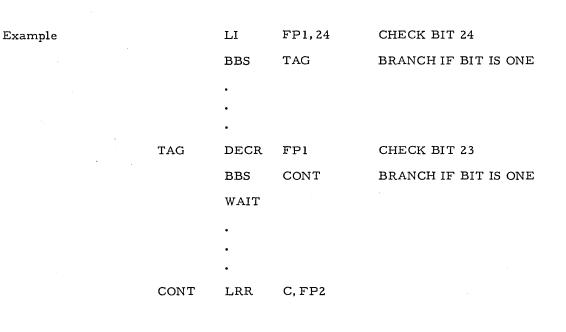
This entry may be one of the following nine registers: R0 through R7, DP. The contents of the specified register is added to the value $a\pm k$ execution time. The result defines the Control Memory Address. The contents of the register can be considered the base address, and the $a\pm k$ expression can be considered the displacement.

• • cd

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an a±k type of expression, where 'a' and k are defined as above.

cd Values	Action
1 2	Decrement BL Increment DP
3	Decrement BL and increment DP
4	Decrement DP
5	Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.



Common Register Contents

0	23	24	3
	0	1	

The first branch (BBS TAG) will take place, since FP1 is loaded with 24 and bit 24 in the Common register is one.

The second branch (BBS CONT) will not take place, since after the DECR, FP1 contains 23 and bit 23 in the Common register is zero. Thus the next instruction executed will be the WAIT.

Branch if Bit Zero

BBZ

The execution of the branch in this instruction is contingent on the status of a selected bit in the Common register. Prior to execution of this instruction, an instruction must be executed to load the FP1 register with the address of the bit in the Common register to be tested for the contingency. If the selected Common register bit is zero, this instruction will transfer control from the current program address to the address specified in the argument field.

Format	Label	Command	Argument	Comment
	symbol	BBZ	$\underline{a}(r) \pm k, cd$	
• Label	Any valid sy	mbol or blank.		
• Command	BBZ			
• Argument	Bulk Core,	Page Memory, or	is a symbolic or al High Speed Data Bu resented by four ter	
• • a	-	s required only if e either a symbol	the optional term () or a constant.	r) is omitted. This
• • k	This optiona	al term must be a	constant and modifi	es 'a'.
• • r	DP. The co at execution The content	ontents of the spec time. The results of the register c	ified register is add t defines the Contro	e base address, and
• • cd	the specifie increment o decrement t	d instruction is co or decrement the d the block length (B cified by an a±k ty	-	esired. This step may gister by one and/or The Control Digit
	cd Values	<u>Ac</u> Decremen	tiont BL	

Image: Construction of the second s

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

В	в	Ζ
---	---	---

 Common Register 	0	10 1		31
	CONT	DECR	FP2	
		•		
		•		
		•		
		WAIT		
		BBZ	CONT	BRANCH IF BIT IS ZERO
	TAG	INCR	FPl	CHECK BIT 11
		•		
		•		
		•		
		BBZ	TAG	BRANCH IF ZERO
Example		LI	FP1,10	CHECK BIT 10

C • Register Contents

0 1

The first branch (BBZ TAG) will take place, since FP1 is loaded with the number 10 and bit 10 in the Common register is zero. The second branch (BBZ CONT) will not take place, since after the INCR, FP1 contains 11 and bit 11 in the Common register is one. Thus the next instruction executed will be the WAIT.

Branch if Response

BRS

This instruction will check the Y response store register. If any Y response store register bit position is set to one in any enabled array, the branch will be executed.

Format	Label	Command	Argument	Comment
	symbol	BRS	<u>a(</u> r)±k, cd	
• Label	Any valid s	ymbol or blank		
• Command	BRS			
• Argument	Bulk Core,	Page Memory, o	•	absolute address in Buffer. The Control erms in the form
• • a	-	is required only if e either a symbol	the optional term or a constant.	(r) is omitted. This
• • k	This option	al term must be a	constant and mod	ifies 'a'.
• • r	DP. The c execution t contents of	contents of the spe ime. The result the register can l	cified register is a lefines the Control	isters: R0 through R7, added to the value $a\pm k$ at Memory Address. The base address, and the $a\pm k$
• • cd	specified in increment decrement	nstruction is comp or decrement the the block length (d by an a±k type o	leted a step is des data pointer (DP) n BL) register by on	it indicates that after the ired. This step may register by one and/or e. The Control Digit may re 'a' and k are as
	cd Values		Action	
	1 2 3 4 5	Decremen	t DP nt BL and increme:	nt DP
				· · · · · ·

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

2-27

BNR

Branch if No Response

This instruction will check the Y response store register. If all Y response store register bit positions are equal to zero in all enabled arrays, the branch will be executed.

Label	Command	Argument	Comment
symbol	BNR	<u>a(</u> r)±k, cd	

Label

Format

Any valid symbol or blank.

BNR

• Command

• Argument The Control Memory Address is a symbolic or absolute address in Bulk Core, Page Memory, or High Speed Data Buffer. The Control Memory Address may be represented by four terms in the form a(r)±k, cd.

- • a This entry is required only if the optional term (r) is omitted. This term may be either a symbol or a constant.
- k This optional term must be a constant and modifies 'a'.

This entry may be one of the following nine registers: R0 through R7, DP. The contents of this specified register is added to the value a±k at execution time. The result defines the Control Memory Address. The contents of the register can be considered the base address, and the a±k expression can be considered the displacement.

• • cd

• r

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an $a\pm k$ type of expression, where 'a' and k are defined as above.

cd Values	Action
1	Decrement BL
2	Increment DP
3	Decrement BL and increment DP
4	Decrement DP
5	Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

Branch if Overflow

BOV

This instruction allows the programmer to test for an overflow or underflow condition after an arithmetic operation. This instruction will perform X exclusive OR Y ANDed with M and store the result in Y. Then if any Y response store bit in any enabled array equals one an overflow condition exists for that word. If such is the case, this instruction will branch to the address specified in the Argument Field. The X response store will equal one for the corresponding word of associative memory if an underflow occurred; otherwise an overflow occurred.

Format	Label	Command	Argument	Comment
	symbol	BOV	<u>a(</u> r)±k, cd	
• Label	Any valid sy	mbol or blank.		
• Command	BOV			
• Argument	Bulk Core,	-	High Speed Data B	bsolute address in uffer. The Control Memory e form a(r)±k, cd.
• • a	-	s required only if e either a symbol o	-	r) is omitted. This
• • k	This optiona	al term must be a o	constant and modif	ies 'a'.
• • r	DP. The co a±k at execu The content	ontents of this spec ation time. The re	ified register is a sult defines the Co an be considered th	ontrol Memory Address. ne base address, and
• • cd	the specifie may increm and/or decr	d instruction is content or decrement to ement the block lent e specified by an a	mpleted a step is d the data pointer (Dingth (BL) register	indicates that after esired. This step P) register by one by one. The Control ion, where 'a' and k are
	<u>cd Values</u> 1 2 3 4 5	Decrement Increment Decrement Decrement	DP BL and increment	DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

2-29

Branch if No Overflow

BNOV

This instruction allows the programmer to test for the absence of an overflow or underflow condition following an arithmetic instruction. This instruction will perform X exclusive OR Y ANDed with M and store the result in Y. If all Y response store bits of all enabled arrays equal zero, i.e., no overflow condition exists, this instruction will branch to the address specified in the Argument Field. If the branch does not take place, the X response store will equal one for the corresponding word of associative memory if an underflow occurred; otherwise, an overflow occurred.

Format	Label	Command	Argument	Comment
	symbol	BNOV	<u>a</u> (r)±k, cd	
• Label	Any valid s	ymbol or blank.		
• Command	BNOV			
• Argument	Core, Page	e Memory, or Hi	gh-Speed Data Buff	r absolute address in Bulk fer. The Control Memory the form a(r)±k, cd.
• • a	-	is required only be either a symbo		n (r) is omitted. This
•• k	This optional term must be a constant and modifies 'a'.			
•• r	The conten time. The register ca	ts of this specific result defines th	ed register is adde le Control Memory the base address,	gisters: R0 through R7, DP. d to the value $a\pm k$ at execution Address. The contents of the and the $a\pm k$ expression can be
• • cd	specified in or decreme block lengt	nstruction is com ent the data point h (BL) register b	apleted a step is de er (DP) register b by one. The Contr	git indicates that after the sired. This step may increment y one and/or decrement the ol Digit may be specified by e defined as above.
	cd Values 1 2 3 4 5	Incre Decr Decr	Action ement BL ment DP ement BL and incr ement DP ement BL and DP	ement DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register. BAL

Branch and Link

This instruction will transfer control to a subroutine after storing the Execution Location Counter of the next instruction in the branch and link register r_1 .

Format	Label	Command	Argument	Comment
	symbol	BAL, r ₁	<u>a(r</u> 2)±k, cd	
• Label	Any valid s	symbol or blank.		· ·
• Command	BAL			
•• r ₁	One of the branch and link registers R0 through R7.			
• Argument	Core, Pag	e Memory, or Hig	gh-Speed Data Buff	absolute address in Bulk er. The Control Memory the form a(r)±k, cd.
• • a	This entry is required only if the optional term (r_2) is omitted. This term may be either a symbol or a constant.			
•• k	This option	nal term must be	a constant and mod	lifies 'a'.
• • r	The conter time. The register c	nts of this specifie result defines th	ed register is added e Control Memory the base address, a	gisters: R0 through R7, DP. I to the value $a\pm k$ at execution Address. The contents of the and the $a\pm k$ expression can be
•• cd	specified i increment decrement	instruction is com or decrement the the block length	pleted a step is de data pointer (DP) (BL) register by or	git indicates that after the sired. This step may register by one and/or ne. The Control Digit may are 'a" and k are defined as

cd Values	Action
1	Decrement BL
2	Increment DP
3	Decrement BL and increment DP
4	Decrement DP
5	Decrement BL and DP

•

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register. •

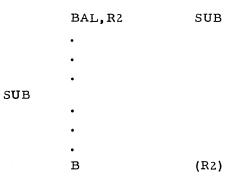
BAL

Note

.....

Example

When a programmer branches and links to a subroutine, he generally will return by issuing an unconditional branch on the register r_1 that specified the branch and link instruction:



RPT

Repeat

This instruction will execute the following instruction the number of times specified in the repeat constant term a±k. If a±k is omitted, it is assumed that FL1 previously has been loaded with the number of times minus one, the next instruction is to be repeated.

Format	Label	Command	Argument	Comment
· .	symbol	<u>RPT</u> , a±k		
• Label	Any valid sy	ymbol or blank.		
• Command	RPT			
• • a±k	'a' may be either a constant or a symbol, and k is an optional constant modifier. The value of this optional term specifies the number of times the following instruction will be repeated, i.e., $l \le a \pm k \le 256$.			
• • blank	that the nex			of times, minus one, should be loaded with
• Argument	No entry re	quired.		
Note	FL1 will be	decremented to z	ero when this instr	uction is executed.

LOOP

Loop

This instruction will sequentially cycle the program from the program counter location following the loop instruction up to and including the address specified in the Argument Field. The loop may be cycled any number of iterations from 1 to 256. After the loop has cycled the specified number of times, the program continues with the next address following the Argument Field address.

Format	Label	Command	Argument	Comment
	symbol	LOOP, altkl	a ₂ (r)±k ₂	

• Label Any valid symbol or blank.

LOOP

the program will be cycled.

Command

• • a₁±k₁

•• blank

APPLE assumes the number of loop iterations, minus one, has already been loaded into FL1 by the programmer.

 \mathbf{a}_1 may be either a constant or a symbol, and \mathbf{k}_1 is an optional constant

modifier. The value of this optional term specifies the number of times

Argument The Control Memory Address is a symbolic or absolute address in Bulk
 Core, Page Memory, or High Speed Data Buffer. The Control Memory
 Address may be represented by three terms in the form a₂(r)±k₂.

• • $a_2^{\pm k_2}$

 a_2 may be either a constant or a symbol, and k_2 is an optional constant modifier. The value of the required term specifies the Control Memory Address of the last instruction of the sequence of instructions cycled by the LOOP.

• r This entry may be one of the following nine registers: R0 through R7, DP. The contents of this specified register is added to the value of $a_2\pm k_2$ at execution time. The result defines the Control Memory Address. The contents of the register can be considered the base address, and the $a_2\pm k_2$ expression can be considered the displacement.

Note 1Instructions that alter the program counter, i.e., branches, skips, externalfunctions, etc., will produce unpredictable results if used within a loop.Also, Load and Store register instructions are illegal within a loop.

Note 2 Execution times can be improved for instructions within a loop.

Note 3 FL1 register will be decremented to zero upon completion of the loop.

Note 4 The register modification term, r, is only legal when the number of iterations term, $a_1\pm k_1$, is omitted.

REGISTER INSTRUCTIONS

The register instructions allow the programmer to either alter or save the contents of STARAN S registers.

The register instructions are:

Instruction
Load Register from Register
Load Register with Immediate Data
Load Register from Control Memory
Store Register in Control Memory
Increment the Register
Decrement the Register
Load Program Status Word
Swap Program Status Word

Mnemonic	Register Name	Length in Bits
AS	Array Selector	32
ASH	Most-Significant Bits of Array Selector	16
ASL	Least-Significant Bits of Array Selector	16
BL	Block Length Counter	16
DP	Data Pointer	16
С	Common Register	32
CH	Most-Significant Bits of Common Register	16
CL	Least-Significant Bits of Common Register	16
F	Field Register group (FL1, FP3, FP1, FP2)	32
FLl	Field Length Counter 1, Bits 0 to 7 of F	8
FP3	Field Pointer 3, Bits 8 to 15 of F	8
FP1	Field Pointer 1, Bits 16 to 23 of F	8
FP2	Field Pointer 2, Bits 24 to 31 of F	8
FL2	Field Length Counter 2	8
FPE	Field Pointer Extra	8
PC	Program Counter, Most-Significant Bits of PSW	16
IMASK	Interrupt Mask, Least-Significant Bits of PSW	4
RO	Branch and Link Register 0	32
Rl	Branch and Link Register 1	32
R2	Branch and Link Register 2	32
R 3	Branch and Link Register 3	32
R4	Branch and Link Register 4	32
R5	Branch and Link Register 5	32
R6	Branch and Link Register 6	32
R7	Branch and Link Register 7	32

Table 2-1. Registers

Table 2-2. Register Combinations

Valid Register Combinations	Length in Bits
(ASH, ASL) or AS	32
(BL,DP)	32
(CH,CL) or C	32
(FL1, FP3, FP1, FP2) or F	32
(FL1, FP3)	16
(FP3, FP1)	16
(FP1, FP2)	16
(FP2, FL1)	16
(FL2, FPE)	16
(PC,IMASK)	32

LRR

Load Register From Register

This instruction will load register or valid register combination r_2 with the contents of register or valid register combination r_1 . The contents the source register is not affected, and the original contents of the destination register is destroyed.

Format	Label	Command	Argument	Comment	
	symbol	LRR, k _s	<u>r₂, r</u> 1		
• Label	Any valid s	ymbol or blank.		ı	
• Command	LRR				
•• ^k s	k_s may be either a constant or a symbol.				
	Legal value	20:			
	1	- Shift the cont around 8 bits	tents of the source before loading.	e register (r _l) left end	
	2		tents of the source ts before loading.	e register (r ₁) left end	
	3		cents of the source s before loading.	e register (r ₁) left end	
	symbol	- Must be equa	al to a value of 1,	2, or 3.	
	blank	register r ₁ t		nstant to the source significant bits of the	
• Argument	Both regist	ter entries are re	quired.		
•• r ₂	The destination	ation register(s)			
	Valid Entr	ies:			
	Any reg	ister or register	combination noted	l in table 2-2.	
•• r ₁	The source	e register(s)			
	Valid Entries:				
	Any reg	ister or register	combination noted	l in table 2-2.	
Note l		ster) error indica which is invalid.	tes both r_1 and r_2	are branch and link	
	R	L	RR R0, R1		
Note 2		ncation) error was all bits cannot be l	-	rger register than r ₂ ;	

LRR

Example

т

w

LRR FP1, (PC, IMASK)

32-bit (PC, IMASK) cannot be loaded into 8-bit FP1.

Note 3

A W (Warning) error warns that r_1 is a smaller register than r_2 . Not only is r_1 loaded into r_2 , but also the other register or registers in r_1 's group are loaded into r_2 . (See Reference Manual, Bus Positions.)

Example

LRR AS, FP2

32-bit register AS is loaded with the four 8-bit registers, FL1, FP3, FP1, FP2.

Load Register with Immediate Data

This instruction will load register or valid register combination r with the value of $a \pm k$ in the Argument Field.

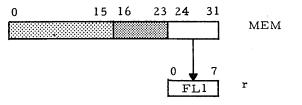
		l		l	
Format	Label	Command	Argument	Comment	
	symbol	LI, k _s	<u>r,a</u> ±k		
• Label	Any valid s	ymbol or blank.		•	
• Command	LI				
•• k _s	k _s may be a	either a constant	or a symbol.		
	Legal value	s:			
	1	- Shift the valu	ae of a±k left end-a	around 8 bits before loading.	
	2	- Shift the valu	ue of a±k left end-a	around 16 bits before loading.	
	3 - Shift the value of a±k left end-around 24 bits before loadi				
	symbol - Must be equal to a value of 1, 2, or 3.				
	blank	least-signifi		nstant to the data to align the with the least-significant	
• Argument	Both entrie	s are required.			
• • r	The destina	tion register(s).			
Valid entries:					
	Any reg	gister or register	combination noted	d in table 2-2.	
● ● a±k	expression	whose value is le		ngle-term or a double-term 'a' may be either a constant r.	

 \mathbf{LI}

LI

Example 1*

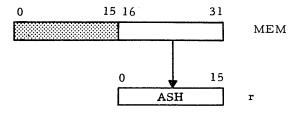
LI FL1, MEM



* Typical for 8-bit registers

Example 2**

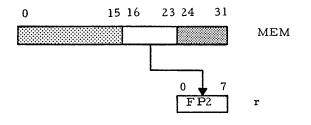




** Typical for 16-bit registers

Example 3

LI,3 FP2, MEM



Load Register From Control Memory

This instruction will load the register or valid register combination r_2 with the contents of the Control Memory Address specified by $a(r_1)\pm k$. The Control Memory Address is a symbolic or absolute address in Bulk Core or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form $a(r)\pm k$, cd. The contents of the control memory address is not affected. However, the contents of the base register may be changed by the control digit (cd). The original contents of the destination register is destroyed.

Format	Label	Command	Argument	Comment	
	symbol	<u>LR</u> , k _s	<u>r₂,a</u> (r ₁)±k,cd		
• Label	Any valid s	ymbol or blank.			
• Command	LR				
•• ^k s	k _s may be e	either a constant	or a symbol.		
	Legal value	es:			
	1	- Shift the cont 8 bits before	tents of the address loading the registe	$a(r_1) \pm k$, left end-around r.	
	2	- Shift the cont 16 bits befor	tents of the address e loading the regist	$a(r_1) \pm k$, left end-around ser.	
	3		contents of the address $a(r_1) \pm k$, left end-around before loading the register.		
	symbol	- Must be equa	al to a value of 1, 2,	, or 3.	
	blank	- APPLE assu	mes that no shifting	g is desired.	
• Argument	Two entries are required.				
• • ^r 2	The destina	tion register(s).			
	Valid entrie	es:			
	Any regi	ister or register	combination noted i	n table 2-2.	
• • a±k	'a' may be	either a constant	or a symbol; k is a	n optional constant modifier.	
	The value o	of this term speci	fies a Control Mem	ory Address.	
•• r ₁ This entry may be one of the follow DP.			e following nine reg	isters: R0 through R7,	
	execution t contents of	ime. The result the register can	d register is added defines the Control be considered the b dered the displacen	Memory Address. The base address, and the	

2-41

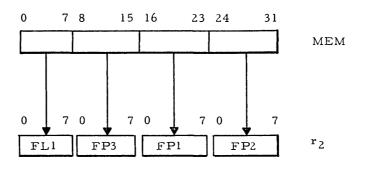
LR

•• cd

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an a±k type of term, where, 'a' and k are as defined as above.

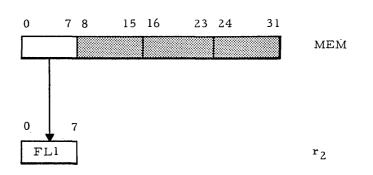
cd Values	Action
1 2 3 4 5	Decrement BL Increment DP Decrement Bl and increment DP Decrement DP Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.



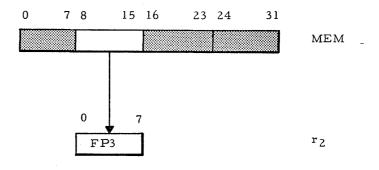
Example 2

LR FL1, MEM



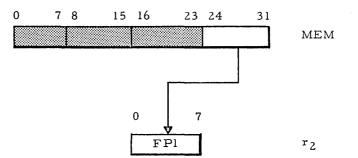
Example 3

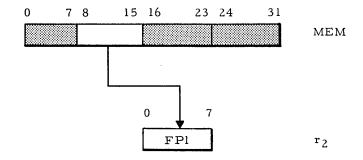
LR FP3, MEM



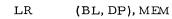
Example 4

LR,1 FP1, MEM

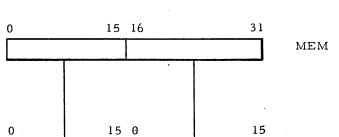




Example 6



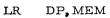
BL

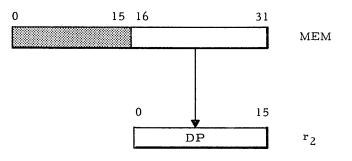


 DP

 \mathbf{r}_2

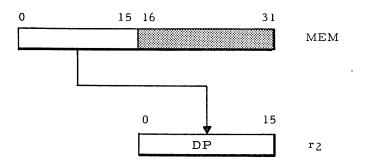
Example 7





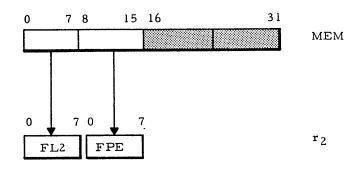
Example 8

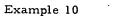
LR,2 DP,MEM

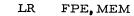


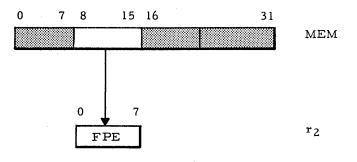
LR (FL2, FPE), MEM





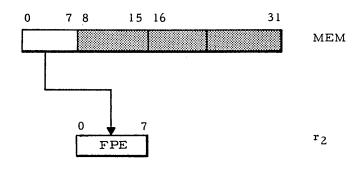






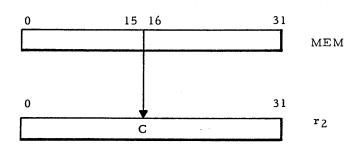
Example 11

LR,3 FPE,MEM



Example 12

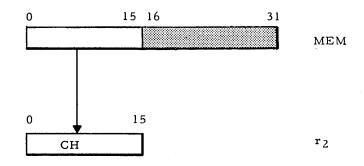




LR

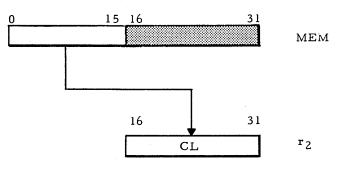
Example 13





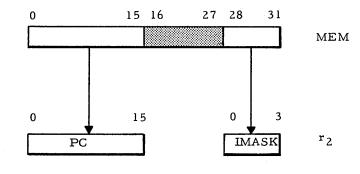
Example 14

LR,2 CL,MEM

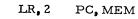


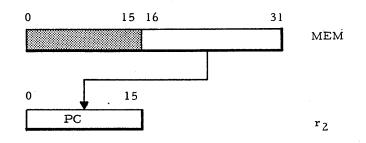
Example 15

LR (PC, IMASK), MEM



Example 16





2 16

Store Register in Control Memory

This instruction will store the contents of the register or valid register combination r_2 in the Control Memory Address specified in $a(r_1)\pm k$, cd. The Control Memory Address is a symbolic or absolute address in Bulk Core or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form $a(r)\pm k$, cd. The contents of the source register is not affected and the contents of the Control Memory Address destination is destroyed. The contents of the base register (r_1) may be changed by the control digit, cd.

Format	Label	Command	Argument	Comment	
	symbol	<u>SR</u> ,k _s	<u>r₂, a</u> (r ₁)±k, cd		
• Label	Any valid s	ymbol or blank.			
• Command	SR				
•• k s	k _s may be a	either a constant or a symbol.			
	Legal value	Legal values:			
	1	- Shift the con before storin	tents of register (r ₂ ng in Control Memor) left end-around 8 bits ry.	
	2	- Shift the con before storin	tents of register (r ₂ ng in Control Memor) left end-around 16 bits ry.	
	3	- Shift the contents of register (r ₂) left end-around 24 bits before storing in Control Memory.			
	symbol	- Must be equa	al to a value of 1, 2,	, or 3.	
	blank	- APPLE assu	mes that no shifting	g is desired.	
• Argument	Two entries	s are required.			
• • r ₂	The source	register(s).			
	Valid entrie	es:			
	Any reg	ister or register	combination noted i	n table 2-2.	
•• a±k	'a' may be	either a constant	or a symbol; k is a	n optional constant modifier.	
	The value of the term specifies the Control Memory Address.				
•• ^r 1	This entry	may be one of th	e following nine reg	isters: R0 through R7, DP.	
	time. The register ca	result defines th	ne Control Memory the base address, a	to the value $a\pm k$ at execution Address. The contents of the and the $a\pm k$ expression can be	
● ● cd	specified i increment the block l	nstruction is com or decrement the ength (BL) regist	npleted a step is des e pointer (DP) regis	it indicates that after the fired. This step may ter by one and/or decrement ntrol Digit may be specified fined as above.	

S	Ľ	F	2
		•	`

S.R

•	•	\mathbf{cd}	Valid	entries:

cd Values	Action (After the Storage Operations)
1	Decrement BL register
2	Increment DP register
3	Increment DP register and decrement BL register
4	Decrement DP
5	Decrement both DP and BL registers

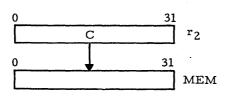
The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

Note

When register r_2 is not a 32-bit register or group, a W (Warning) error is produced to remind the programmer that the whole 32-bit group will be stored.



SR C, MEM

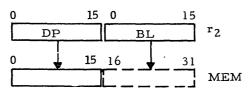


Example 2

SR DP, MEM

W

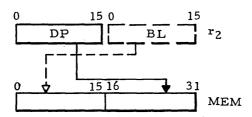
W



When register r_2 is not a 32-bit register or group, a W (Warning) error is produced to remind the program mer that the whole 32-bit group will be stored. Location MEM will contain both DP and BL.

Example 3

SR,2 DP,MEM



INCR

Increment the Register

The contents of the registers specified in the argument field by the term r_1, \ldots, r_n will be incremented by one.

Format	Label	Command	Argument	Comment		
	symbol	INCR	$\underline{r}_1,\ldots,\underline{r}_n$			
• Label	May be any	valid symbol or	blank.			
• Command	INCR					
• Argument	The term r_1, \ldots, r_n identifies the specific registers to be incremented by one. <u>Valid entries:</u> FP1 - Field Pointer 1 FP2 - Field Pointer 2 FP3 - Field Pointer 3 FPE - Field Pointer Extra FL1 - Field Length Counter 1, Decrementable only FL2 - Field Length Counter 2, Decrementable only BL - Block Length Counter, Decrementable only DP - Data Pointer					
Note 1	If the DP is listed.	specified in the	Argument Field, it	must the only register		
Note 2	may be use	d in a single inst		gisters, FLl, FL2, and BL, chosen it will be decremented, incremented.		
Note 3	FP2 and FP3 may not be used in the same instruction.					
Note 4	A register may not appear more than once in any given INCR instruction.					
Note 5		specified. Such		"decrementable only" decremented rather than		

2-49

DECR

.

Decrement the Register

The contents of the registers specified in the argument field by the term r_1, \ldots, r_n will be decremented by one.

Format	Label	Command	Argument	Comment			
	symbol	DECR	$\underline{r}_1, \ldots, \underline{r}_n$				
• Label	Any valid symbol or blank.						
• Command	DECR						
• Argument	The term r_1, \ldots, r_n identifies the specific registers to be decremented by one.						
	Valid entrie	es:					
	FP1 FP2 FP3 FL1 FL2 BL DP	- Field Length	rr 2 rr 3 er Extra n Counter 1, Decre n Counter 2, Decre h Counter, Decrem	ementable only			
Note 1	If the DP is listed.	specified in the	Argument Field, i	it must be the only register			
Note 2	Only one of the three "decrementable only" registers, FL1, FL2, and BL, may be chosen in a single instruction.						
Note 3	FP2 and FP3 may not be used in the same instruction.						
Note 4	A register	may not appear 1	more than once in a	any given DECR instruction.			

LPSW

Load Program Status Word

This instruction will load the contents of a designated Control Memory Address into the Program Counter (PC) and Interrupt Mask (IMASK) registers. From the Control Memory word, bits 0 through 15 are loaded into the Program Counter and bits 28 through 31 into the Interrupt Mask. The contents of the source memory word is not affected. The original contents of the registers are destroyed.

Format	Label	Command	Argument	Comment			
	symbol	<u>LPSW</u> ,k _s	$\underline{a}(\mathbf{r})\pm\mathbf{k}, \mathbf{cd}$				
• Label	Any valid symbol or blank.						
• Command	LPSW						
• • k _s	k _s may be	either a constant	or a symbol.				
	Legal value	es:					
	1	- Shift the contents of the address $a(r_1)\pm k$, left end-around 8 bits before loading the register.					
	2	 Shift the contents of the address a(r₁)±k, left end-around 16 bits before loading the register. Shift the contents of the address a(r₁)±k, left end-around 24 bits before loading the register. 					
	3						
	symbol	1 - Must be equal to a value of 1, 2, or 3.					
	blank	- APPLE assu	imes that no shiftin	ng is desired.			
• Argument	The Contro	ol Memory Addre	ss is a symbolic or	absolute address in Bulk			
	Core or Hi	igh-Speed Data Bu	iffer. The Control	Memory Address may be			
	represente	ed by four terms	in the form a(r)±k,	cd.			
• • a±k	'a' may be	either a constant	or a symbol; k is	an optional constant modifier.			
	The value	of this term spec	ifies a Control Mer	mory address.			

• • r

The contents of the specified register is added to the value $a\pm k$ at execution time. The result defines the Control Memory Address. The contents of the register can be considered the base address; and the $a\pm k$ expression can be considered the displacement.

This entry may be one of the following nine registers: R0 through R7, DP.

• • cd

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the data pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by $a\pm k$ type of term, where 'a' and k are defined as above.

De eu cut BI
Decrement BL Increment DP Decrement BL and increment DP Decrement DP Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register.

SPSW

Swap Program Status Word

This instruction will store the current Program Counter (PC) and Interrupt Mask (IMASK) registers in the designated Control Memory Address. If the Control Memory Address is an even address (see EVEN), the contents of the next location will be loaded into the PC and IMASK registers. Loading the PC register causes a branch to the address loaded. If the Control Memory address is an odd address, only the store takes place.

Format	Label	Command	Argument	Comment
	symbol	SPSW	<u>a</u> (r)±k, cd	

• Label Any valid symbol or blank.

Command SPSW

- Argument The Control Memory Address is a symbolic or absolute address in Bulk
 Core or High-Speed Data Buffer. The Control Memory Address may be represented by four terms in the form a(r)±k, cd.
- a±k 'a' may be either a constant or a symbol; k is an optional constant modifier. The value of this term specifies a Control Memory Address.
 - This entry may be one of the following nine registers: R0 through R7, DP. The contents of the specified register is added to the value a±k at execution time. The result defines the Control Memory Address. The contents of the register can be considered the base address, and the a±k expression can be considered the displacement.
- •• cd

r

This entry is the Control Digit. A Control Digit indicates that after the specified instruction is completed a step is desired. This step may increment or decrement the pointer (DP) register by one and/or decrement the block length (BL) register by one. The Control Digit may be specified by an $a\pm k$ type of term, where 'a' and k are defined as above.

cd Values	Action
1 2 3 4 5	Decrement BL Increment DP Decrement BL and increment DP Decrement DP Decrement BL and DP

The Control Digit is a valid entry only when the base register option has been selected, and the register forming the base register is the DP register. SPSW

Example

PUT

GET

EVEN DS DC X'90000009' .

SPSW PUT

The current PC and IMASK registers will be stored at location PUT (note that the address of PUT will always be even); then X'9000' will be loaded into PC register and X'9'loaded into IMASK register. Since the program counter (PC) is being loaded, a branch to location X'9000' occurs.

ASSOCIATIVE INSTRUCTIONS

The associative instructions allow the programmer to load, store, search, move, and perform arithmetic operations on the associative array memory and the response store registers X, Y, and M.

LOADS

This group of associative instructions allows the programmer to load the response store registers or the Common register from an associative memory bit column or an associative memory field, respectively. All instructions dealing with response store registers and/or associative memory fields or bit columns, only affect those associative array memory modules enabled via the Array Select register. The response store registers and associative array memory modules disabled via the Array Select register remain unchanged.

Mnemonic	Instructions
L	Load Response Store Register
LN	Load Complemented
LOR	Load Logical OR
LORN	Load Logical OR Complemented
LAND	Load Logical AND
LANDN	Load Logical AND Complemented
LXOR	Load Logical Exclusive OR
LXORN	Load Logical Exclusive OR Complemented
LC	Load Common Register from an Associative Memory Word
LCM	Load a Common Register Field from an Associative Memory Word
SET	Set Response Store Register
CLR	Clear Response Store Register
ROT	Rotate Response Store Register

L

Load Response Store Register

This instruction will load the response store register, rs_2 , with the designated source. The content of the source is not affected, and the original content of the destination, rs_2 , is destroyed.

Format	Label	Command	Argument	Comment
		L	$\frac{rs}{2^{2}} \left[\frac{rs}{a^{\pm k}} \right]$	

Label

Any valid symbol or blank.

L

Command

• Argument Two entries are required. The first entry is the destination; the second entry is the source. As shown there are three distinct types of source expressions. The brackets are not a part of the possible argument field terms.

• • rs₂

The destination response store register.

Valid entries:

X - X response store register
Y - Y response store register
M - M response store register

The source response store register.

Valid entries:

X - X response store register Y - Y response store register M - M response store register

• • a±k

• • r

 $\bullet \bullet rs_1$

'a'may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a'is present as a symbol. If 'a'was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a source bit position in all words of enabled associative memory. The value of a±k should be $0 \le a\pm k \le 255$.

A field pointer register, which may be post-incremented or postdecremented. If this form is used, the response store register, rs₂, is loaded indirectly through this register. This register contains the address of the source bit column.

Valid entries:

FPl Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement

1 ì

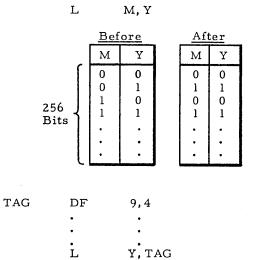
I

t

Example 1

Example 2

L

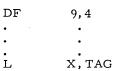


Y, TAG

		Before		After		
_	Y	Array Memory Column 9	Y	Array Memory Column 9		
256 Bits	0 0 1 1	0 1 0 1 •	0 1 0 1	0 1 0 1		

Example 3

TAG



•

• . L

	Before				After			
	x	Array Memory Column 9		x	Array Memory Column 9			
256 Bits	0 0 1 1	0 1 0 1 •		0 1 0 1	0 1 0 1			

2-57

Load Complemented

This instruction will load the response store register, rs_2 , with the one's complement value of the designated source. The content of the source is not affected and the original content of the destination, rs_2 , is destroyed.

Format	Label	Command	Argument	Comment		
	symbol	LN	$\frac{rs}{2^{2}} \begin{bmatrix} \frac{rs}{a\pm k} \\ \frac{r}{r} \end{bmatrix}$			
• Label	Any valid symbol or blank.					
• Command	LN					
• Argument	Two entries are required. The first entry is the destination; the second entry is the source. As shown, there are three distinct types of source expressions. The brackets are not a part of the possible argument field terms.					
• • rs ₂	The destina Valid entri	ation response s	store register.			
	Y - Y	response store response store response store	register			
• • rs ₁	The source	response store	e register.			
	Valid entries:					

X - X response store register Y - Y response store register M - M response store register

••a±k

• r

LN

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a source bit position in all words of enabled associative memory. The value of a±k should be $0 \le a \pm k \le 255$.

A field pointer register. If this form is used, the response store register, rs_2 , is loaded indirectly through this register. This register contains the address of the source bit column.

Valid entries:

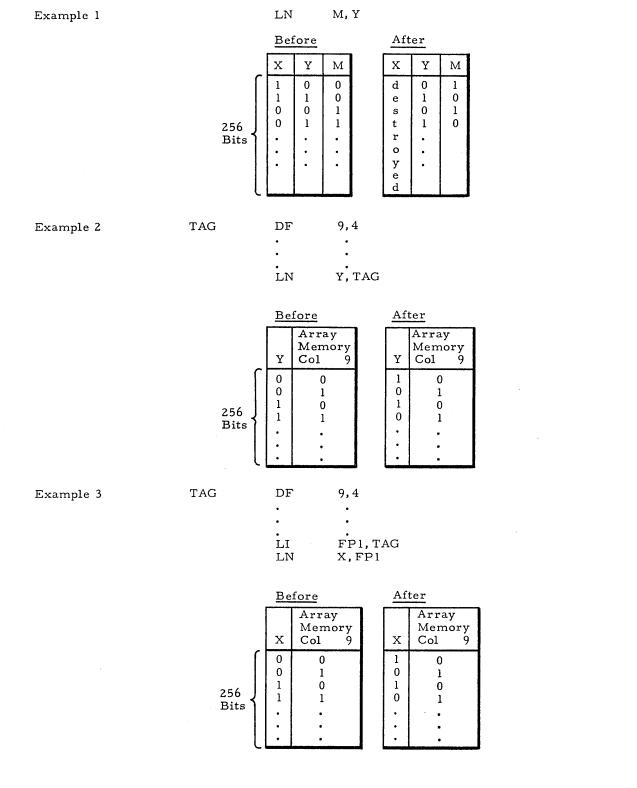
FP1 Field Pointer 1 Field Pointer 1 with a post-increment FP1+ Field Pointer 1 with a post-decrement FP1-FP2 Field Pointer 2 Field Pointer 2 with a post-increment FP2+ FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement

2-58

LN

Note

If M is chosen for the rs_2 entry, the original content of the X response store register is destroyed when this multiple instruction is executed.



Load Logical OR

LOR

This instruction will load the response store register, rs_2 , with a logical inclusive OR of itself and the value of the designated source. The content of the source is not affected and the original content of the destination, rs_2 , is destroyed.

		ł	1	1
Format	Label	Command	Argument	Comment
	symbol	LOR	<u>rs</u> 2, <u>rs</u> 1 <u>a±k</u> r	
		1		1
• Label	Any valid	symbol or blank.		
• Command	LOR			
• Argument	entry is th	e source. As sh	own, there are thre	the destination; the second ee distinct types of source the possible argument field
• • ^{rs} 2	The destin	ation response s	tore register.	
	Valid entr	ies:		
	Y - Y	response store i response store i response store i	register	
• • ^{rs} l	The sourc	e response store	register.	
	Valid entr	ies:		
	Y - Y	response store i response store i response store i	egister	
● ● a±k	'a' may be	a constant or a s	ymbol; k is an op	tional constant modifier.
	k is legal	only when a' is p	resent as a symbol	l. If'a' was defined as a
	field via a	DF instruction,	the most_significat	nt bit position is the value
	used. Thi	s term represen	ts a source bit pos	ition in all words of
	enabled as	sociative memor	y. The value of $a \pm b$	should be $0 \le a \pm k \le 255$.
• • r	A field poi	nter register. I	f this form is used	, the response store
	0	8		is register. This register
	contains th	ne address of the	source bit column	•
	Valid entr	ies:		
	FP1 FP1+ FP2- FP2+ FP2- FP3 FP3+	Field Pointer 1 Field Pointer 1 Field Pointer 2 Field Pointer 2 Field Pointer 2 Field Pointer 3 Field Pointer 3	with a post-incro with a post-decr with a post-incro with a post-incro with a post-decr	ement ement

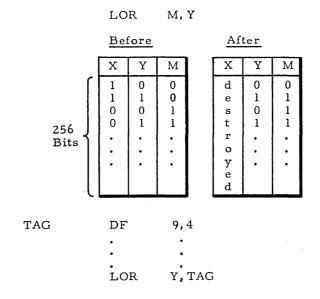
FP3+ Field Pointer 3 with a post-increment FP3- Field Pointer 3 with a post-decrement LOR

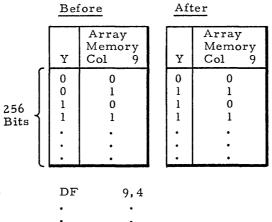
Note

If M is chosen for the rs_2 entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

Example 1

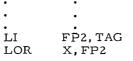
Example 2

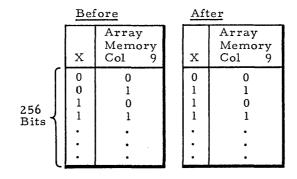




Example 3







2-61

LORN

Load Logical OR Complemented

This instruction will load the response store register, rs_2 , with the logical inclusive OR of itself and the one's complement of the value of the designated source. The content of the source is not affected and the original content of the destination, rs_2 , is destroyed.

Label Command Argument Comment Format LORN symbol rs₂ <u>a</u>±k Any valid symbol or blank. Label LORN Command Two entries are required. The first entry is the destination; the Argument second entry is the source. As shown, there are three distinct types of source expressions. The brackets are not a part of the possible argument field terms. The destination response store register. • rs₂ Valid entries: X - X response store register Y - Y response store register M - M response store register The source response store register. • rs1 Valid entries: ${\rm X}$ - ${\rm X}$ response store register Y - Y response store register M - M response store register 'a' may be a constant or a symbol; k is an optional constant modifier. • a±k k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most significant bit position is the value used. This term represents a source bit position in all words of The value of a±k should be $0 \le a \pm k \le 255$. enabled associative memory. A field pointer register. If this form is used, the response store • r register, rs₂, is loaded indirectly through this register. This register contains the address of the source bit column. Valid entries: FP1 Field Pointer 1 Field Pointer 1 with a post-increment FP1+ Field Pointer 1 with a post-decrement FP1-FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3

FP3+ Field Pointer 3 with a post-increment FP3- Field Pointer 3 with a post-decrement LORN

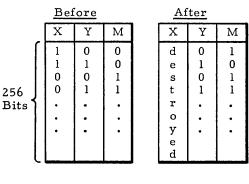
Note

٦

If M is chosen for the $\ensuremath{\mathsf{rs}_2}$ entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

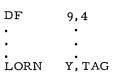
Example 1

LORN M,Y



Example 2

TAG



.

After Before Arra, Memory 7-1 9 Array Memory Col 9 Y Y 0 0 0 1 0 0 1 1 1 0 0 1 256 1 1 1 1 Bits • • . . • ٠ .

Example 3

LORN Υ,9

	Before			After		
	Y	Array Memory Col 9		Y	Array Memory Col 9	
256 Bits	0 0 1 1	0 1 0 1 •		1 0 1 1	0 1 0 1 •	

LAND

Load Logical AND

This instruction will load the response store register, rs_2 , with a logical AND of itself and the value of the designated source. The content of the source is not affected and the original content of the destination, rs_2 , is destroyed.

Format	Label	Command	Argument	Comment
	symbol	LAND	$\frac{rs_{2i}}{\left[\frac{a}{2}\pm k\right]}$	

- Label Any valid symbol or blank.
- Command LAND

• Argument Two entries are required. The first entry is the destination; the second entry is the source. As shown, there are three distinct types of source expressions. The brackets are not a part of the possible argument field terms.

• • rs₂

Valid entries:

X - X response store register Y - Y response store register M - M response store register

The destination response store register.

• rs1

The source response store register.

Valid entries:

X - X response store registerY - Y response store registerM - M response store register

• • a±k

• • r

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most significant bit position is the value used. This term represents a source bit position in all words of enabled associative memory. The value of a±k should be $0 \le a \pm k \le 255$.

A field pointer register. If this form is used, the response store register, rs_2 , is loaded indirectly through this register. This register contains the address of the source bit column.

Valid entries:

FPl Field Pointer 1 Field Pointer 1 with a post-increment FP1+ FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 Field Pointer 2 with a post-increment FP2+ FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3- Field Pointer 3 with a post-decrement

LAND

Note

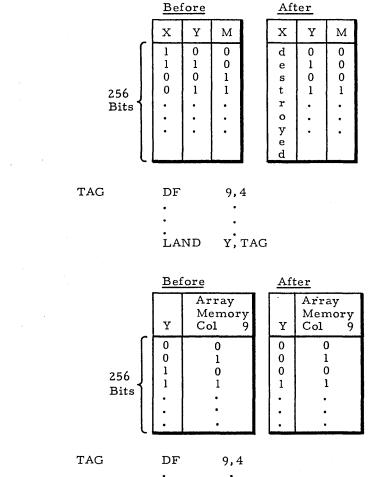
note

If M is chosen for the rs₂ entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

Example 1

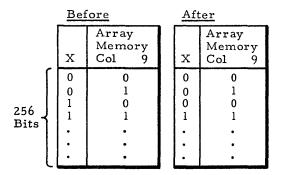
Example 2





Example 3

LI FP3, TAG LAND X, FP3



2-65

LANDN

Logical AND Complemented

This instruction will load the response store register, rs₂, with the logical AND of itself and the one's complement of the designated source. The content of the source is not affected and the original content of the destination, rs2, is destroyed in the execution of the load.

	T 1 1		1	Comment			
Format	Label	Command	Argument	Comment			
	symbol	LANDN	$\frac{rs}{2^{2}} \begin{bmatrix} \frac{rs}{1} \\ \frac{a \pm k}{r} \\ \frac{r}{r} \end{bmatrix}$				
• Label	Any valid	symbol or blank	. .				
• Command	LANDN	·					
• Argument		_	-	s the destination; the are three distinct			
	,.	ource expression rgument field te		are not a part of the			
• • ^{rs} 2	The destin	nation response	store register.				
	Valid entr	ies:					
	Y - Y	response store response store response store	register				
•• rs ₁	The sourc	e response stor	e register.				
	Valid entr	ies:					
	Y - Y	response store response store response store	register				
• • a±k	'a' may b	e a constant or	a symbol; k is an	optional constant modifier.			
	k is legal	only when 'a' is	s present as a sym	bol. If 'a' was defined as			
			-	icant bit position is the			
	_			bit position in all words of			
	enabled as	ssociative memo	ry. The value of a	a±k should be 0≤a±k≤255.			
• • r	A field po	inter register.	If this form is use	d, the response			
	-	2		ough this register.			
	This register contains the address of the source bit column.						
	Valid entr	ies:					
	FP1 FP1+ FP1- FP2 FP2+ FP2-	Field Pointer Field Pointer Field Pointer	<pre>1 with a post-incr 1 with a post-decr 2 2 with a post-incr 2 with a post-decr</pre>	ement ement			
	FP3 FP3+	Field Pointer					

Field Pointer 3 with a post-increment

FP3- Field Pointer 3 with a post-decrement

FP3 FP3+

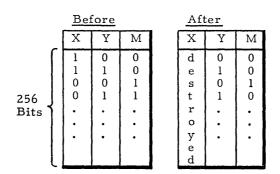
LANDN

Note

If M is chosen for the ${\tt rs}_2$ entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

Example 1

LANDN M,Y



Example 2

	Be	fore	Af	<u>ter</u>
_	Y	Array Memory Col 9	Y	Array Memory Col 9
256 Bits	0 0 1	0 1 0 1 •	0 0 1 0	0 1 0 1

LANDN Y,9

Example 3

TAG

D • •

2-67

LXOR

Load Logical Exclusive OR

This instruction will load the response store register, rs_2 , with the logical exclusive OR of itself and the value of the designated source. The content of the source is not affected and the original content of the destination, rs_2 , is destroyed.

Format	Label	Command	Argument	Comment
	symbol	LXOR	$\frac{rs}{2} \left[\frac{\frac{rs}{a \pm k}}{\frac{a \pm k}{r}} \right]$	

- Label Any valid symbol or blank.
- Command LXOR

• Argument Two entries are required. The first entry is the destination; the second entry is the source. As shown, there are three distinct types of source expressions. The brackets are not a part of the possible argument field terms.

The destination response store register.

Valid entries:

X - X response store register Y - Y response store register M - M response store register

• • rs₁

• • ^{rs}2

The source response store register.

Valid entries:

X - X response store registerY - Y response store registerM - M response store register

● ● a±k

• • r

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a'was defined as a field via a DF instruction, the most significant bit position is the value used. This term represents a source bit position in all words of enabled associative memory. The value of $a \pm k$ should be $0 \le a \pm k \le 255$.

A field pointer register. If this form is used, the response store register rs₂ is loaded indirectly through this register. This register contains the address of the source bit column.

Valid entries:

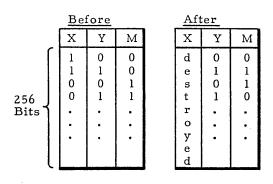
FPI Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement Field Pointer 3 FP3 FP3+ Field Pointer 3 with a post-increment FP3- Field Pointer 3 with a post-decrement LXOR

Note

If M is chosen for the rs₂ entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

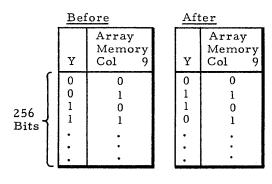
Example 1

LXOR M,Y



Example 2

LXOR Y,9



Example 3

LI FP2,X'9' LXOR X,FP2

	Before				er
	x	Array Memory Col 9		x	Array Memory Col 9
256 Bits	0 0 1	0 1 0 1		0 1 1 0	0 1 0 1

LXORN

Load Logical Exclusive OR Complemented

This instruction will load the response store register, rs_2 , with the logical exclusive OR of itself and the one's complement of the designated source. The content of the source is not affected and the original content of the destination, rs_2 , is destroyed in the execution of the load.

Format

Label	Command	Argument	Comment
symbol	<u>LX ORN</u>	$\frac{rs_{2}}{r} \begin{bmatrix} \frac{rs_{1}}{a^{\pm k}} \\ \frac{r}{r} \end{bmatrix}$	

Label

Any valid symbol or blank.

Command

Argument

• • rs₂

Two entries are required. The first entry is the destination; the second entry is the source. As shown, there are three distinct types of source expressions. The brackets are not a part of the possible argument field terms.

The destination response store register.

Valid entries:

LXORN

X - X response store register Y - Y response store register M - M response store register

The source response store register.

Valid entries:

X - X response store register Y - Y response store register M - M response store register

● ● a±k

• • rs₁

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a source bit position in all words of enabled associative memory. The value of a±k should be $0 \le a \pm k \le 255$.

A field pointer register. If this form is used, the response store register, rs_2 , is loaded indirectly through this register. This register contains the address of the source bit column.

Valid entries:

PF1Field Pointer 1FP1+Field Pointer 1 with a post-incrementFP1-Field Pointer 1 with a post-decrementFP2Field Pointer 2FP2+Field Pointer 2 with a post-incrementFP2-Field Pointer 3FP3+Field Pointer 3 with a post-incrementFP3-Field Pointer 3 with a post-decrement

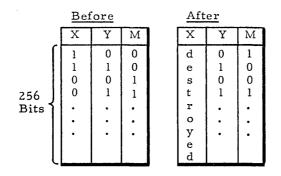
LXORN

Note

If M is chosen for the rs2 entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

Example 1

LXORN M,Y



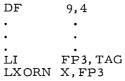
Example 2

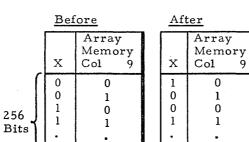
LXORN Y,9

	Before				er
	Y	Array Memory Col 9		Y	Array Memory Col 9
256 Bits	0 0 1 1	0 1 0 1 •		1 0 1 •	0 1 0 1 •

Example 3







.

.

•

•

.

Load Common Register from an Associative Memory Word

This instruction will load the Common register, right-justified with a field of the associative memory word whose address is in the link pointer (FP1, FP2).

Format	Label	Command	Argument	Comment		
	symbol	LC	<u>a</u>			
• Label	Any valid s	ymbol or blank.				
• Command	LC					
• Argument	One entry i	is required, an as	sociative memory	field expression.		
• • a	There are	two ways of denot	ing a field expressi	.on:		
	-	v be in the form D±i				
	modifie instruct and the associa	er. b should have ction. b represen e number of contig ative memory. T	bol, and i is an opt been previously d ts the most-signific guous bits occupied he optional constan significant bit posit	efined in a DF cant bit position by a field in t modifier, i,		
	-	be in the form b, i)±j				
	signifi a prev the val of cont	cant bit position o ious DF instructio ue used. i must iguous bits occup	f a field. If b was on, the most-signifi be a constant and r	represents the most- defined as a field via cant bit position is epresents the number is an optional constant tion of the field.		
Note 1	The link pointer (FP1 and FP2 registers) must be loaded with the address of the particular associative memory word prior to execu this instruction. Loading the link pointer is generally accomplish by use of the FIND, STEP, or RESVFST instruction.					
Note 2			ngth is less than 32 e Common registe:	bits, the most- r are cleared to zero.		
Note 3	If the array memory field length is greater than 32 bits, the most significant bits are truncated and the instruction is flagged with a T on the listing.					
Note 4	The X resp	oonse store regist	er is destroyed if s	hifting is required.		

 \mathbf{LC}

2-72

Example

FIND

LC (10,6)

•

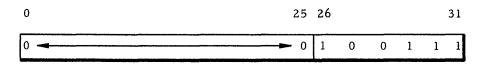
.

			Array Memory Bit Column				
	Y	10	11	12	13	14	15
Word 0	0	0	0	0	1	0	1
Word 1	0	1	1	1	0	1	0
•	•		•	•		•	•
•	! •	•	•	•	•	•	•
•	.		•		•	•	•
Word n	1	1	0	0	1	1	1
•	.	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•		•	•	•	•	•
Word 255	1	1	1	0	0	1	0

FP1 FP2

Address of Word n

Common Register



Load a Common Register Field From an Associative Memory Word

This instruction will load a field, a_1 , in the Common register with a field, a_2 , from the word of associative memory whose address is in the link pointer (FP1, FP2). All other bits in the Common register remain unchanged.

Label	Command	Argument	Comment
symbol	LCM	<u>a</u> 1, <u>a</u> 2	

Any valid symbol or blank.

LCM

Argument

• a₁, a₂

Command

Two entries are required. The first entry is the destination, a field in the Common register, the second entry is the source, a field in a word of associative memory.

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form (b, i)±j

> where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant modifying only the most significant bit position of the field.

The link pointer (FP1 and FP2 registers) must be loaded with the address of the particular associative memory word prior to executing this instruction. Loading the link pointer is generally accomplished by use of the FIND, STEP, or RESVFST instruction.

If the associative memory field length is less than the Common register field length, a W (warning flag) is noted on the listing. In this case the associative memory field will be loaded, right justified in the Common register field, and all remaining bits are unchanged.

If the associative memory field length is greater than the Common register field length, a T (truncation flag) is noted on the listing. In this case the most-significant bits of the associative memory field are truncated.

Note l

Note 2

Note 3

Note 4

The X response store register is destroyed if shifting is required.

2-74

Format

Label

LCM

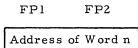
LCM

Example

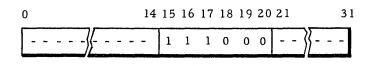
FIND LCM (15,6),(10,6) .

		Array Memory Bit Column					
	Y	10	11	12	13	14	15
Word 0	0	0	1	0	1	0	1
Word 1	0	1	0	1	0	1	0
•	•	•	•		•	•	
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
Word n	1	1	1	1	0	0	0
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•		•	•	•
Word 255	1	0	0	1	1	0	0

•



Common Register



Set Response Store Register

This instruction will set the designated response store, rs, to all ones.

Format

Label	Command	Argument	Comment
symbol	SET	rs	,

Label

• rs

Any valid symbol or blank.

Command

• Argument

One entry is required.

The designated response store to be set by the instruction.

Valid entries:

SET

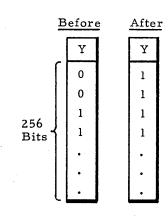
X - X response store register Y - Y response store register M - M response store register

Note

If M is chosen for the rs entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

Example

SET



Y

Clear Response Store Register

Any valid symbol or blank.

One entry is required.

This instruction will clear the designated response store, rs, to all zeroes.

Label	Command	Argument	Comment
symbol	CLR	rs	

Label

Format

CLR

• Command

CLR

• Argument

• • rs

The designated response store to be cleared by the instruction.

Valid entries:

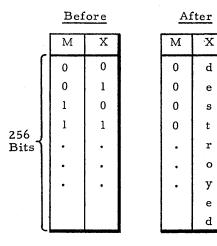
X - X response store Y - Y response store M - M response store

Note

If M is chosen for the rs entry, the original content of the X response store register is destroyed when this multiple instruction is executed.

Example

CLR M



ROT

Rotate Response Store Registers

This instruction will rotate the selected response store register right, end around.

Format	Label	Command	Argument	Comment
	symbol	ROT	<u>rs,a</u> 1±k1,a2±k2	

Any valid symbol or blank.

Label

Command

Argument

• rs

The response store register to be rotated.

Valid entries:

of the modulus $a_2 \pm k_2$.

ROT

X - X response store register Y - Y response store register M - M response store register

• a₁±k₁

• a₂±k₂

Note

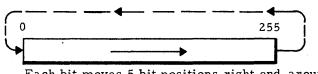
The modulus to be rotated. This optional term defines the length of the equal sections within the response store register. a_2 may be either a constant or a symbol: k2 is an optional constant modifier. The value of this term must be a power of 2, such that $1 \le a_2 \pm k_2 \le 128$. A default value of 256 is assumed.

The number of end-around bit positions to be rotated. a1 may be

either a constant or a symbol: k_1 is an optional constant modifier. A negative value indicates a left end-around rotate from leastsignificant bit position toward a more significant bit position. A positive value indicates a right end-around rotate from the mostsignificant bit position toward a less significant bit position. The absolute value of the rotate constant must be less than the value

If the M response store register is chosen, the X response store register is destroyed.

> ROT Y,5

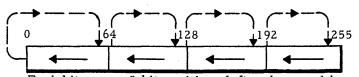


Each bit moves 5 bit positions right end-around.

Example 2

Example 1

ROT Y, _2,64



2-78

Each bit moves 2 bit positions left end-around in each section.

STORES

This group of associative instructions allows the programmer to store the response store registers or the Common register into an associative memory bit column or an associative memory field respectively. All instructions dealing with response store registers, and/or associative memory fields or bit columns only affect those associative array memory modules enabled via the Array Select register. The response store registers and associative array memory modules disabled via the Array Select register remain unchanged.

Mnemonic

Instructions

S	Store Response Store Into Associative Memory
SM	Store Response Store Masked Into Associative Memory
SN	Store Complement Into Associative Memory
SNM	Store Complement Masked Into Associative Memory
SOR	Store Logical Inclusive OR Into Associative Memory
SORM	Store Logical Inclusive OR, MASKED Into Associative Memory
SORN	Store Logical Inclusive OR, Complemented Into Associative Memory
SORNM	Store Logical Inclusive OR, Complemented, Masked Into Associative Memory
SAND	Store Logical AND Into Associative Memory
SANDM	Store Logical AND Masked Into Associative Memory
SANDN	Store Logical AND Complemented Into Associative Memory
SANDNM	Store Logical AND, Complemented, Masked Into Associative Memory
SC	Store Common Register Into Associative Memory
SCW	Store Common Register Into Associative Word

S

Format

Store Response Store Into Associative Memory

This instruction will store the content of the designated response store register, rs, into the specified bit column of enabled associative memory. The content of the source response store is not affected, and the original content of the bit column is destroyed.

Label	Command	Argument	Comment
symbol	<u>S</u>	$\frac{rs, \left[\underline{a} \pm \mathbf{k} \right]}{\left[\underline{r} \right]}$	

• Label Any valid symbol or blank.

S

Command

• Argument Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

The source response store register.

Valid entries:

X - X response store Y - Y response store M - M response store

• a±k

• r

• rs

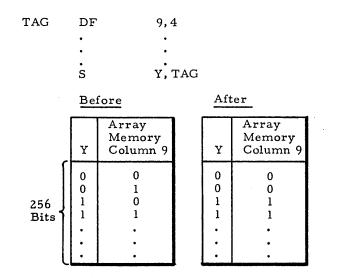
'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all words of enabled associative memory. The value of $a\pm k$ should be $0 \le a\pm k \le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

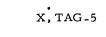
FP1	Field Pointer 1
FPl+	Field Pointer 1 with a post-increment
FPl-	Field Pointer 1 with a post-decrement
FP2	Field Pointer 2
FP2+	Field Pointer 2 with a post-increment
FP2-	Field Pointer 2 with a post-decrement
FP3	Field Pointer 3
FP3+	Field Pointer 3 with a post-increment
FP3-	Field Pointer 3 with a post-decrement

Example 1



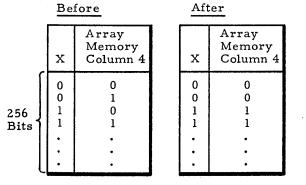
Example 2

TAG DF 9,4 • • . s



.

.



2-81

Store Response Store Masked Into Associative Memory

This instruction will store the content of the designated response store register, rs, into the specified bit column of enabled associative memory in all words whose M response store bit is set. The content of the source response store is not affected, and the original content of the bit column is destroyed in those words of associative memory whose M response store bit is set.

Format	Label	Command	Argument	Comment
	symbol	SM	<u>rs, a</u> ±k r	

Label Any valid symbol or blank.

SM

Command

Argument

Two entries are required. The first entry is the source; the second entry is the destination. As shown there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

The source response store register.

Valid entries:

X - X response store Y - Y response store M - M response store

● ● a±k

• • r

• rs

'a' may be a constant or a symbol: k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all selected words of enabled associative memory. The value of $a\pm k$ should be $0\le a\pm k\le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FP1 Field Pointer 1 Field Pointer 1 with a post-increment FP1+ FP1-Field Pointer 1 with a post-decrement Field Pointer 2 FP2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3- Field Pointer 3 with a post-decrement

SM

Example

TAG

DF	152,40
•	•
•	•
SM	X, TAG

Before

·	x	М	Array Memory Col 152
256 Bits	0 0 0 1 1 1 1	0 0 1 1 0 0 1 1	0 1 0 1 0 1 0 1

After

x	М	Array Memory Col 152		
0	0	0		
0	0 0	1		
0	1	0		
0 0 0 1 1	1 1	0		
1	0	0		
	0	1		
1 1	1	1		
1	1	1		
•	•			
•	•			
•	•	•		

Store Complement Into Associative

This instruction will store the one's complement of the value of the designated response store register, rs, into the specified bit column of enabled associative memory. The content of the source response store is not affected, and the original content of the bit column is destroyed.

Label	Command	Argument	Comment
symbol	<u>SN</u>	<u>rs</u> , [<u>a</u> ±k] r	

• Label

Format

Command

• Argument

Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

The source response store register.

Valid entries:

SN

Y - Y response store M - M response store

Any valid symbol or blank.

● ● a±k

• • r

• • rs

'a' may be a constant or a symbol: k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all words of enabled associative memory. The value of $a\pm k$ should be $0\le a\pm k\le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FP1	Field Pointer 1
FP1+	Field Pointer 1 with a post-increment
FP1-	Field Pointer 1 with a post-decrement
FP2	Field Pointer 2
FP2+	Field Pointer 2 with a post-increment
FP2-	Field Pointer 2 with a post-decrement
FP3	Field Pointer 3
FP3+	Field Pointer 3 with a post-increment
FP3-	Field Pointer 3 with a post-decrement

SN

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

DF 9,4 • . • ٠ . SN M, TAG Before

х

1 1 0

0

•

٠

•

TAG

ore		Aft	er	
М	Array Memory Col 9	x	м	Array Memory Col 9
0 0 1 1	0 1 0 1 •	d s t r o y e d	0 0 1	1 0 0

SN

Note

Store Complement Masked Into Associative Memory

This instruction will store the one's complement of the value of the Y response store register into the specified bit column of enabled associative memory in all words of associative memory whose M response store bit is set. The content of the Y response store is not affected and the original content of the bit column is destroyed in those words of associative memory whose M response store bit it set.

Format	Label	Command	Argument	Comment
	symbol	<u>SNM</u>	Y a±k r	

• Label Any valid symbol or blank.

• Command SNM

• Argument Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

• Y Required and only valid entry.

🛛 🖕 a±k

• • r

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all selected words of enabled associative memory. The value of a±k should be $0 \le a \pm k \le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FPI Field Pointer 1 Field Pointer 1 with a post-increment FP1+ FP1-Field Pointer 1 with a post-decrement Field Pointer 2 FP2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement

SNM

SNM

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

> • •

Y, TAG

 DF 152,40 • . SNM

TAG

	Bei	ore			After				
	x	Y	м	Array Memory Col. 152		x	Y	М	Array Memory Col. 152
256 Bits		0 0 0 1 1 1 1	0 0 1 1 0 0 1 1	0 1 0 1 0 1 0 1 •		d s t r o y e d	0 0 0 1 1 1 1	0 0 1 1 0 0 1 1 1	0 1 1 0 1 0 0 0 • •

Store Logical Inclusive OR Into Associative Memory

This instruction will logical inclusive OR the contents of the designated response store register, rs, and the bit column of enabled associative memory, and store the resultant value into the bit column. The content of the source response store, rs, is not affected, and the original content of the bit column is destroyed.

Label	Command	Argument	Comment
symbol	<u>SOR</u>	<u>rs,[a</u> ±k] <u>r</u>]	

Label

Format

SOR

Any valid symbol or blank.

SOR

Command

• Argument Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

The source response store register.

Valid entries: Y - Y response store M - M response store

● ● a±k

• • r

• • rs

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all words of enabled associative memory. The value of $a\pm k$ should be $0\le a\pm k\le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FPI Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post decrement FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement SOR

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

TAG	\mathbf{DF}		9,4				
	•		•				
	•		•				
	soi	R	Y, TAG-1	1			
	Bef	ore			Aft	er	
	x	Y	Array Memory Column 8		x	Y	Array Memory Column 8
ſ	1	0	0		d	0	0
	1	0	1		е	0	1
	0	1	0		s	1	1
256	0	1	1		t	1	1
Bits J	•	•	· · ·		r	•	
	•	•			0	•	•
	•	•	•		У	•	•
				-	e		
					d		

١

Store Logical Inclusive OR, Masked

This instruction will logical inclusive OR the contents of the Y response store register and the designated bit column of enabled associative memory, and store the resultant value into the designated bit column in all words of associative memory whose M response store bit is set. The content of the Y response store is not affected and the original content of the bit column is destroyed in those words of associative memory whose M response store bit is set.

Format	Label	Command	Argument	Comment
	symbol	<u>SORM</u>	<u>Y</u> , <u>a</u> +k <u>r</u>	

• Label Any valid symbol or blank.

• Command SORM

SORM

• r

• Argument Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

- ••Y Required and only valid entry.
- a[±]k
 'a' may be a constant or a symbol; k is an optional constant modifier.
 k is legal only when 'a' is present as a symbol. If 'a' was defined as
 a field via a DF instruction, the most-significant bit position is the
 value used. This term represents a destination bit position in all selected
 words of enabled associative memory. The value of a±k should be 0≤a±k≤255.
 - A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FP1 Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment Field Pointer 2 with a post-decrement FP2-FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement SORM

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

SORM Y,200

١

	Be	fore			After			
	x	Y	М	Array Memory Col 200	x	Y	М	Array Memory Col.200
256 Bits		0 0 0 1 1 1 1	0 0 1 1 0 0 1 1 1	0 1 0 1 0 1	d s t y e d	0 0 0 1 1 1 1	0 0 1 0 0 1 1	0 1 0 1 1 1 1

SORN Store Logical Inclusive OR Complemented Into Associative Memory This instruction will logical inclusive OR the one's complement of the contents of the designated response store register, rs, with the specified bit column of enabled associative memory, and store the resultant value into the designated bit column of all words. The content of the source response store is not affected, and the original content of the bit column is destroyed.

Label	Command	Argument	Comment
symbol	<u>SORN</u>	<u>rs</u> ,[<u>a</u> ±k] <u>r</u>]	

• Label

• Command SORN

Argument

Format

Two entries are required. The first entry is the source; the second entry is the destination. As shown there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

The source response store register.

Valid entries:

Y - Y response store M - M response store

Any valid symbol or blank.

🔹 🖕 a±k

• • r

• rs

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all words of enabled associative memory. The value of $a\pm k$ should be $0\le a\pm k\le 255$.

A field pointer register, which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FP1 Field Pointer 1
FP1+ Field Pointer 1 with a post-increment
FP1- Field Pointer 1 with a post-decrement
FP2 Field Pointer 2
FP2+ Field Pointer 2 with a post-increment
FP2- Field Pointer 3 with a post-decrement
FP3+ Field Pointer 3 with a post-increment
FP3- Field Pointer 3 with a post-decrement

SORN

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

	SORN			Μ	ι , Χ'Ο			
	Before					After		
	x	М	Array Memory Column 0		x	М	Array Memory Column 0	
256 Bits	1 1 0 0	0 0 1	0 1 0 1		d s t r o y e d	0 0 1	1 0 1 • •	

١

Store Logical Inclusive OR, Complemented, Masked Into Associative Memory

SORNM

This instruction will logical inclusive OR the one's complement of the contents of the Y response store register with the specified bit column of enabled associative memory. The resultant value is then stored into the designated bit column in all words of associative memory whose M response store bit is set. The content of the Y response store is not affected, and the original content of the bit column is destroyed in those words of associative memory whose M response store bit is set.

Format	Label	Command	Argument	Comment					
	symbol	<u>SORNM</u>	$\frac{\underline{Y}}{\underline{a}} = \underline{x}$						
• Label	Any valid	symbol or blank.							
• Command	SORNM								
• Argument	Two entries are required. The first entry is the source; the second entry is the destination. As shown there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.								
• • Y	Required a	and only valid entr	·y.						
• • a±k	k is legal as a field value used	only when 'a' is p via a DF instructi . This term repr ords of enabled as	resent as a symbol on, the most-signif esents a destinatio	tional constant modifier. . If 'a' was defined icant bit position is the n bit position in all The value of a±k should					
• • r	decrement is stored i	ed. If this form i	this register. Thi	remented or post- se store register, rs, s register contains the					
	Valid entr FP1 FP1- FP2 FP2+ FP2- FP3	Field Pointer 1 Field Pointer 1 Field Pointer 1 Field Pointer 2 Field Pointer 2	with a post-increm with a post-decrem with a post-increm with a post-decrem	ent					

Field Pointer 3

Field Pointer 3 with a post-increment

Field Pointer 3 with a post-decrement

FP2-FP3

FP3+ FP3-

2-94

SORNM

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

SORNM Y,0

۱

Before

After

	x	Y	М	Array Memory Col. 0		x	Y	М	Array Memory Col 0
256 Bits	-	0	0	0		d	0	0	0
	-	0	0	1	1	е	0	0	1
	-	0	1	0		s	0	1	1
		0	1	1		t	0	1	1
		1	0	0		r	1	0	0
	-	1	0	1		0	1	0	1
	- 1	1	1	0		у	1	1	0
	- 1	1	1	1		е	1	1	1
	•		•	• •		d	•	•	•
	•	•	•	•			•	•	•
	•	•	•	•			•	•	•

SAND Store Logical AND Into Associative Memory

This instruction will logical AND the contents of the designated response store register, rs, with the specified bit column of enabled associative memory, and store the resultant value into the bit column. The content of the source response store is not affected, and the original content of the bit column is destroyed.

Format	Label	Command	Argument	Comment
	symbol	SAND	$\frac{rs}{\left[\frac{a}{r}\pm k\right]}$	

- Label Any valid symbol or blank.
- Command SAND

• • r

- Argument Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.
- • rs The source response store register.

Valid entries:

Y - Y response store M - M response store

- a±k
 'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all words of enabled associative memory. The value of a±k should be 0≤a±k≤255.
 - A field pointer register, which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FPI Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 Field Pointer 2 with a post-increment FP2+ FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement

SAND

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

TAG \mathbf{DF} • .

> . SAND Y, TAG

9,4

.

.

Array Memory Column 9

0

1

0

1

•

•

•

Before

Y

0

0

1

1

•

٠

.

Х

1

1

0

0

•

٠

•

٦.

256

Bits

After Array Memory х Y Column 9 d 0 0 0 0 е s 0 1 t 1 1 r ٠ • o y e d . •

•

•

Store Logical AND Masked Into Associative Memory

This instruction will logical AND the contents of the Y response store register and the specified bit column of enabled associative memory. The resultant value will then be stored into the designated bit column in all words of associative memory whose M response store bit is set. The content of the Y response store is not affected, and the original content of the bit column is destroyed in those words of associative memory whose M response store bit is set.

Format	Label	Command	Argument	Comment
	symbol	SANDM	$\frac{Y_{j}}{\frac{a \pm k}{r}}$	

• Label Any valid symbol or blank.

• Command SANDM

• Argument Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

••Y Required and the only valid entry.

• • a±k

• r

SANDM

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all selected words of enabled associative memory. The value of a±k should be $0 \le a \pm k \le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FPI Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement SANDM

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

Example

Y,10 SANDM

۱

	Be	fore			Aft	er		
	x	Y	M	Array Memory Column 10	x	Y	М	Array Memory Column 10
256 Bits		0 0 0 1 1 1 1	0 0 1 1 0 0 1 1 1	0 1 0 1 0 1 0 1	d estroyed	0 0 0 1 1 1 1	0 0 1 1 0 0 1 1	0 1 0 0 1 0 1

Store Logical AND Complemented Into Associative Memory

This instruction will logical AND the one's complement of the contents of the designated response store register, rs, with the specified bit column of enabled associative memory. The resultant value is then stored into the designated bit column in all words of associative memory. The content of the Y response store is not affected and the original content of the bit column is destroyed.

Label	Command	Argument	Comment
symbol	<u>SANDN</u>	$\frac{rs}{r} \begin{bmatrix} a \pm k \\ r \end{bmatrix}$	

- Label Any valid symbol or blank.
- Command SANDN
- Argument

SANDN

Format

Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

The source response store register.

Valid entries:

Y - Y response store M - M response store

• • a±k

• r

• rs

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all words of enabled associative memory. The value of $a\pm k$ should be $0 \le a\pm k \le 255$.

A field pointer register which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is stored indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FP1 Field Pointer 1 FP1+ Field Pointer 1 with a post-increment FP1-Field Pointer 1 with a post-decrement FP2 Field Pointer 2 Field Pointer 2 with a post-increment FP2+ FP2-Field Pointer 2 with a post-decrement FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement

SANDN

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.



TAG DF 9,4 • • • ٠ . SANDN Y, TAG After Before Array Memory Column 9 Array Memory х Y Column 9 х Y 0 d 0 1 1 0 0 0 0 0 1 е 0 1 s t 0 1 0 1 1 0 256 Bits 1 1 r • • • • • o ٠ ٠ • . ٠ у е d

SANDNM

Store Logical AND, Complemented, Masked Into Associative Memory

This instruction will logical AND the one's complement of the contents of the Y response store register with the specified bit column of enabled associative memory. The resultant value is then stored into the designated bit column in all words of associative memory whose M response store bit is set. The content of the Y response store is not affected, and the original content of the bit column is destroyed in those words of associative memory whose M response store bit is set.

Format Label Command Argument Comment symbol <u>SANDNM</u> <u>Y</u>, <u>a</u>±k <u>r</u>

Any valid symbol or blank.

Label

• Command SANDNM

• Argument

• Y

• • r

🛛 🖕 a±k

Two entries are required. The first entry is the source; the second entry is the destination. As shown, there are two distinct ways of specifying the destination bit column. The brackets are not a part of the argument field terms.

Required and only valid entry.

'a' may be a constant or a symbol; k is an optional constant modifier. k is legal only when 'a' is present as a symbol. If 'a' was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a destination bit position in all selected words of enabled associative memory. The value of $a\pm k$ should be $0 \le a\pm k \le 255$.

A field pointer register, which may be post-incremented or postdecremented. If this form is used, the response store register, rs, is loaded indirectly through this register. This register contains the address of the destination bit column.

Valid entries:

FP1 Field Pointer 1 FP1+ Field Pointer 1 with a post-increment Field Pointer 1 with a post-decrement FP1-FP2 Field Pointer 2 FP2+ Field Pointer 2 with a post-increment Field Pointer 2 with a post decrement FP2-FP3 Field Pointer 3 FP3+ Field Pointer 3 with a post-increment FP3-Field Pointer 3 with a post-decrement

SANDNM

Note

The original content of the X response store register is destroyed when this multiple instruction is executed.

SANDMN Y, 0

	Be	fore		After					
	x	Y	М	Array Memory Column 0		x	Y	М	Array Memory Column 0
256 Bits		0 0 0 1 1 1 1	0 0 1 1 0 0 1 1 1	0 1 0 1 0 1 0 1		d s t y e d	0 0 0 1 1 1 1	0 0 1 1 0 0 1 1	0 1 0 1 0 1 0 0

Store Common Register into Associative Memory

This instruction will store a Common register field, a₁, into a field, a₂, of all words of enabled associative memory whose M response store bit is set.

Label	Command	ommand Argument	
symbol	<u>SC</u>	<u>a</u> 1, a2	

Label

Format

SC

Any valid symbol or blank.

SC

Command

• Argument

Two entries are required. The first entry is the source, a field in the Common register; the second entry is the destination, a field in words of associative memory.

• • ^a1,^a2

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

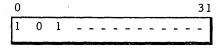
Note 1 If the Common register field length is less than the associative memory field length, a W (warning flag) is noted on the listing. In this case, the Common register field will be stored right justified into the associative memory field. Note 2 If the Common register field length is greater than the associative memory field length, a T (truncation flag) is noted on the listing. In this case, the most significant bits of the Common register field are truncated. Note 3 The content of the X response store register is destroyed. Also, the following field definition registers are used: FP1, FP2, and FL1.

Example

(0, 3), (10, 3)

		Afte	r Exe	cution			
			Array Memory Bit Column				
		М	10	11	12		
ſ	ſ	1	1	0	1		
		1	1	0	1		
		1	1	0	1		
		Ō	unchanged				
256		0	unchanged				
Bits		1	1	0	1		
		1	1	0	1		
		•	•	•			
		•	•	•	•		
		•	•	•	•		

Common Register



SC

SCW

Format

• Label

Store Common Register into Associative Word

This instruction will store a Common register field, a_1 , into a field, a_2 , of one word of associative memory whose address is in the link pointer (FP1, FP2). All other words in the associative memory remain unchanged.

Label	Command	Argument	Comment
symbol	SCW	<u>a1,a2</u>	

Any valid symbol or blank.

SCW

Command

• Argument

• • a₁, a₂

Two entries are required. The first entry is the source, a field in the Common register; the second entry is the destination, a field in a word of associative memory.

There are two ways of denoting a field expression:

1) a₁ or a₂ may be in the form b±i

> where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a₁ or a₂ may be in the form (b, i)±j

> where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note 1 The link pointer (FP1 and FP2 registers) must be loaded with the address of the particular word of associative memory prior to execution of the instruction. Loading the link pointer is generally accomplished by the use of the FIND, STEP, or RESVFST instruction.

Note 2 If the Common register field length is less than the associative memory field length, a W (warning flag) is noted on the listing. In this case, the Common register field will be stored right justified into the associative memory field.

- Note 3 If the Common register field length is greater than the associative memory field length, a T (truncation flag) is noted on the listing. In this case, the most significant bits of the Common register field are truncated.
- Note 4 The content of the X response store register is destroyed. The content of the Y response store register will be destroyed if field-alignment shifting is required.

scw

Example

FIND SCW

(0,3),(10,3)

Array Memory Bit Column

11

.

•

•

٠

.

•

.

12

.

.

•

٠

•

•

.

10

.

•

•

•

•

•

•

.

FPl	FP2
ADDR of	WORD 3



Y

0

0

0

1

1

•

•

ò

х

1 0

0

1

0

٠

•

. 0

Aft	er			
			ay Mei C o lum	
х	Y	10	11	12
d	d	ur	chang	ed
е	е	un	chang	ed
s	s	l un	chang	ed
t	t	1	0	
r	r	un	ichang	ed
0	0	•	•	•
y	у	•	•	•
е	e	•		•
d	d	un	chang	ed

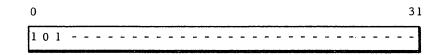
Common Register

Word 0 Word 1 Word 2 Word 3 Word 4

.

.

Word 255



2-107

SEARCHES

These associative instructions allow the programmer to search for a particular set of conditions in associative memory. All instructions dealing with response store registers and/or associative memories affect only those associative array memory modules enabled via the Array Select register. The response store registers and associative array memory modules disabled via the Array Select register remain unchanged. Except for MAXF and MINF, the most-significant bit of all fields is considered to be the sign bit.

Mnemonic	Instructions
FIND	Find the First Bit Set in Y Response Store
STEP	Step to First Y Set and Clear It
RESVFST	Step to First Y Set and Clear All Others
EQC	Equal to Common Register Field
EQF	Equal Fields
NEC	Not Equal To Common Register Field
NEF	Not Equal Fields
GTC	Greater Than Common Register Field
GTF	Greater Than Fields
GEC	Greater Than or Equal To Common Register Field
GEF	Greater Than or Equal Fields
LTC	Less Than Common Register Field
LTF	Less Than Fields
LEC	Less Than or Equal Common Register Field
LEF	Less Than or Equal Fields
MAXF	Maximum Fields
MINF	Minimum Fields

FIND

Find the First Bit Set in Y Response Store

The instruction loads FPl with the array address of the first array module containing a Y response store register bit set to one. FP2 is then loaded with the bit address of the first Y response store register bit set to one.

Format	Label	Command	Argument	Comment
	symbol	FIND		
• Label	Any valid s	symbol or blank.		
• Command	FIND			

• Argument No entries required.

STEP

Step to First Y Set and Clear It

This instruction loads FPl with the array address of the first array module containing a Y response store register bit set to one. FP2 is then loaded with the bit address of the first Y response store register bit set to one. This selected first bit will than be cleared to zero.

Format	Label	Command	Argument	Comment
	symbol	STEP		
• Label	Any valid s	ymbol or blank.		
• Command	STEP			
• Argument	No entries	required.		

RESVFST Step to First Y Set and Clear All Others This instruction loads FPl with the array address of the first array module containing a Y response store register bit set to one. FP2 is then loaded with the bit address of the first Y response store register bit set to one. This selected first bit will remain set to one, but all other bits in the Y response store register are cleared to zero.

Format 🐁	Label	Command	Argument	Comment
	symbol	RESVFST		
• Label	Any valid	symbol or blank.		
• Command	RESVFST			

١

• Argument No entries required.

EQC

Equal to Common Register Field

For the field a₁ in each word of associative memory, this instruction will set the corresponding Y response store register bit if, and only if, the following is true:

Conditions

- 1) The particular array is enabled in the Array Select register so it may participate in the search.
- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a₁ is equal to Common register field a₂.

Format	Label	Command	Argument	Comment
	symbol	EQC	<u>a1,a2</u>	

Label Any valid symbol or blank.

- Command EQC
- Argument Two entries are required. The first entry, a₁, is a field in associative memory; the second entry, a₂, is a field in the Common register. The lengths of the fields must be equal and greater than one.
- • ^a1, ^a2

Note

There are two ways of denoting a field expression.

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most significant bit position.

2) $a_1 \text{ or } a_2 \text{ may be in the form}$

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after the search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a_1 .

EQF	•	Equal Fields			
		This instruction will set the Y response store register bit for each word of associative memory if, and only if, the following is true:			
Conditions	l) The may	1) The particular array is enabled in the Array Select register so it may participate in the search.			
		M response store icipating in the se		et for the particular word	
• •	3) The search criteria is met; array field a_1 of word _i is equal to array field a_2 of word _i .				
Format	Label	Command	Argument	Comment	
	symbol	EQF	$\frac{a_1,a_2}{a_1}$		
• Label	Any valid	symbol or blank.			

EQF

Argument

• • ^a1, ^a2

Command

Two entries are required. Both entries represent fields in associative memory that are compared with each other. The lengths of the fields must be equal and greater than one.

There are two ways of denoting a field expression:

 a_1 or a_2 may be in the form 1)

b±i

where b must be symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a₁ or a₂ may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

Register values after search:

- 1) FL1 Zero
- FP1 Address of the most-significant bit of a₂. 2)
- FP3 Address of the most-significant bit of a₁. 3)

The X response store register is utilized.

NEC	Not Equal To Common Register Field				
	For the field	ld a _l in each word	l of associative me	mory, this instruction	
			response store re	gister bit if, and only if,	
	the followin	ng is true:			
Conditions	1) The particular array is enabled in the Array Select register so it				
Conditions	• •	participate in the		ray Select register so it	
	2) The M response store register bit is set for the particular word participating in the search.				
	3) The sregis	search criteria is ter field a ₂ .	met; array field a	\mathfrak{a}_1 is not equal to Common	
Format	Label	Command	Argument	Comment	
	symbol	NEC	<u>a₁, a₂</u>		
• Label	Any valid s	ymbol or blank.			
	·	-			
• Command	NEC				
• Argument	Two entrie	s are required.	The first entry, a ₁	, is a field in associative	
			-	e Common register. The	
	lengths of f	the fields must be	equal and greater	than one.	
• • a ₁ ,a ₂	There are	two ways of denot	ing a field express	ion:	
1, 2	l) a _l or	a, may be in the	form		
	. 1	z - ² b±i			
			1 1		
				otional constant modifier. a DF instruction. b	
			-	on and the number of	
	-		-	ner the Common register	
	or as	sociative memory	y. The optional co	nstant modifier, i,	
	modi	fies only the most	significant bit pos	sition.	
	2) a ₁ or	a ₂ may be in the	form		
	t	- (b,i)±j			
	wher	e b may be a cons	tant or a symbol a	nd represents the most-	
	-	-		as defined as a field via	
			-	ificant bit position is the	
				presents the number of s an optional constant.	
	contiguous bits occupied by the field. j is an optional constant,				

Register values after the search:

1) FL1 - Zero

Note

2) FP1 - Address of the most significant bit of a₂.

modifying only the most-significant bit position of the field.

3) FP3 - Address of the most significant bit of a_1 .

NEF	Not Equal Fields This instruction will set the Y response store register bit for each word				
		tive memory if, an	-	5	
Conditions		 The particular array is enabled in the Array Select register so it may participate in the search. 			
		M response store cipating in the sea		t for the particular word	
	3) The search criteria is met; array field a_1 of word is not equal to array field a_2 of word i.				
Format	Label	Command	Argument	Comment	
	symbol	NEF	$\frac{a_1}{a_2}$		
• Label	Any valid	symbol or blank.			

• Command NEF

• Argument Two entries are required. Both entries represent fields in associative memory that are compared with each other. The lengths of the fields must be equal and greater than one.

• • ^a1, ^a2

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) $a_1 \text{ or } a_2 \text{ may be in the form}$

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a_2 .
- 3) FP3 Address of the most-significant bit of a_1 .

The X response store register is utilized.

Note

GTC

Greater Than Common Register Field

For the field a₁ in each word of associative memory, this instruction will set the corresponding Y response store register bit if, and only if, the following is true:

Conditions

Format

1) The particular array is enabled in the Array Select register so it may participate in the search.

- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a₁ is greater than Common register field a₂.

Label	Command	Argument	Comment
symbol	GTC	<u>a</u> 1 . a2	*

Label Any valid symbol or blank.

- Command GTC
- Argument Two entries are required. The first entry, a_1 , is a field in associative memory; the second entry, a_2 , is a field in the Common register. The lengths of the fields must be equal and greater than one.
- • ^a1,^a2

Note

There are two ways of denoting a field expression:

1) $a_1 \text{ or } a_2 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after the search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a_2 .
- 3) FP3 Address of the most-significant bit of a_1 .

Greater Than Fields

This instruction will set the Y response store register bit for each word of associative memory if, and only if, the following is true:

Conditions

GTF

1) The particular array is enabled in the Array Select register so it may participate in the search.

- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a_1 of word_i is greater than array field a_2 of word_i.

Format	Label	Command	Argument	Comment
	symbol	GTF	$\frac{a}{1}$	

• Label Any valid symbol or blank .

• Command GTF

• Argument Two entries are required. Both entries represent fields in associative memory that are compared with each other. The lengths of the fields must be equal and greater than one.

• • ^a1,^a2

There are two ways of denoting a field expression:

1) $a_1 \text{ or } a_2 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant modifying only the most-significant bit position of the field.

Register values after search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a_1 .

The X response store register is utilized.

Note

GEC

Greater Than or Equal To Common Register Field

For the field a₁ in each word of associative memory, this instruction will set the corresponding Y response store register bit if, and only if, the following is true:

Conditions

Format

- 1) The particular array is enabled in the Array Select register so it may participate in the search.
- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a_1 is greater than or equal to Common register field a_2 .

Label	Command	Argument	Comment
symbol	GEC	$\underline{a}_1, \underline{a}_2$	

Label Any valid symbol or blank.

• Command GEC

Argument

Two entries are required. The first entry, a_1 , is a field in associative memory; the second entry, a_2 , is a field in the Common register. The lengths of the fields must be equal and greater than one.

• • ^a1, ^a2

Note

There are two ways of denoting a field expression:

1) $a_1 \text{ or } a_2 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after the search:

- 1) FL1 Zero
- 2) FPl Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a_1 .

Greater Than or Equal Fields

GEF

This instruction will set the Y response store register bit for each word of associative memory if, and only if, the following is true:

Conditions

1) The particular array is enabled in the Array Select register so it may participate in the search.

- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a₁ of word_i is greater than or equal to array field a₂ of word_i.

Format	Label	Command	Argument	Comment
	symbol	GEF	$\frac{a_1, a_2}{a_1, a_2}$	

• Label Any valid symbol or blank.

• Command GEF

• Argument Two entries are required. Both entries represent fields in associative memory that are compared with each other. The lengths of the fields must be equal and greater than one.

• • ^a1,^a2

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

Register values after search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a_1 .

The X response store register is utilized.

LTC

Less Than Common Register Field

For the field a₁ in each word of associative memory, this instruction will set the corresponding Y response store register bit if, and only if, the following is true:

Conditions

Format

- 1) The particular array is enabled in the Array Select register so it may participate in the search.
- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a₁ is less than Common register field a₂.

Label	Command	Argument	Comment
symbol	LTC	$\frac{a_{1,a_2}}{a_1$	

Label Any valid symbol or blank.

• Command LTC

Argument

• • a₁,a₂

Note

Two entries are required. The first entry, a_1 , is a field in associative memory; the second entry, a_2 , is a field in the Common register. The lengths of the fields must be equal and greater than one.

There are two ways of denoting a field expression:

1) $a_1 \text{ or } a_2 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after the search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a₁.

Less Than Fields

This instruction will set the Y response store register bit for each word of associative memory if, and only if, the following is true:

Conditions

LTF

1) The particular array is enabled in the Array Select register so it may participate in the search.

- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a₁ of word_i is less than array field a₂ of word_i.

Format	Label	Command	Argument	Comment
	symbol	LTF	$\frac{a_1, a_2}{a_2}$	

Label

. .

LTF

Any valid symbol or blank.

Command

Argument

• a1,a2

Two entries are required. Both entries represent fields in associative memory that are compared with each other. The lengths of the fields must be equal and greater than one.

There are two ways of denoting a field expression:

1) $a_1 \text{ or } a_2 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after search:

- 1) FL1 Zero
- 2) FP1 Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a_1 .

The X response store register is utilized.

Note

LEC

Less Than or Equal Common Register Field

For the field a₁ in each word of associative memory, this instruction will set the corresponding Y response store register bit if, and only if, the following is true:

Conditions

- 1) The particular array is enabled in the Array Select register so it may participate in the search.
- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a_1 is less than or equal to Common register field a_2 .

Label	Command	Argument	Comment
symbol	LEC	$\frac{a_{1,a_2}}{a_1,a_2}$	

Label

Format

Any valid symbol or blank.

• Command LEC

Argument

• a₁, a₂

Note

Two entries are required. The first entry, a_1 , is a field in associative memory; the second entry, a_2 , is a field in the Common register. The lengths of the fields must be equal and greater than one.

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) $a_1 \text{ or } a_2 \text{ may be in the form}$

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after the search:

- 1) FL1 Zero
- 2) FPl Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a₁.

Less Than or Equal Fields

This instruction will set the Y response store register bit for each word of associative memory if, and only if, the following is true:

Conditions

LEF

1) The particular array is enabled in the Array Select register so it may participate in the search.

- 2) The M response store register bit is set for the particular word participating in the search.
- 3) The search criteria is met; array field a₁ of word_i is less than or equal to array field a₂ of word_i.

Format	Label	Command	Argument	Comment
· ·	symbol	LEF	$\frac{a_1, a_2}{a_1, a_2}$	

Label

Any valid symbol or blank.

LEF

Command

Argument

• • a₁,a₂

Two entries are required. Both entries represent fields in associative memory that are compared with each other. The lengths of the fields must be equal and greater than one.

There are two ways of denoting a field expression:

1) $a_1 \text{ or } a_2 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) $a_1 \text{ or } a_2 \text{ may be in the form}$

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Register values after search:

- 1) FL1 Zero
- 2) FPl Address of the most-significant bit of a₂.
- 3) FP3 Address of the most-significant bit of a_1 .

The X response store register is utilized.

2-122

Note

MAXF

• a

Maximum Fields

This instruction will compare a field of those words of the associative memory whose M response store bit is set. The Y response store will be set for the word(s) containing the field with the maximum (greatest) unsigned value. The Y response store for all other words will be cleared to zero.

Format	Label	Command	Argument	Comment
	symbol	MAXF	<u>a</u>	

- Label Any valid symbol or blank.
- Command MAXF
- Argument One entry is required.

There are two ways of denoting a field expression:

1) 'a' may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) 'a' may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the most significant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

- Register values after search:
 - 1) FL1 Zero
 - 2) FP3 Address of the least-significant bit of 'a'.

The X response store register is utilized.

MINF

Minimum Fields

This instruction will compare a field of those words of associative memory whose M response store bit is set. The Y response store will be set for the word(s) containing the field with the minimum (least) unsigned value. The Y response store for all other words will be cleared to zero.

Format	Label	Command	Argument	Comment
	symbol	MINF	a	

Label Any valid symbol or blank.

MINF

- Command
- Argument
 One entry is required.

• • a

There are two ways of denoting a field expression:

1) 'a' may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memroy. The optional constant modifier, i, modifies only the most-significant bit position.

2) 'a' may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the most significant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

- Register values after the search:
- 1) FL1 Zero
- 2) FP3 Address of the least-significant bit of 'a'.

The X response store register is utilized.

MOVES

This group of associative instructions allows the programmer to move an array memory field to another array memory field within the same word of associative memory.

This group of instructions will operate only on those associative array memory modules (including response store registers) enabled via the Array Select register. Also, only those words within enabled associative array memory modules whose M response store register bit is set will participate in the execution of the instructions in this group. The most significant bit of all fields is considered to be the sign bit.

Mnemonic	Instructions
MVF	Move Field
MVCF	Move the One's Complement of a Field
MVNF	Move the Negative of a Field
MVAF	Move the Absolute Value of a Field
INCF	Move Field with Increment
DECF	Move Field with Decrement

2-125

Move Field

This instruction will move the contents of field a_1 into field a_2 within the same word for each word of enabled associative memory whose M response store bit is set. The content of the source field is not affected unless overlaid by the destination field. The original content of the destination field is destroyed.

Label	Command	Argument	Comment
symbol	MVF	<u>a</u> 1, <u>a</u> 2	

• Label Any valid symbol or blank.

MVF

Command

• Argument Two entries are required. The first entry is the source; the second entry is the destination. Both entries represent fields within the same word of associative memory.

• • a₁, a₂

MVF

Format

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b, i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

The X response store, FP1, FP3, and FL1 registers are used by this instruction.

Example

MVF

(2,3),(10,3)

Before

After

	A:					У
1/1	2	3	4	10	11	12
0	0	0	0	1	1	0
0	0	0	1	1	0	1
1	0	1	0	1	0	0
1	1	1	1	0	1	1
•	•		•	.	•	•
•	•	•	•	•	•	•
•	•	•	•	•	•	•
	0	M 2 0 0 0 0 1 0	Bi 2 3 0 0 0 0 0 0 1 0 1	Bit C 2 3 4 0 0 0 0 0 0 0 1 1 0 1 0	M Bit Column 2 3 4 10 0 0 0 1 1 0 0 0 1 1 1 0 1 0 1	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$

	A:			Men olu	nory mn	r
М	2	3	4	10	11	12
0	0	0	0	1	1	0
0	0	0	1	1	0	1
1	0	1	0	0	1	0
1	1	1	1	1	1	1
•	•	•	•	•	•	•
 •	•	•	•	.	•	•
 •	•	•	•	•	•	•

2-127

Move the One's Complement of a Field

This instruction will move the one's complement of the contents of field a_1 into field a_2 within the same word for each word of enabled associative memory whose M response store bit is set. The content of the source field is not affected unless overlaid by the destination field. The original content of the destination field is destroyed.

Format	Label	Command	Argument	Comment
	symbol	MVCF	$\frac{a_{1,a_2}}{a_{1,a_2}}$	

Label

MVCF

Any valid symbol or blank.

MVCF

Command

Argument

Two entries are required. The first entry is the source; the second entry is the destination. Both entries represent fields within the same word of associative memory.

• • ^a1^{, a}2

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±'i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b, i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

The X response store, FP1, FP3, and FL1 registers are used by this instruction.

Note

MVCF

Example

MVCF (2,3),(20,3)

After Execution

		Array Memory Bit Column						
	M	2	3	4	20	21	22	
256 Bits	1	0	0	0	1	1	1	
	1	0	0	1	1	1	0	
	1	0	1	0	1	0	1	
	1	0	1	1	1	0	0	
	1	1	0	0	0	1	1	
	1	1	0	1	0	1	0	
	1	1	1	0	0	0	1	
	1	1	1	1	0	0	0	

Move the Negative of a Field

Any valid symbol or blank.

MVNF

This instruction will move the two's complement of the contents of field a1 into field a2 within the same word for each word of enabled associative memory whose M response store bit is set. The content of the source field is not affected unless overlaid by the destination field. The original content of the destination field is destroyed.

Format	Label	Command	Argument	Comment
	symbol	MVNF	$\frac{a}{1,a}$ 2	

• Label

Command

• Argument

MVNF

Two entries are required. The first entry is the source; the second entry is the destination. Both entries represent fields within the same word of associative memory.

• • ^a1,^a2

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)**±j**

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

The X response store, Y response store, FP1, FP3, and FL1 registers are used by this instruction.

Example

MVNF (5, 3), (15, 3)

After Execution

		Array Memory Bit Column					
	М	5	6	7	15	16	17
ſ	1	0	0	0	0	0	0
	1	0	0	1	1	1	1
	1	0	1	0	1	1	0
	1	0	1	1	1	0	1
	1	1	0	0	1	0	0*
$\left \right $	1	1	0	1	0	1	1
	1	1	1	0	0	1	0
	1	1	1	1	0	0	1
	•	•	•	•	.	•	•
	•	•	•	•	•	•	•
	•	•	•	•		•	•

256 Bits

.

* An overflow condition is set in the response store registers.

.

Move the Absolute Value of a Field

This instruction will move the absolute value of the contents of field a₁ into field a₂ within the same word for each word of enabled associative memory whose M response store bit is set. The content of the source field is not affected unless overlaid by the destination field. The original content of the destination field is destroyed.

Format	Label	Command	Argument	Comment
	symbol	MVAF	$\frac{a_{1,a_2}}{a_{1,a_2}}$	

• Label Any valid symbol or blank.

Command MVAF

• Argument Two entries are required. The first entry is the source; the second entry is the destination. Both entries represent fields within the same word of associative memory.

• • ^a1^{, a}2

MVAF

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

The X response store, Y response store, FP1, FP3, and FL1 registers are used by this instruction.

Note

MVAF

Example

MVAF (2,3),(10,3)

After Execution

		Array Memory Bit Column							
	М	2	3	4	10	11	12		
ſ	1	0	0	0	0	0	0		
	1	0	0	1	0	0	1		
	1	0	1	0	0	1	0		
	1	0	1	1	0	1	1		
	1	1	0	0	1	0	0*		
s j	1	1	0	1	0	1	1		
	1	1	1	0	0	1	0		
	1	1	1	1	0	0	1		
	•		•	•	.	•			
		•	•	•	.	•			
L	•	•	•	•	•	•	•		

256 Bits

* An overflow condition is set in the response store registers.

Move Field with Increment

This instruction will add one to the value of field a₁ and store the incremented value into field a₂ within the same word for each word of enabled associative memory whose M response store bit is set. The content of the source field is not affected unless overlaid by the destination field. The original content of the destination field is destroyed.

Label	Command	Argument	Comment
symbol	INCF	$\frac{a}{1}$	

• Label

Format

INCF

Any valid symbol or blank.

INCF

• Command

Argument

Two entries are required. The first entry is the source; the second entry is the destination. Both entries represent fields within the same word of associative memory.

• • ^a1, ^a2

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Note

X response store, Y response store, FL1, FP2, FP3, and R0 registers are used by this instruction.

2-134

Example

INCF

(2,3),(10,3)

After Execution

		Array Memory Bit Column						
	M	2	3	4	10	11	12	
ſ	1	0	0	0	0	0	1	
	1	0	0	1	0	1	0	
4.1	1	0	1	0	0	1	1	
	1	0	1	1	1	0	0*	
256	1	1	0	0	1	0	1	
Bits	1	1	0	1	1	1	0	
	1	1	1	0	1	1	1	
	1	1	1	1	0	0	0	
			•					
			•				· .	
. l			•					

* An overflow condition is set in the response store registers.

Move Field with Decrement

Any valid symbol or blank.

DECF

This instruction will subtract one from the value of field a_1 and store the decremented value into field a_2 within the same word for each word of enabled associative memory whose M response store bit is set. The content of the source field is not affected unless overlaid by the destination field. The original content of the destination field is destroyed.

Label	Command	Argument	Comment
sybmol	DECF	$\frac{a}{-1}$ $\frac{1}{-2}$ 2	

• Label

Format

DECF

Command

• Argument

• a1, a2

Two entries are required. The first entry is the source; the second entry is the destination. Both entries represent fields within the same word of associative memory.

There are two ways of denoting a field expression:

1) a_1 or a_2 may be in the form

b± i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1 or a_2 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

X response store, Y response store, FL1, FP2, FP3, and R0 registers are used by this instruction.

Note

DECF

Example

DECF

(2,3),(10,3)

		Array Memory Bit Column						
	М	2	3	4	10	11	12	
ſ	1	0	0	0	1	1	1	
	1	0	0	1	0	0	0	
	1	0	1	0	0	0	1	
	1	0	1	1	0	1	0	
354	1	1	0	0	0	1	1*	
256 J Bits	1	1	0	1	1	0	0	
	1	1	1	0	1	0	1	
	1	1	1	1	1	1	0	
	.		•			•		
	•		•			•		
Į			•			•		

* An overflow condition is set in the response store registers.

ARITHMETICS

This group of associative instructions allows the programmer to perform arithmetic operations between associative memory fields, and between a Common register field with an associative memory field.

This group of instructions will operate only on those associative array memory modules (including response store registers) enabled via the Array Select register. Also, only those words within enabled associative array memory modules whose M response store register bit is set will participate in the execution of the instructions in this group. The most significant bit of all fields is considered to be the sign bit.

Mnemonic	Instructions
ADC	Add Common Register to Field
ADF	Add Field to Field
SBC	Subtract Common Register from Field
SBF	Subtract Field from Field
MPC	Multiply Field by Common Register
MPF	Multiply Field by Field
DVF	Divide Field by Field

Add Common Register to Field

This instruction will add field a₂ of the Common register (addend) to field a₁ of word_i in associative memory (augend) and then store the resultant sum into field a₃ of word_i. Only those words of associative memory whose M response store bit is set will participate in this instruction. The original content of the addend field a₂ is undisturbed. The content of the augend field a₁ will be undisturbed unless the sum field a₃ overlays it.

Format	Label	Command	Argument	Comment
	symbol	ADC	$\frac{a_1 \cdot a_2 \cdot a_3}{a_1 \cdot a_2 \cdot a_3}$	

• Label Any valid symbol or blank.

1)

- Command ADC
- Argument Three entries are required. The first entry represents the augend and is a field in associative memory. The second entry is the addend and is a field in the Common register. These two fields are added together and the sum is stored into the third entry, a field in associative memory.
- • ^a1,^a2,^a3

ADC

There are two ways of denoting a field expression:

a₁, a₂, or a₃ may be in the form b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position of the field.

2) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$ (b, i)±j

> where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

X response store, Y response store, FL1, FP1, FP2, FP3, and R0 registers are used by this instruction.

ADC (0,3),(5,3),(11,3)

After Execution

		Array Memory Bit Column					
	М	0	1	2	11	12	13
ſ	1	0	0	0	0	0	1
	1	0	1	0	0	1	1
	1	1	0	0	1	0	1
	1	1	1	0	1	1	1
256 Words	1	0	1	1	1	0	0*
	1	1	0	1	1	1	0
	1	1	1	1	0	0	0
	•		•	•	. . .	•	•

*An overflow condition will be set in the response store registers.

Common Register



Example

Add Field to Field

This instruction will add field a_1 of word_i to field a_2 of word_i and store the resultant sum into field a_3 of word_i. Only those words of associative memory whose M response store bit is set will participate in this instruction The original content of the source fields a_1 and a_2 will remain undisturbed unless overlaid by the sum field a_3 .

Label	Command	Argument	Comment
symbol	ADF	<u>a</u> 1, a2, a3	

Label

Format

ADF

Any valid symbol or blank.

Command

ADF

• Argument Three entries are required. Each represents a field in associative memory. The first field a₁ represents the augend; the second field a₂ represents the addend; and the third field a₃ represents the sum.

• • ^a1,^a2,^a3

There are two ways of denoting a field expression:

1) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2)

 a_1 , a_2 , or a_3 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, mofifying only the most-significant bit position of the field.

Note

X response store, Y response store, FL1, FP1, FP2, FP3, and R0 registers are used by this instruction.

ADF (0,3),(5,3),(10,3)

After Execution

			Array Memory Bit Column							
	М	0	1	2	5	6	7	10	11	12
ſ	1	0	0	0	0	0	0	0	0	0
	1	0	1	0	0	0	1	0	1	1
	1	1	0	0	0	1	0	1	1	0
256	1	1	1	0	0	1	1	0	0	1
Words	1	0	0	1	1	0	0	1	0	1
	1	0	1	1	1	0	1	0	0	0
	1	1	0	1	1	1	0	0	1	1*
l	1	1	1	1	1	1	1	1	1	0

* An overflow condition will be set in the response store registers.

Example

Subtract Common Register from Field

This instruction will subtract field a_2 of the Common register (subtrahend) from field a_1 of word₁ in associative memory (minuend) and then store the resultant difference into field a_3 of word₁. Only those words of associative memory whose M response store bit is set will participate in this instruction. The origianl content of the Common register field is undisturbed. The content of field a_1 will be undisturbed unless the difference field a_3 overlays it.

Format	Label	Command	Argument	Comment
	symbol	<u>SBC</u>	<u>a</u> 1, <u>a</u> 2, <u>a</u> 3	

Label Any valid symbol or blank.

SBC

- Command
- Argument Three entries are required. The first entry represents a field in associative memory and is the minuend. The second entry represents a field in the Common register and is the subtrahend. The third entry represents a field in associative memory and is the difference of the minuend minus the subtrahend.
- • ^a1,^a2,^a3

SBC

There are two ways of denoting a field expression:

1) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2) a_1, a_2 , or a_3 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

X response store, Y response store, FL1, FP1, FP2, FP3, and R0 registers are used by this instruction.

Note

SBC (5,3), (8,3), (11,3)

After Execution

		Array Memory Bit Column					
	м	5	6	7	11	12	13
	1	0	0	1	1	1	1
	1	0	1	0	0	0	0
	1	0	1	1	0	0	1
	1	1	0	0	0	1	0*
256	1	1	0	1	0	1	1*
Words	1	1	1	0	1	0	0
	1	1	1	1	1	0	1
	1	0	0	0	1	1	0
	•		•	•	.	•	•
			•		.		
	•	•	•	•		•	•

*An overflow condition will be set in the response store registers.

Common Register

0	7	8	9	10	11						3
		0	1								

Example

SBC

Subtract Field from Field

This instruction will subtract field a_2 of word_i from field a_1 of word_i and store the resultant difference into field a_3 of word_i. Only those words of associative memory whose M response store bit is set will participate in this instruction. The original content of the source fields a_1 and a_2 will remain undisturbed unless overlaid by the difference field a_2 .

Label	Command	Argument	Comment
symbol	SBF	<u>a</u> 1,a2,a3	

Label

Format

SBF

Any valid symbol or blank.

SBF

Command

Argument

Three entries are required. Each entry represents a field in associative memory. The first field, a₁, represents the minuend; the second field, a₂, represents the subtrahend; and the third field, a₃, represents the difference.

• • ^a1,^a2,^a3

There are two ways of denoting a field expression:

1) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2)

a₁, a₂, or a₃ may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

X response store, Y response store, FL1, FP1, FP2, FP3, and R0 registers are used by this instruction.

Note

SBF (5,3),(8,3),(11,3)

After Execution

				Array Memory Bit Column							
		М	5	6	7	8	9	10	11	12	13
$\begin{bmatrix} 1 \end{bmatrix}$	1	0	0	0	0	0	0	0	0	0	
		1	0	0	0	0	0	1	1	1	1
	0	0	0	1	1	1	0	0	1		
	1	1	0	0	1	0	1	0	0		
		1	0	1	0	0	0	1	0	0	1
256 Words		1	1	1	1	1	1	1	0	0	0
		1	1	0	1	0	1	0	0	1	1*
		1	1	1	1	1	1	0	0	0	1
.	•			•	.	•	•	.	•		
		•	•	•	•	••	•	•		•	•
			•	•	•	•	•	•	.	•	•

 $*\ensuremath{\mathsf{An}}$ overflow condition will be set in the response store registers.

Example

MPC

Multiply Field by Common Register

This instruction will multiply associative memory field a_1 of word_i (multiplicand) by field a_2 of the Common register (multiplier) and store the product into associative memory field a_3 of word_i. Only those words in associative memory whose M response store bit is set will participate in the multiplication. The original content of the multiplier field a_2 and the multiplicand field is undisturbed i.e., the product field a_3 must not overlay the multiplicand field a_1 .

Format	Label	Command	Argument	Comment
	symbol	<u>MPC</u>	$\frac{a}{2}1, \frac{a}{2}2, \frac{a}{3}3$	

Label Any valid symbol or blank.

MPC

2)

• Argument Three entries are required. The first entry represents a field in associative memory and is the multiplicand. The second entry represents a field in the Common register and is the multiplier. The third entry represents a field in associative memory and is the product. The product field width must equal the sum of the multiplier and multiplicand field widths.

• • ^a1,^a2,^a3

Command

There are two ways of denoting a field expression:

1) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

 a_1 , a_2 , or a_3 may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

X response store, Y response store, FL1, FP1, FP2, FP3, FL2, FPE, and R0 registers are used by this instruction.

Note

MPC

Example

MPC (0,3), (5,3), (8,6)

After Execution

•		Array Memory Bit Column								
	М	0	1	2	8	9	10	11	12	13
	1	0	0	1	0	0	0	0	1	0
	1	0	1	0	0	0	0	1	0	0
	1	0	1	1	0	0	0	1	1	0
256	1	1	0	0	1	1	1	0	0	0
Words	1	1	0	1	1	1	1	0	1	0
	1	1	1	0	1	1	1	1	0	0
	1	1	1	1	1	1	1	1	1	0
	· ·	·	٠	•		•	•	•	•	•
			•	•		•	•	•	•	•

Common Register



Multiply Field by Field

This instruction will multiply field a_1 of word_i by field a_2 of word_i, and store the resultant product into field a_3 of word_i. Only those words of the associative memory whose M response store bit is set will participate in this instruction. The original content of the multiplicand field a_1 must remain intact, i.e., it cannot be overlaid by the product field a_3 . The original content of the multiplier field a_2 may be overlaid by the product field a_3 .

Label	Command	Argument	Comment
symbol	MPF	<u>a1,a2,a</u> 3	

Label

Format

MPF

Any valid symbol or blank.

MPF

- Command
- Argument Three entries are required. The first entry represents a field in associative memory and is the multiplicand. The second entry represents a field in the associative memory and is the multiplier. The third entry represents a field in associative memory and is the product. The product field must equal the width of the sum of the multiplier and multiplicand fields.
- • ^a1,^a2,^a3

There are two ways of denoting a field expression:

1) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$ b±i

> where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2)

a₁, a₂, or a₃ may be in the form

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

X response store, Y response store, FL1, FP1, FP2, FP3, FL2, FPE, and R0 registers are used by this instruction.

Note

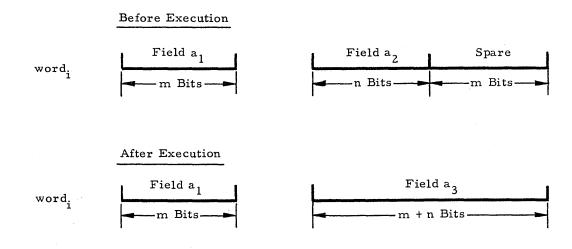
MPF (0,2),(2,3),(5,5)

After Execution

	Array Memory Bit Column									
М	0	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0	0
1	1	1	0	0	1	1	1	1	1	1
1	1	1	0	1	0	1	1	1	1	0
1	0	1	0	1	1	0	0	0	1	1
1	1	0	1	0	0	0	1	0	0	0
1	· 1	1	1	0	1	0	0	0	1	1
1	1	0	1	1	0	0	0	1	0	0
1	0	1	1	1	1	1	1	1	1	1
•	•			•				•		
•	•			•				•		
	1 1	1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1	M 0 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0	M 0 1 2 1 1 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 1 0 1 0 1 0 1 1 0 1 1 1 1 1 1 1 0 1	M 0 1 2 3 1 1 0 0 0 1 1 1 0 0 1 1 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 1 0 1 0 1 1 1 1 0 1 1 0 1 1	M 0 1 2 3 4 1 1 0 0 0 0 1 1 1 0 0 1 1 1 1 0 1 0 1 1 1 0 1 0 1 1 1 0 1 1 1 1 0 1 0 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 0 1 1 0	M 0 1 2 3 4 5 1 1 0 0 0 0 0 1 1 1 0 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 0 1 1 0 1 0 0 0 1 1 1 0 1 1 0 1 1 1 1 0 1 0 1 1 0 1 1 0 0	M 0 1 2 3 4 5 6 1 1 0 0 0 0 0 0 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 1 0 1 1 0 0 1 1 0 1 0 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0	M 0 1 2 3 4 5 6 7 1 1 0 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	M 0 1 2 3 4 5 6 7 8 1 1 0 1

Example 2

Overlaying the multiplier field a₂ must be handled carefully by the programmer. Array memory storage may be condensed (minimum bit positions) in the following manner if there are at the least m''spare'' bits to the right of the multiplier field as shown:



The number of spare bits must be equal to the length of the multiplicand field a₁.

Example 1

Divide Field by Field

This instruction will divide field a_1 of word_i by field a_2 of word_i. Only those words of associative array whose M response store bit is set will participate in the divide instruction. The quotient and the remainder are stored into field a_3 of word_i, with the remainder being right justified and having the same length and sign as the divisor, a_2 . The quotient is stored adjacent to the remainder and must have a length of 2 or more. The contents of the divisor must not be overlaid by the quotient-remainder field a_3 .

Overflow Check Unlike other arithmetic routines, DVF does not check for overflow unless specifically requested in the command field. When requested, the overflow check is made prior to performing the divide. The associative memory words where overflow would occur will have their M response store bit cleared to zero and therefore will not participate in the divide. After the divide, the M is restored, and the possible overflow condition is recorded in the response store registers.

Label	Command	Argument	Comment
symbol	<u>DVF</u> , b	$\frac{a}{1}$ $\frac{a}{2}$ $\frac{a}{3}$	

Label

Format

Command

• • b

scratch bit column position in all words of enabled associative arrays and is used to save the original content of the M response store register. The value of b should be $0 \le b \le 255$.

Any valid symbol or blank.

DVF

• Argument Three entries are required. Each represents a field in associative memory. The first field a₁ represents the dividend; the second field a₂ represents the divisor; and the third field a₃ represents the quotient-remainder. The field length of the quotient-remainder field must be at least two bit positions longer than the divisor field and at least one bit position longer than the dividend.

b may be a constant, a symbol, or a symbol plus or minus an optional

constant modifier. If b was defined as a field via a DF instruction, the most-significant bit position is the value used. This term represents a

•• a₁, a₂, a₃

There are two ways of denoting a field expression:

1) $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$

b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in associative memory. The optional constant modifier, i, modifies only the most-significant bit position.

2-151

• ^a1,^a2,^a3 (cont) 2)

A B C $a_1, a_2, \text{ or } a_3 \text{ may be in the form}$

(b,i)±j

where b may be a constant or a symbol and represents the mostsignificant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant, modifying only the most-significant bit position of the field.

Overlapping

The most efficient field layout is overlapping. This technique can save execution time as well as memory. The least-significant bit of the dividend and quotient-remainder fields should have the same address, and the length of field a_3 must be at least one bit column wider than a_1 . The address of the least-significant bit of a_3 and a_1 must be the same to reduce execution time.

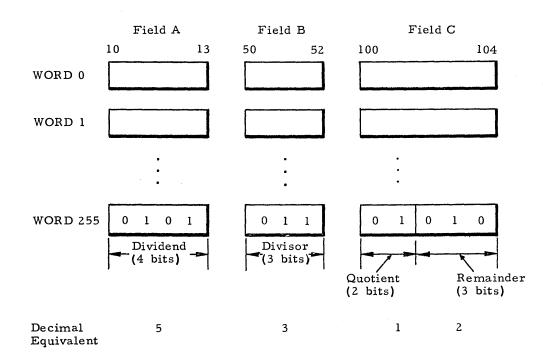
X response store, Y response store, FL1, FP1, FP2, FP3, FL2, FPE, and BL registers are used by this instruction.

Example 1

Note

Dividend and divisor not affected:

DF	10,4
DF	50,3
DF	100,5
•	
•	
•	
DVF	А, В, С



2-152

Example 2 Most efficient memory layout: OVERFLOW SCRATCH BIT со EQU 4 DIVISOR \mathbf{DF} 5,5 DIVISOR DIVIDEND \mathbf{DF} 100,10 DIVIDEND QUOTREM 99,11 QUOTIENT, REMAINDER \mathbf{DF} • . • . • DVF, CO DIVIDEND, DIVISOR, QUOTREM

In this example overflow will occur in a word if the quotient requires more than 6 bits.

Example 3

Most efficient memory layout with no overflow condition

DIVISOR	$_{ m DF}$	5,5	DIVISOR
DIVIDEND	\mathbf{DF}	100,10	DIVIDEND
QUOTREM	DF	95,15	QUOTIENT, REMAINDER
	•	•	
	•	•	
	•	•	
	DVF	DIVIDEN	1D, DIVISOR, QUOTREM

CONTROL AND TEST

This group of instructions allows the programmer to control and test the AP control.

Mnemonic	Instructions
INT	Interrupt Control and Test
ILOCK	Interlock Control and Test
WAIT	Deactivate the AP

Interrupt Control and Test

Label

This instruction will generate an interrupt and/or interrogate the current state of an interrupt according to the value of the argument field expression $a_2\pm k_2$. The interrupt number is denoted by the expression $a_1\pm k_1$.

Comment

be optionally

Argument

			1 <u> </u>			
	symbol	<u>INT,a</u> 1 ^{±k} 1	$\frac{a}{2}2^{\pm k}2$			
Label	Any valid s	symbol or blank				
Command	INT					
• ^a 1 ^{±k} 1	+	either a constant y plus or minus t	t or a symbol whos the constant k _l .	se value may		
• Associative Processor Interrupts	Valid Entr Interrupt N	Vecto	or Address 11k Core			
	X'1' X'8001'		01'			
	X'1' X'8002'					
	•					

Command

• • Sequential Processor Interrupts

Interrupt Number	Vector Address in Sequential Processor
O'300'	O'300'
O'304'	O'304'
O'310'	O'310'
	•
•	•
•	•
O'334'	O'334'

X'800F'

Argument

• • a₂±k₂

 a_2 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_2 .

Legal Values for $a_2 \pm k_2$:

X'F'

Valid Entries:

- 0 Unconditionally disable the interrupt.
- 1 Skip the next instruction if interrupt is enabled, and then unconditionally disable the interrupt
- 2 Skip the next instruction if interrupt is disabled, and then unconditionally disable the interrupt
- 3 Unconditionally skip the next instruction and then disable the interrupt

2-155

INT

Format

Legal Values for $a_2 \pm k_2$: (cont)

7

- 5 Skip the next instruction if the interrupt is enabled.
- 6 Skip the next instruction if the interrupt is disabled.
 - Unconditionally skip the next instruction.
- 8 Unconditional complement of current state.
- 9 Skip the next instruction if interrupt is enabled, and then unconditionally complement current state.
- 10 Skip the next instruction if interrupt is disabled, and then unconditionally complement current state.
- 11 Uncontitionally skip the next instruction, and then unconditionally complement current state.
- 12 Unconditionally enable the interrupt.
- 13 Skip the next instruction if the interrupt is enabled, and then unconditionally enable the interrupt.
- 14 Skip the next instruction if the interrupt is disabled, and then unconditionally enable the interrupt.
- 15 Unconditionally skip the next instruction and then enable the interrupt.

ILOCK

Interlock Control and Test

This instruction will set or reset the specified interlock number $a_1 \pm k_1$ and/or interrogate the current state of this interlock number according to the value of the expression $a_2 \pm k_2$.

Format

Label	Command	Argument	Comment
symbol	ILOCK,al ^{±k} 1	$\frac{a}{2}2^{\pm k}2$	

Label Any valid symbol or blank

Command ILOCK

• • $a_1 \pm k_1$

 a_1 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_1 . The value of $a_1 \pm k_1$ must be in the range $0 \le a_1 \pm k_1 \le 63$. These interlocks have no predetermined meaning. The programmer can assign any meaning to any interlock.

Argument

• • ^a2^{±k}2

 a_2 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_2 . The value of $a_2 \pm k_2$ determines the action taken on the specified interlock number $a_1 \pm k_1$.

Legal Values for $a_2 \pm k_2$:

- 0 Unconditionally reset the interlock.
- 1 Skip the next instruction if set, and then unconditionally reset the interlock number.
- 2 Skip the next instruction if reset, and then unconditionally reset the interlock number.
- 3 Unconditionally skip the next instruction and then reset the interlock number.
- 4 No operation.
- 5 Skip if the interlock number is set.
- 6 Skip if the interlock number is reset.
- 7 Unconditionally skip the next instruction.
- 8 Unconditionally complement current state.
- 9 Skip the next instruction if set, and then unconditionally complement current state.
- 10 Skip the next instruction if reset, and then unconditionally complement current state.
- 11 Unconditionally skip the next instruction and then unconditionally complement current state.
- 12 Unconditionally set the interlock number.
- 13 Skip the next instruction if the interlock is set, and then unconditionally set the interlock number.
- 14 Skip the next instruction if the interlock is reset, and then unconditionally set the interlock number.
- 15 Unconditionally skip the next instruction, and then set the interlock

2-157

WAIT	Deactivate the AP				
	This instrue	This instruction will cause the associative processor to go inactive.			
Format	Label	Command	Argument	Comment	
	symbol	WAIT			
• Label	Any valid sy	ymbol or blank.			
• Command	WAIT				
• Argument	No entry is required.				

PAGER INSTRUCTIONS

These instructions allow the programmer to utilize the page memories.

Mnemonic	Instructions
STRTSG	Start Segment
ENDSG	End Segment
MVSG	Move a Page Segment
MVSGI	Move a Page Segment Immediate
PAGER	Pager Control

STRTSG

Start Segment

This instruction marks the beginning of a page segment by reinitializing the Execution Location Counter as specified in the Command Field.

Format	Label	Command	Argument	Comment	
	symbol	<u>STRTSG,a</u> ±k			
• Label	Any valid symbol or blank. This will be the name of the following segment.				
• Command	STRTSG				
• • a±k	'a' may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k. Moreover,'a' may be one of the following special symbols: PAGE0, PAGE1, PAGE2. The value of the				

expression a±k initializes the Execution Location Counter and represents where succeeding assembled APPLE instructions are to be loaded and then executed.

• Argument

No entries required.

ENDSG	End Segment This instruction marks the end of a page segment.					
Format	Label Command Argument Commen					
	symbol	ENDSG				
• Label	Any valid symbol or blank.					
• Command	ENDSG					
• Argument	No entries required.					
Note	Nested STRTSG-ENDSG pairs are illegal.					

Move a Page Segment

This instruction will command the Pager to move a segment of instructions referenced by the Memory Address $a_2\pm k_2$ if the Pager is not busy with a previous move. If the Pager is busy, AP Control will wait until the previous move is completed before initiating this move.

LabelCommandArgumentCommentsymbol $\underline{MVSG, a_1 \pm k_1}$ $\underline{a_2 \pm k_2}$

Any valid symbol or blank.

- Command
 MVSG
- • a₁±k₁

Label

 a_1 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_1 . Moreover, a_1 may be one of the following special symbols: PAGE0, PAGE1, PAGE2. This term corresponds to the same term in the STRTSG mnemonic. Its value is used to tell the Pager where to begin storing the program segment.

Argument One term is required.

 $\bullet \bullet a_2 \pm k_2$

 a_2 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_2 . The value of this term should reference the address of a STRTSG mnemonic in bulk core.

Format

MVSGI

Move a Page Segment Immediately

This instruction will command the Pager to move a segment of instructions referenced by $a_2\pm k_2$ to the Page Memory address $a_1\pm k_1$ immediately. If the AP Control encounters the MVSGI instruction while the Pager is busy with a previous move, it will interrupt the Pager and initiate the new move immediately. The remainder of the previous move is forgotten. If the Pager is not busy when the AP control encounters an MVSGI instruction, it acts like an MVSG instruction.

Label		Command	Argument	Comment
symbol	L	<u>MVSGI,a</u> l ^{±k} l	$\frac{a}{2}2^{\pm k}2$	

Label

Format

Command

• • ^a1^{±k}1

 a_1 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_1 . Moreover, a_1 may be one of the following special symbols: PAGE0, PAGE1, PAGE2. This term corresponds to the same term in the STRTSG mnemonic. Its value is used to tell the Pager where to begin storing the program segment.

Argument

One term is required.

Any valid symbol or blank.

MVSGI

 $\bullet \bullet a_2 \pm k_2$

 a_2 may be either a constant or a symbol whose value may be optionally modified by plus or minus the constant k_2 . The value of this term should reference the address of a STRTSG mnemonic in bulk core.

PAGER

Pager Control

This instruction will command and/or interrogate the state of the Pager. The Pager can be considered to be on (busy) or off (not busy).

Format	Label	Command	Argument	Comment			
	symbol	PAGER	<u>a</u> ±k				
• Label	Any valid	symbol or blank.					
• Command	PAGER						
• Argument	One entry	is required.					
•• a±k		e either a constant by plus or minus th		e value may be optionally			
· · · · · · · · · · · · · · · · · · ·	Legal Val	ues of a±k:					
	0	Unconditionally tu	ırn Pager off.				
	1	Skip the next instr turn Pager off.	ruction if Pager is	on, and then unconditionally			
	2	Skip the next instruction if Pager is off, and then unconditionally turn Pager off.					
	3	Unconditionally skip the next instruction and then turn Pager off.					
1	4	No operation.					
	5	Skip the next instruction if Pager is on.					
	6	Skip the next instruction if Pager is off.					
	7	Unconditionally sh	cip the next instru	ction.			
	8	Unconditionally complement current state.					
	9	Skip the next instr complement curre		on, and then unconditionally			
	10	Skip the next instr complement curre		off, and then unconditionally			
	11	Unconditionally sl complement curre		ction, and then unconditionally			
	12	Unconditionally tu	ırn Pager on.				
	13	Skip the next instr turn Pager on.	ruction if Pager is	on, and then unconditionally			
	14	Skip the next inst turn Pager on.	ruction if Pager is	off, then unconditionally			
	15	Unconditionally sl	kip the next instru	ction, and then turn Pager on.			

CHAPTER 3

SUPERVISOR CALLS

INTRODUCTION

STARAN Program Supervisor (SPS) provides services to supplement the APPLE language such as managing input/output, handling errors, and controlling STARAN processors. This chapter describes all services available to APPLE programs executed by the associative processor.

SPS consists of two program modules which are resident in the sequential control memory at execution time. Module zero (SPS0) manages the sequential controller and its associated peripherals. Module one (SPS1) manages the remaining STARAN processors. SPS0 and SPS1 together manage the entire stand-alone configuration; that is, the sequential controller and its I/O devices, the associative processor, the Pager, the Parallel I/O unit* and additional Custom I/O, as implemented for a specific installation.

The main purpose of SPS is to make input/output operations possible. A programmer can use the supervisor through the APPLE SVC (Supervisor Call) mnemonic by specifying the particular service desired, a buffer, and some I/O device.

SLOT NUMBERS

The convention is not to refer to devices directly, but to the slots assigned them (a slot is similar to a logical unit, see figure 3-1). An I/O request is made on a slot which, in turn, refers to a device. (Slots 0, 1, and 2 are anchored to the keyboard, teleprinter, and high-speed reader. While it is necessary to refer to these slots, any attempt to reassign them will cause an error message to be printed at execution time.) Slot assignments are recorded in the Device Assignment Table (DAT) (figure 3-1). More than one slot may be assigned to a device, but only one device can be attached to a slot. Additional devices will be added to satisfy requirements of specific customers (for example - disk units, magnetic tape units, remote terminals, etc).

STARAN registers, associative memory, and control memory are referenced in SPS calls as though they were devices. They are given negative device codes which may be used to attach them to slots (see figure 3-1). SPS provides for data transfer between these associative processor elements, or between them and any peripheral device.

* Parallel I/O unit is an optional STARAN feature.

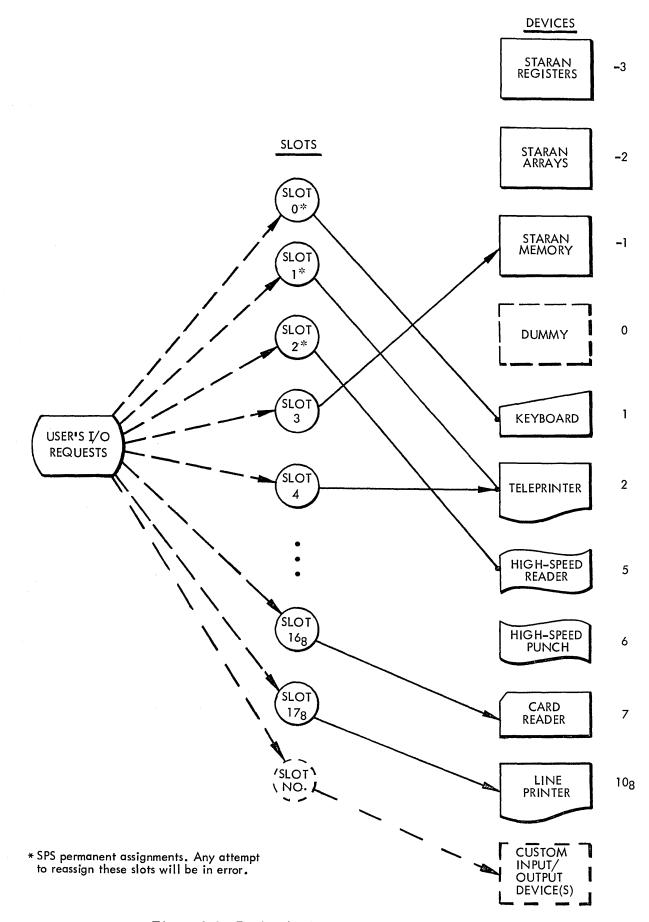


Figure 3-1. Device Assignment Table (DAT)

INSTRUCTION DESCRIPTION

This section of the manual is concerned with the description of two mnemonics and their possible variations. Their basic format is:

BUFFER PSEUDO-OP FORMAT

Label Command Argument Comment BUFFER symbol <u>a1, a2, a3, a4, a5, a6, a7</u> and

SUPERVISOR CALL	Label	Command	Argument	Comment
FORMAT	symbol	SVC	<u>a</u> 1, ^a 2, ^a 3	

Note

The terms in the argument field for these mnemonics will be described in more detail later.

BUFFER

All I/O functions included in this basic manual, except Parallel I/O, require a buffer area (usually either in the HSDB or in BULK core memory) to contain the data as they are input or output. The purpose of the BUFFER mnemonic is to create for the SPS I/O routine required buffer-area header information describing in more detail the exact nature of the intended I/O process. For example, data input from the highspeed paper tape unit may be in the form of ASCII characters (formatted or unformatted) or pure binary values (formatted or unformatted). These cases will be described in more detail later. In other words there is more than one way to input or output data on a given device, and the purpose of the BUFFER pseudo-op is to fully describe the desired method.

SVC

There are currently twelve different variations (SPS services) of the SVC mnemonic. The services will be expanded for future requirements as necessary for Customized Input/Output features. Some of the variations require a corresponding BUFFER mnemonic counterpart to fully describe the nature of the I/O operation to the system. The twelve SVC functions are:

Source Statement Format

SPS SERVICES OR CALLS

Function	Label	Com- mand	Argument	Comment
Attach	symbol	<u>svc</u>	l, slot-number, device-code-address	
Read	symbol	SVC	9, slot-number, buffer-address	
Write	symbol	<u>svc</u>	10, slot-number, buffer-address	
Reset	symbol	<u>svc</u>	2	
Free	symbol	SVC	5, slot-number	
Exit	symbol	SVC	7	
Restart	symbol	SVC	8	
Timer	symbol	SVC	13, timer-number, interrupt number, time-value	
Int	symbol	SVC	14, interrupt-number	
I Setup	symbol	SVC	15, interrupt-number, status, done-address	
Pager Control	symbol	<u>svc</u>	18, command, start-address	
PI/O Control	symbol	SVC	19, command, start-address	

Each of the above SVC instructions will be described separately in the following pages. Note the distinguishing feature of each SVC instruction is the first term in the argument field. All argument field terms shown are required. Each term in the argument field may be in the form

a±k

where 'a' may be either a constant or a symbol optionally modified by plus or minus the constant k.

ATTACH

Attach an SPS Slot Number to an I/O Device

This function allows the programmer to assign SPS slot numbers to different I/O devices at execution time (see figure 3-1). Note that slot numbers 0, 1, and 2 always refer to the keyboard, teleprinter, and high-speed reader, respectively. These slot numbers may not be reassigned to different devices.

Format	Label	Command	Argument	Comment
	symbol	SVC	l, slot-number, device-code-address	
• Label	Any valid s	symbol or blank		
• Command	SVC			
• Argument	The argum	ent field consis	ts of three entries.	
••1	The value	of the first entr	y (must be equal to one) denotes an attac	h
	Tune tron.			
• • Slot Number	The value	of the second en	try (must be a value between 0 and 17_8)	
Number	denotes on	e of the SPS slo	t numbers (see figure 3-1).	
• • Device-	The third o	entry represent	s an address (must be in either HSDB or	in
Code- Address	BULK core	e memory)whicl	n contains a device code value.	

• • Device Codes	Value (Octal)	Device
Codes	-3	STARAN Registers
	-2	STARAN Associative Memory
	-1	STARAN Control Memory
	0	Dummy
	1	Keyboard (KBD)
	2	Teleprinter (TTY)
	3	Low-Speed Reader (LSR)
	4	Low-Speed Punch (LSP)
	5	High-Speed Reader (HSR)
	6	High-Speed Punch (HSP)
	7	Card Reader (CDR)
	10	Line Printer (LPT)

STARAN Special Device Codes For ATTACH Function Note that STARAN storage is given device codes. To allow access to parts of STARAN not directly addressable, the memory elements are formally treated as devices. SVC calls utilizing STARAN device codes require corresponding specially formatted BUFFER pseudo-op mnemonics.

• STARAN Control Memory When STARAN control memory is referenced by an SVC instruction, it should be for the purpose of a memory-to-memory transfer of a block of instructions or data within STARAN Control Memory, which consists of the Page, HSDB, and Bulk core memories.

For an SVC instruction referencing a slot assigned to device -1 (STARAN Control Memory) the form of the corresponding BUFFER mnemonic is:

Label	Command	Argument	Comment
symbol	BUFFER	address, 0, byte-count	

• • • Label

Format

Any valid symbol or blank.

• • • Command BUFFER

••• Argument The argument field consists of three terms. The value of the first term represents any STARAN control memory address and is the destination address. The value of the second term must be zero. The value of the third term represents the byte-count of the number of bytes of data following the BUFFER pseudo-op involved in the memory-to-memory transfer.

The above BUFFER pseudo-op will generate the following two 32-bit words of memory:

• • BUFFER Format For Device -1

0		15 16		31
	Address			
0		15 Iú	2324	31
	Byte Count	Sta	atus	
0				31
		Data		
0		0 0 0		31
		Data		

As shown the upper half of the first word will contain the value of the first item of the argument field. The upper half of the second word will contain the value of the number of bytes of data involved in the memory-to-memory transfer. The bytes of data must be contained in the words of memory immediately following the BUFFER pseudo-op mnemonic as shown.

This value represents the actual number of bytes of data to be moved to or from the buffer area that follows.

• • Status

Byte

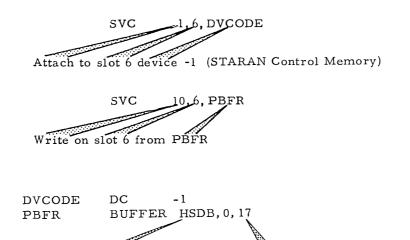
• • • Byte

Count

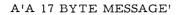
Bits of the Status Byte shown in bits 16-23 of the second word set by SPS upon completion of the memory-to-memory transfer are bits 16 and 17:

16	17	18	19	20	21	22	23
Done Bit	Error Bit						

The Done bit will be set to a value of one by SPS upon completion of the operation. The ERROR bit will be set to a value of one by SPS if an error occurred during the operation. If an error occurred before the memory-to-memory transfer was completed, the Byte Count value in the upper half of the second word will contain the actual number of bytes of data transferred by SPS before the operation was terminated. Otherwise the Byte Count value will remain intact. • • Example



Transfer the following 17 bytes



Destination address in HSDB

Execution of the above two SVC instructions will result in the memoryto-memory transfer of the above 17 bytes of ASCII characters to the HSDB.

 STARAN Associative Memory When STARAN Associative Memory is referenced by an SVC instruction, it should be for the purpose of a memory-to-memory transfer between associative memory and a buffer located either in the HSDB or Bulk core memories. For an SVC instruction referencing a slot assigned to device -2 (STARAN Associative Memory) the form of the corresponding BUFFER mnemonic is:

• • Format	Label	Command	Argument	Comment
	symbol	BUFFER	maxsize, address-mode, byte count, a1, a2, b1, b2	

• • • Label

Any valid symbol or blank.

• • • Command BUFFER

• • • Argument

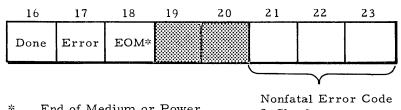
The argument field consists of seven terms. Before describing these terms consider the three words of object-code generated by this particular BUFFER pseudo-op mnemonic (the words of data are not generated by this mnemonic). • BUFFER Format For Device -2

0		15	16	31
		n size of in bytes		
0		15	16 23	3 2 4 3 1
	Byte	Count	Status	Address Mode
0	7	8 15	16 23	24 31
	b2	bl	az	^a 1
		co-or	dinates	
	Data	Data	Data	Data
			0 0 0	
	Data	Data	Data	Data

• • • Maximum Size The upper half of the first word will contain the value of the first term of the argument field. This value represents the maximum size (in bytes) of the data words portion of the buffer and also the maximum allowable number of bytes of data that may be moved to or from the buffer.

• • Byte Count The upper half of the second word will contain the value of the third term of the argument field. This value represents the actual number of bytes of data to be moved to or from the buffer area that follows. If for some reason an error condition occurs before a buffer transfer is completed, this value will be modified to reflect the actual number of data transfers.

• • Status Byte Bits 16-23 of the second word contain the Status Byte. The bit value in this byte are set and maintained by SPS.



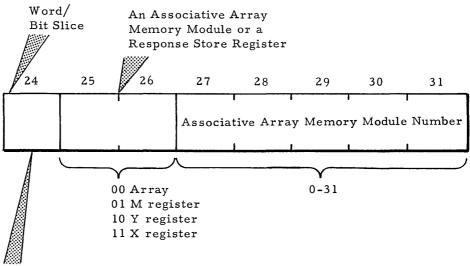
* End of Medium or Power Off or Out of Tape, etc.

** Formatted Binary Data*** End of Tape Code

Nonfatal Error Code 2=Checksum error 3=Long-line error 4=Improper Mode (FBIN**) 5=EOT*** will occur if an ASCII EOT code is read outside of a FBIN block of data. • • • Done This bit will be set by SPS upon completion of the buffer transfer Bit regardless of whether or not an error occurred.

••• Error This bit will be set by SPS if an error is detected during the buffer transfer operation. Examine bits 18 and 21-23 for the exact nature of the error.

• • • Address Mode Byte Bits 24-31 of the second word comprise the Address Mode Byte and is loaded with the value of the second term in the argument field. The interpretation of the bit and sub-field values of the byte by SPS is:



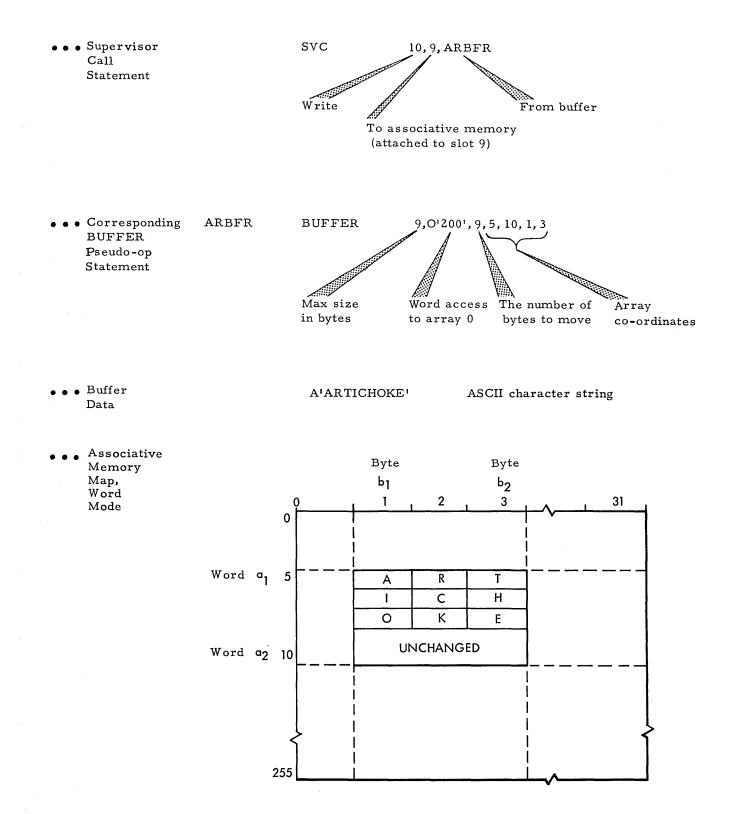
Bit 24=1 means access by word. Bit 24=0 means access by bit column.

• • Array Co-ordinates The third word of the buffer header information is loaded with the associative memory coordinate values as shown from the fourth through seventh terms in the argument field of the BUFFER pseudo-op mnemonic. a_1 and a_2 define the starting and ending "line" which may be bit column numbers or word row numbers, depending on bit 24 of the Address Mode; i.e., $0 \le a_1 \le a_2 \le 255$. b_1 and b_2 are the starting and ending byte numbers for the data; i.e., $0 \le b_1 \le b_2 \le 31$. If a response store register instead of associative memory is specified, a maximum of 32 bytes of data can be transferred.

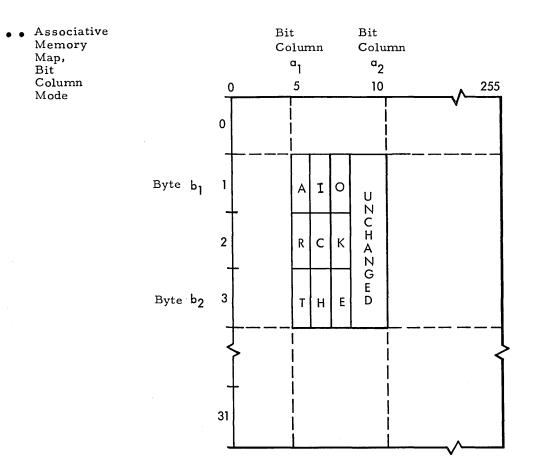
Example

Writing Into Associative Memory

If slot 9 has been attached to associative memory (device code -2), the following statements will load array 0 as shown.



 a_1 and a_2 define the starting and ending "line", $0 \le a_1 \le a_2 \le 255$. b_1 and b_2 are the starting and ending byte numbers for the data, $0 \le b_1 \le b_2 \le 31$.



• STARAN Registers When STARAN registers are referenced by an SVC instruction, it should be for the purpose of obtaining the value of those STARAN registers accessable only by the sequential controller. For an SVC READ instruction referencing a slot assigned to device -3 (STARAN Registers) the form of the corresponding BUFFER mnemonic is:

• • Format	Label	Command	Argument	Comment
	symbol	BUFFER	register-code	

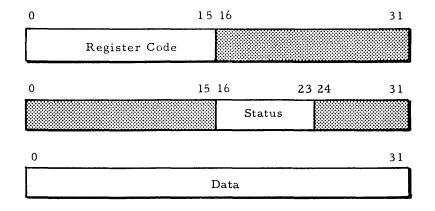
• • • Label

Any valid symbol or blank.

• • • Command BUFFER

••• Argument The argument field consists of one term. Before describing this term consider the two words of object code generated by this particular BUFFER pseudo-op mnemonic (the word of data is not generated by this mnemonic).

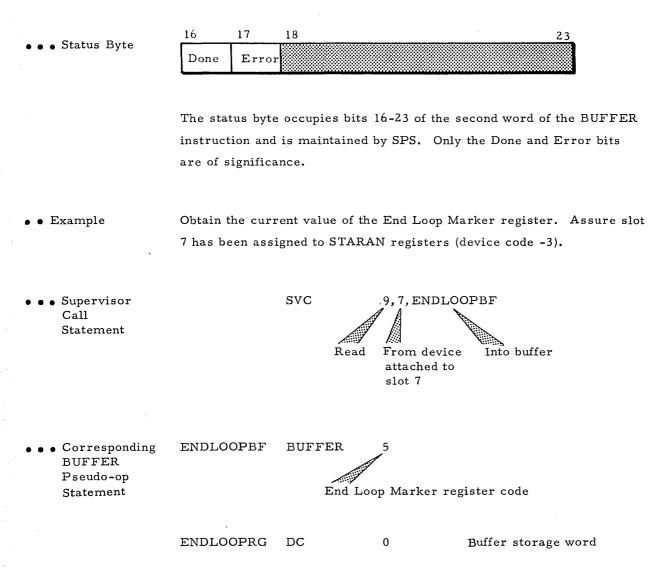
• • Buffer Format For Device -3



• • • Register Code

The upper half of the first word will be loaded with the value of the argument field term. This value represents the code assigned to a particular register.

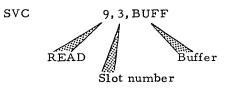
Register	Register Code	Length (bytes
Instruction	0	4
Pager Put	1	2
Pager Get	2	2
Pager Count	3	2
Start Loop Marker	4	2
End Loop Marker	5	2
<u>PI/O Ur</u>	it* Main Frame	
Instruction	6	4
Start Loop Marker	7	2
End Loop Marker	8	2
Buffer Register	9	4
Buffer Control Word No. 1	10	2
Buffer Control Word No. 2	11	2
Buffer Control Word No. 3	12	2
Block Length Counter	13	2
Data Pointer	14	2
Program Counter	15	2
Field Length Counter	16	1
Field Pointer No. 1	17	1
Field Pointer No.2	18	1
Field Pointer No.3	19	1
STARAN Address Mode	20	2
PI/O Address Mode	21	2
Performance Timer	22	2
Performance Counter	23	2



The value of the End Loop Marker register may be obtained from the contents of the one word buffer ENDLOOPRG.

READ This SVC function permits the transfer of data from a valid input device to a buffer area in Bulk core or the High-Speed Data Buffer. The input device is referenced by a slot number. Detail on memory to memory transfers for special AP devices is described in the ATTACH section.

Format	Label	Command	Argument	Comment
	symbol	SVC	9, slot-number, buffer-address	
• Label	Any valid sy	mbol or blank.		
• Command	SVC			
• Argument	Three entrie	s are required.		
• • 9	This entry n	nay be a constant	or a symbol whose value is 9.	
				1
• • Slot- Number	-	•	or a symbol whose value is the s device (see figure 3-1).	5100
• • Buffer-	This entry n	nay be a constant	or a symbol representing the	
Address			er which is set-up in a BUFFER	
			O/WRITE BUFFER Pseudo-Op	
	instruction	is described in a	later section.)	
5				-
Example	tape reader.	number 5 nas be	en attached to the high speed pape	
	tape reader.			



This example will cause data to be read from the high speed paper tape reader and stored into the buffer area defined by BUFF (buffer area in Bulk core or High Speed Data Buffer). BUFF must be defined in a BUFFER Pseudo-Op instruction.

13

WRITE This SVC function permits the transfer of data from a buffer area in Bulk core or the High-Speed Data Buffer to a valid output device. The device is referenced by a slot number. Detail on memory-tomemory transfers for special AP devices is described in the ATTACH section.

Format	Label	Command	Argument	Comment
	symbol	SVC	10, slot-number, buffer-address	
• Label	Any valid	symbol or bla	nk.	
• Command	SVC			
• Argument	Three end	ries are requi	red.	
• • 10	This opta	r mar ha a aa	nstant or a symbol whose value is	10
• • 10	THIS ENU	y may be a con	istant of a symbol whose value is	10.
• • Slot-Number	This entr	y may be a con	nstant or a symbol whose value is	the slot-
		ssigned to an o		
		·		
• • Buffer - Address			nstant or a symbol representing th	
nuur ess			which is set-up in a BUFFER Ps RITE BUFFER Pseudo-Op instruc	_
		l in a later sec	_	
Example	Assume s	lot number O'	17' has been attached to the line p	rinter.
		SVC	10, O'17', BUFF	

This example will cause the contents of BUFF (buffer area in Bulk core or High-Speed Data Buffer) to be printed on the line printer. BUFF must be defined in a BUFFER Pseudo-Op instruction.

WRITE Slot number Buffer

READ/The BUFFER Pseudo-Op instruction sets up a properly formattedWRITEbuffer area in control memory for the READ and WRITE supervisorPSEUDO-OPcall functions. Detail on special AP device buffers is described in the
ATTACH section.

Format	Label Command	Argument	Comment
	symbol <u>BUFFER</u>	max-size address register code, mode, byte-count, a1, a2, b1, b2	
• Label	Any valid symbol or	blank.	
• Command	BUFFER		
• Argument	except the AP regist	es are required for all READ/WRITE operations er transfers (device code -3) which requires only remaining four entries are required for Associa ers.	
• • Max-size	This entry may be a maximum size of the	constant or symbol whose value represents the buffer in bytes.	
• • Address		constant or symbol representing a Bulk core or fer address in a memory to memory transfer.	
• • Register Code		constant or symbol representing an AP register ers involving AP registers.	
• • Mode	mode of data transfe	constant or symbol whose value represents the r. The mode entry will always be zero for a transfer since mode is not applicable.	
	2 Unfo 1 For	<u>Mode</u> prmatted Binary (UBIN) prmatted ASCII (UASCII) matted Binary (FBIN) matted ASCII (FASCII)	

3-17

- • Unformatted Binary
 Eight bit bytes are transferred as specified by the buffer byte count. This mode is suitable for paper tape readers and punches.
 Possible errors are Error bit set and End of Medium (EOM)
- • Unformatted ASCII
 Seven bits per byte are transferred as specified by the buffer byte count. This mode is suitable for keyboard, teleprinter, and line printer. Possible errors are Error bit set, End of Medium, Checksum, and Long line.

• • • Formatted Binary This format is used primarily for paper tape I/O (used by APPLE.) For output SPS blocks the data as follows:

${}^{201_8}_{000}$	The header for the APPLE assembler output
$\begin{pmatrix} xxx \\ xxx \end{pmatrix}$	The byte count of the block, equal to the number of data bytes plus 4.
ууу ууу :	Data bytes
zzz zzz	The checksum, the two's complement of the sum of all the preceding bytes in the block (Sum plus checksum equals zero)

When a block with a header 201000 (i.e. APPLE block) is encountered, the number of bytes transferred is the block byte count -4. The data are followed by a checksum. On output, SPS creates the header, the block byte-count, and the checksum.

• • • Formatted ASCII

Seven bits per byte are transferred until a terminating character is encountered. Terminating characters are, a line feed (012), a form feed (014), or a carriage return (015). These characters will end transmission of data. Possible errors are Error bit set, End of medium, and Long line.

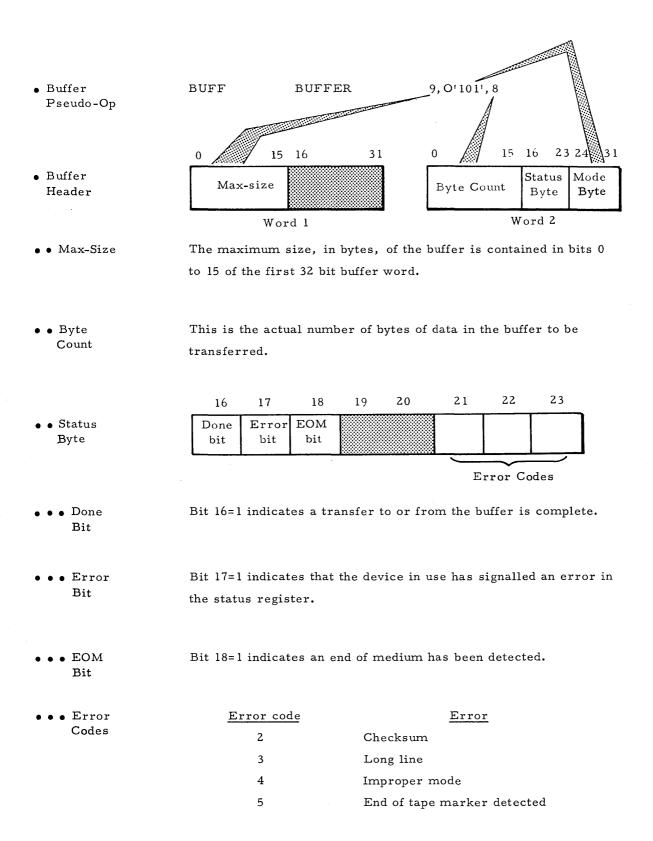
• • Byte Count Actual number of data bytes in the block to be transferred.

• • a₁, a₂, b₁, b₂

These entries are used only when an associative memory data transfer is performed. a_1 and a_2 define the start and end of the array words or bit slices (ranges: $0 \le a_1 \le a_2 \le 255$). b_1 and b_2 define the start and end byte numbers for the data (range: $0 \le b_1 \le b_2 \le 31$). If a response store register is specified, up to 32 bytes will be moved in or out of the buffer.

Example

The following example illustrates the buffer format for transfer of data to or from an I/O device.



• • Mode Byte 24 25 26 27 28 29 30 31 Byte Binary/ Echo Count Update Format Bit ASCII 11 unformatted binary 01 formatted binary 10 unformatted ASCII 00 formatted ASCII • • Echo Normally data entered through the keyboard are printed on the Bit teleprinter (referred to as echo). If bit 7=1 in the mode byte, the echo is inhibited. This bit applies only to keyboard input.

- Byte Count Update
 SPS revises the byte count in the buffer header after each I/O call using that buffer. This is not always desirable. For example, an error during attempted printing of a message might zero the byte count. Subsequent calls using that buffer would output zero bytes. Bit 6=1 will inhibit the updating and should be used only for a read operation.
- • Format Bit 30 indicates formatted or unformatted data mode. If bit 30 is zero, the mode is formatted, if bit 30 is equal to 1, the mode is unformatted.

••• Binary/ ASCII RESTARTThis statement will completely reinitialize the STARAN and returnPROGRAMcontrol to a sequential control program (refer to STARAN SystemsProgrammer's Reference Manual)

Format	Label Command		Argument	Comment
	symbol	SVC	8	

• Label Any valid symbol or blank.

• Command SVC

• Argument The argument must be an expression whose value is 8.

RESET To reinitialize peripheral devices, issue RESET. This has the effect of RESTART, but control will return to the following instruction, not to a restart address. It is not usually necessary to RESET since conditions that call for reinitializing usually call for restarting, too.

Format	Label	Command	Argument	Comm ent
	symbol	SVC	2	

• Label Any valid symbol or blank.

SVC

Command

• Argument Any expression whose value is 2 denotes the reset function.

FREE	The device attached to the designated slot will be made ready to				
DEVICE For	start a new I/O process. If the device is busy, its current I/O				
NEW TASK	process will be halted and cannot be resumed.				
	This service is most useful for getting an urgent - and usually				
	terminal - message in or out.				

Format	Label	Command	Argument	Comment
	symbol	SVC	5, slot-number	

• Label Any valid symbol or blank.

- Command SVC
- Argument Two entries are required.

KILL

••5 The first entry may be any expression whose value is 5; this denotes the FREE function of the supervisor call.

•• Slot-Number The second entry may be any expression whose value is between 0 and 15 (see figure 3-1).

Example

SVC 5,0

The statement named KILL will cause the device attached to slot 0 (the teleprinter) to halt any current output and make it ready for a WRITE request.

EXIT A STARAN program may return control to the supervisor with TO SUPERVISOR this command.

Format	Label	Command	Argument	Comment
	symbol	SVC	.7	

• Label Any valid symbol or blank.

• Command SVC

• Argument

Any expression whose value is 7 denotes the EXIT function of the supervisor call.

TIMER START

The TIMER statement allows clocking of an interval beginning with the execution of the statement. At the end of the specified interval, STARAN interrupt will be triggered.

Format	Label	Command	Argument	Comment
	symbol	SVC	13, timer-number, interrupt- number, time-value	
• Label	Any valid	symbol or blar	ık.	
• Command	SVC			
• Argument	All entries	are required		
• • 13	The first e	entry may be a	ny expression whose value is 13;	
	this denote	es the TIMER f	function of the supervisor call.	
• • Timer- Number			e an expression whose value is 0, our different "clocks" which may	
• • Interrupt- Number	or 15, It		n expression whose value is 1,2, TARAN interrupt which will be tr	
•• Time- Value		entry will be units of 1/300	evaluated and taken as an unsigned) sec.	d 16-bit
Example	EXCELS	SVC	13, 2, 5, 300	
		above statemer 5 in l second (nt is executed, timer 2 will trigger 300/300ths).	r AP

INTERRUPT SIGNAL This command is used to cause the sequential processor to execute a program. SPS simulates sixteen interrupt vectors in the sequential processor. (This is not a STARAN hardware interrupt into the sequential processor, signalled by external-function codes. It is a software facility to make possible user linkages between STARAN and the sequential processor.) When the software interrupt is triggered, the sequential processor will execute a program at the address specified in the ISETUP call (discussed on following page).

Format	Label	Command	Argument	Comment
	symbol	SVC	14, interrupt-number	
• Label	Any valid s	ymbol or blan	k.	
• Command	SVC			
• Argument	The argum	ent field consi	sts of two entries.	
• • 14	The first e	ntry must be a	n expression whose valu	e is 14;
	this denote	s the INT func	tion of the supervisor ca	11.
• • Interrupt- Number		14,15. It spe	expression whose value m ecifies a software interru	
Example		SVC	14,0	
	This will to	igger the inte	rrupt setup as shown in t	he next section
	(I SETUP e	example).		

I SETUP This SVC function creates a software interrupt vector for the sequential controller.

Format	Label	Command	Argument	Comment
	symbol	SVC	<u>15, interrupt-number, status,</u> interrupt-vector-address	

• Label

Any valid symbol or blank.

Argument

All entries are required.

• • 15

The first entry must be an expression whose value is 15; it denotes the I SETUP function of the supervisor call.

 Interrupt-Number
 The second entry is an expression whose value must be 0, 1, 2, ..., 14, or 15. It specifies a software interrupt vector maintained by SPS. (See Staran System Programmer's Reference Manual.)

• • Status

The third argument specifies the status to be assumed when the interrupt is signalled. It will be evaluated and taken to be a number from zero to seven.

• Interrupt-Vector The fourth argument is an expression whose value is a sequential control address in Bulk core memory.

Example

EXETER SVC 15,0,7,HANDLR

The above line of coding will attach sequential control software interrupt 0 to a routine called HANDLR. Priority 7 will be assumed.

3-27

PAGERThis SVC function is used to control certain Pager operations.CONTROL

Format	Label	Command	Argumen	it	Comment
	symbol	SVC	18, operation, sta	rt-address	
	·		•		
• Label	Any valid	symbol or blar	ık.		
• Command	SVC				
• Argument		es are require nires all three	d in all operations entries.	except the S	tart Pager ,
• • 18	The first e	entry may be a	constant or a syn	nbol whose va	lue is 18.
• • Operation	The secon	d entry may be	e a constant or a s	ymbol with th	e following values:
	Valu		peration		
	0	Start Pa	ger at start-addre	ess	
	1	Stop Pag	ger		
	2	Pause P	Pager		
	3	Continue	e Pager from paus	e	
• • Start-	The start	-address is u	sed only in the Sta	rt Pager oper	ation (i.e. when
Address	the opera	tion value is :	zero).		
Example	TAG	DC ·	0		
		svc	18, TAG, X'00'	Start Pager	at address 0
		svc	18,1	Stop Pager	

PI/O This SVC function is used to control certain parallel I/O operations. CONTROL*

Format	Label	Command	Argument	Comment
	symbol	SVC	19, operation, start-address	
• Label	Any valid a	symbol or blar	ık.	
• Command	SVC			
• Argument		es are require nires all three	d in all operations except the entries.	Start PI/O ,
• • 19	The first e	entry may be a	constant or a symbol whose v	value is 19.
• • Operation	The second values:	d entry may be	a constant or a symbol with	the following
	Valu	e Ope	ration	
	0	Start PI	O at start-address	
	1	Stop PI/	0	
	2	Pause P	I/O	

• • Start-Address The start-address is used only in the Start PI/O operation (i.e. when the operation value is zero).

Continue PI/O from pause

3

* Parallel I/O is an optional STARAN feature. Other Custom I/O features may be handled similarily.

APPENDIX A

SUMMARY OF APPLE MNEMONICS

AND

INSTRUCTION FORMATS

ASSEMBLER
DIRECTIVES

	Mnemonic (Command)	Argument	Instruction	Pago
	<u>(command)</u>	<u>In guillent</u>	<u>Instruction</u>	Page
ASSEMBLER	START *		Start APPLE source	2-10
DIRECTIVES	END	a±k	End APPLE source	2-10
	ORG	<u>a</u> ±k	Initialize location counter	2-11
	Label EQU	<u>a</u> ±k	Equate (Define Symbol)	2-12
	Label DF	$\underline{a}_1 \pm k_1 \underline{a}_2 \pm k_2$	Define Field	2-12
	DS, a±k	<u>_</u>	Define Storage	2-13
	TOF	· · · · · · · · · · · · · · · · · · ·	Top of Form	2-13
•	EVEN		Make location counter EVEN	2-14
	$\underline{DC}, a_1 \pm k_1$	$\frac{a}{2}2^{\pm k}2$	Define Constant	2-14
	$\underline{\operatorname{GEN}, k_1, \ldots k_n}$	$\underline{a_1^{\pm j_1, \dots a_n^{\pm j_n}}}$	Generate Machine Instruction	2-15
	NOP		No Operation	2-15
	$\underline{Axc_1c_2\cdots c_{i-1}}$	c.x	Character String Generator	2-16
	$\frac{1}{\underline{\text{Exc}}_{1}c_{2}\cdots c_{i-1}}$	<u>1-</u>	Character String Generator	2.16
	1~2····i-1	1-		2-16
BRANCH	B	<u>a</u> (r)±k, cd	Unconditional Branch	2-18
INSTRUCTIONS	$\underline{BZ, r_1}$	$\underline{a(r_2)}$ ±k, cd	Branch if Zero	2-19
	BNZ,r ₁	$\underline{a(r_2)}$ ±k, cd	Branch if Not Zero	2-21
	BBS	<u>a(r)±k, cd</u>	Branch if Bit Set	2-23
	BBZ	a(r)±k, cd	Branch if Bit Zero	2-25
	BRS		Branch if Response	2-27
	BNR	<u></u> <u>a(r)</u> ±k, cd	Branch if No Response	2-28
	BOV	$\underline{a(r)}\pm k, cd$	Branch if Overflow	2-29
	BNOV	<u>a(r)±k, cd</u>	Branch if No Overflow	2-30
	BAL, r	$a(r_2) \pm k, cd$	Branch and Link	2-31
	<u></u> I <u>RPT</u> , a±k	. 2. 7	Repeat	2-33
	$\underline{\text{LOOP}}, a_1 \pm k_1$	$\underline{a}_2(\mathbf{r}) \pm k_2$	Loop	2-34
		-22		
REGISTER	<u>LRR</u> , k _s	r _o ,r,	Load Register from Register	2-37
INSTRUCTIONS	<u>LI</u> , k _s	$\frac{\mathbf{r}_{2}\mathbf{r}_{1}}{\mathbf{r}, \mathbf{a} \pm \mathbf{k}}$	Load Register with Immediate Data	2-39
	LR,k _s	$r_{2,a}(r_1)$ ±k, cd	Load Register from Control Memory	2-41
	<u>SR</u> , h	$\frac{\underline{r}_{2,\underline{a}}(r_1) \pm k, cd}{\underline{r}_{2,\underline{a}}(r_1) \pm k, cd}$	Store Register in Control Memory	2-47
	INCR		Increment the Register	2-49
	DECR	$\frac{r_1,\ldots,r_n}{r_n}$	Decrement the Register	2-49
		$\frac{\underline{r}_1, \ldots, \underline{r}_n}{\underline{a}(r) \pm k, cd}$	Load Program Status Word	2-51
•	LPSW, k _s	$\underline{a(r)} \pm k, cd$	Swap Program Status Word	
	SPSW	<u>a(r)=k</u> , cu	Swap 110gram Status Word	2-53
	* Required entries ar	e underlined through	hout	
		and and and the state of the state of any state of the st	n et negerinen en nie gegeneen en je 'n die begehein in die en die stere die die neer die die bestere en die be	

ASSOCIATIVE INSTRUCTIONS

1

LOAD RESPONSE STORE REGISTERS AND COMMON REGISTER

Mnemonic (Command)	Argument	Instruction	Page
<u>L</u>	$\frac{rs}{2} \left\{ \frac{rs}{a \pm k} \right\}$	Load Response Store Register	2 - 56
LN	$\frac{\mathbf{rs}_{2'}}{\sum_{i=1}^{n}} \left\{ \frac{\mathbf{rs}_{1}}{\sum_{i=1}^{n}} \right\}$	Load Complemented	2 - 58
LOR	$\frac{\mathbf{rs}_{2}}{\underline{r}_{2}} \left\{ \frac{\mathbf{rs}_{1}}{\underline{a} \pm \mathbf{k}} \right\}$	Load Logical OR	2-60
LORN	$\frac{rs}{2} \cdot \left\{ \frac{\frac{rs}{a+k}}{\frac{r}{r}} \right\}$	Load Logical OR Complemented	2-62
LAND	$\frac{\mathbf{rs}_{2^{\perp}}}{\sum_{i=1}^{n}} \left\{ \frac{\mathbf{rs}_{1}}{\sum_{i=1}^{n}} \right\}$	Load Logical AND	2-64
LANDN	$\frac{\mathbf{rs}_{2}}{\frac{\mathbf{rs}_{1}}{\frac{\mathbf{a}\pm\mathbf{k}}{\mathbf{r}}}}$	Load Logical AND Complemented	2-66
LXOR	$\frac{\mathbf{rs}_{2}}{\mathbf{r}} \left\{ \frac{\mathbf{rs}_{1}}{\underline{a} \pm \mathbf{k}} \right\}$	Load Logical Exclusive OR	2-68
LXORN	$\frac{\mathbf{rs}_{2^{\prime}}}{\left\{\frac{\mathbf{rs}_{1}}{\underline{a}\pm\mathbf{k}}\right\}}$	Load Logical Exclusive OR Complemented	2-70
LC	<u>a</u>	Load Common Register from an Associative Memory Word	2 - 72
<u>LCM</u>	<u>a</u> 1,a2	Load Common Register Field from an Associative Memory Word	2-74
<u>SET</u>	rs	Set Response Store Register	2-76
CLR	<u>rs</u>	Clear Response Store Register	2_77
ROT	<u>rs,a</u> 1 ^{±k} 1,a2 ^{±k} 2	Rotate Response Store Register	2-78

	[]			_
	Mnemonic (Command)	Argument	Instruction	Page
STORE RESPONSE	<u>s</u>	$\underline{rs}, \left\{\frac{\underline{a}\pm k}{\underline{r}}\right\}$	Store Response Store Into Associative Memory	2-80
STORE REGISTERS AND	<u>SM</u>	$\underline{rs}, \left\{\frac{a^{\pm k}}{r}\right\}$	Store Response Store Masked Into Associative Memory	2-82
COMMON REGISTER	<u>SN</u>	$rs_{t}\left\{\frac{a^{\pm k}}{r}\right\}$	Store Complement Into Associative Memory	2 - 84
	<u>SNM</u>	$\underline{Y}, \left\{\frac{\underline{a}\pm k}{\underline{r}}\right\}$	Store Complement Masked Into Associative Memory	2-86
	SOR	$\underline{rs}, \left\{\frac{a\pm k}{r}\right\}$	Store Logical Inclusive OR Into Associative Memory	2-88
	<u>SORM</u>	$\underline{\underline{Y}}, \left\{ \underline{\underline{a}}^{\pm k} \right\}$	Store Logical Inclusive OR, Masked Into Associative Memory	2-90
	<u>SORN</u>	$\frac{\mathbf{rs}_{\star}\left\{\frac{\mathbf{a}\pm\mathbf{k}}{\mathbf{r}}\right\}}{\mathbf{r}}$	Store Logical Inclusive OR, Complemented Into Associative Memory	2-92
	<u>SORNM</u>	$\underline{\underline{Y}}, \left\{\frac{\underline{a} \pm k}{\underline{r}}\right\}$	Store Logical Inclusive OR, Complented, Masked Into Associative Memory	2-94
	SAND	$\frac{rs}{r}\left\{\frac{a\pm k}{r}\right\}$	Store Logical AND Into Associative Memory	2-96
	<u>SANDM</u>	$\underline{\mathbf{Y}}, \left\{\frac{\underline{\mathbf{a}} \pm \mathbf{k}}{\underline{\mathbf{r}}}\right\}$	Store Logical AND Masked Into Associative Memory	2-98
	<u>SANDN</u>	$\underline{rs,} \left\{ \frac{a^{\pm k}}{r} \right\}$	Store Logical AND Complemented Into Associative Memory	2-100
	SANDNM	$\underline{Y}, \left\{\frac{\underline{a}\pm k}{\underline{r}}\right\}$	Store Logical AND, Complemented, Masked Into Associative Memory	2-102
	<u>SC</u>	$\frac{a}{1\cdot a_2}$	Store Common Register Into Associative Memory	2-104
	<u>SCW</u>	$\frac{a}{1}$	Store Common Register Into Associative Word	2-106
SEARCHES	FIND		Find the First Bit Set in Y Response Store	2-109
	STEP		Step to First Y Set and Clear It	2-109
	RESVFST		Step to First Y Set and Clear All Others	2-110
	EQC	$\frac{a}{1}$	Equal to Common Register Field	2-111
	EQF	$\frac{a}{a}$ $\frac{a}{a}$ $\frac{a}{a}$ $\frac{a}{a}$ $\frac{a}{a}$	Equal Fields	2-112
	NEC	$\underline{a}_1 \underline{a}_2$	Not Equal to Common Register Field	2-113
	NEF	$\underline{a_1, a_2}$	Not Equal Fields	2-114
	GTC	$\underline{a}_1, \underline{a}_2$	Greater Than Common Register Field	2-115
	GTF	<u>a</u> 1 <u>,a</u> 2	Greater Than Fields	2-116
	GEC	$\frac{a_1, a_2}{a_1, a_2}$	Greater than or Equal to Common Register Field	2-117
	GEF	<u>a</u> 1 <u>,a</u> 2	Greater than or Equal Fields	2-118
	LTC	<u>a</u> 1 <u>,a</u> 2	Less Than Common Register Field	2-119
	LTF	<u>a</u> 1, a ₂	Less Than Fields	2-120
	LEC	$\frac{a_1,a_2}{a_1}$	Less Than or Equal Common Register Field	2-121
	LEF	<u>a</u> 1 <u>,a</u> 2	Less Than or Equal Fields	2-122
	MAXF	<u>a</u>	Maximum Fields	2-123
	MINF	<u>a</u>	Minimum Fields	2-124
1				

ł

MOVES

ARITHMETICS

Mnemonic (Command)	Argument	Instruction	Page
MVF	$\underline{a}_1, \underline{a}_2$	Move Field	2-126
MVCF	$\underline{a}_1, \underline{a}_2$	Move the One's Complement of a Field	2 - 128
MVNF	$\frac{a_1, a_2}{a_1, a_2}$	Move the Negative of a Field	2-130
MVAF	<u>a₁, a₂</u>	Move the Absolute Value of a Field	2-132
INCF	$\frac{a_1,a_2}{a_1,a_2}$	Move Field with Increment	2 - 134
DECF	$\frac{a_1, a_2}{2}$	Move Field with Decrement	2-136
ADC	$\underline{a}_1, \underline{a}_2, \underline{a}_3$	Add Common Register to Field	2-139
ADF	$\frac{a_{1}, a_{2}, a_{3}}{a_{1}, a_{2}, a_{3}}$	Add Field to Field	2-14
SBC	$\frac{a_{1}, a_{2}, a_{3}}{a_{1}, a_{2}, a_{3}}$	Subtract Common Register from Field	2-143
SBF	$\frac{a_{1}, a_{2}, a_{3}}{a_{1}, a_{2}, a_{3}}$	Subtract Field from Field	2-14
MPC	$\frac{a_{1}, a_{2}, a_{3}}{a_{1}, a_{2}, a_{3}}$	Multiply Field by Common Register	2-14
MPF	$\frac{a_{1}, a_{2}, a_{3}}{a_{1}, a_{2}, a_{3}}$	Multiply Field by Field	2-149
DVF	$\frac{a_{1}, a_{2}, a_{3}}{a_{1}, a_{2}, a_{3}}$	Divide Field by Field	2-15
$\underline{INT}_{a_1}^{\pm k_1}$	$\underline{a}_2, \pm k_2$	Interrupt Control and Test	2 - 15
ILOCK, a1 ±k1	$a_{2}, \pm k_{2}$	Interlock Control and Test	2-15
WAIT		Deactivate the AP	2-158
<u>STRTSG,a</u> ±k		Start Segment	2-160
ENDSG		End Segment	2-160
$MVSG_{a_1}^{\pm k_1}$	$\underline{a}_2^{\pm k}_2$	Move a Page Segment	2-16
$MVSGI, a_1 \pm k_1$	$\underline{\underline{a}_2^{\pm k}}_2$	Move a Page Segment Immediate	2-162
PAGER	a±k	Pager Control	2-163

CONTROL AND TEST

PAGER INSTRUCTIONS

	Mnemonic (Command)	Arguments	Instruction	Page
APPLE I/O Statements	BUFFER	Maxsize, mode, byte-count	Buffer-header Pseudo-op	3-17
	SVC	l,slot-number,device-code-address	Attach Device to Slot	3-5
	SVC	2	Reset Peripherals	- 3-22
	SVC	5, slot-number	Free a Device for I/O	3-23
	SVC	7	Exit From Program	3-24
	SVC	8	Restart Program	3-21
	SVC	9, slot-number, buffer-address	Read Into Buffer	3-15
	SVC	10, slot-number, buffer-address	Write From Buffer	3-16
	SVC	13, timer-number, interrupt-number, time-value	Start a Timer	3-25
	SVC	14, interrupt-number	Int-Signal Interrupt	3-26
	SVC	15, interrupt-number, status, done-address	I Setup-Interrupt	3-27
	SVC	18, operation, start-address	Pager Control	3-28
	SVC	19, operation, start-address	PI/O Unit Control	3-29

A-5

APPENDIX B

ERROR CODES

ERROR CODES When APPLE scans source statements to produce the object code, it checks for improper use of the defined grammar. Up to two error codes can be printed in the left hand margin for each statement in error. Error code meanings are listed below.

Error Code	Meaning
А	\underline{A} ddressing error. An address within the instruction is incorrect.
В	\underline{B} oundary error. An address that should be even (odd) is odd (even).
D	Doubly-defined symbol referenced. Reference is made to a symbol that is defined more than once.
F	Illegal <u>f</u> orward reference of a symbol.
I	<u>Il</u> legal character detected.
К	Array address out of range.
L	Lengths of array fields incompatible.
Μ	<u>Multiple</u> definition of a label. A label is encountered that is identical to a previously encountered label.
Р	A <u>page</u> segment boundary syntax error.
Q	\underline{Q} uestionable syntax. There are missing arguments or the instruction scan was not completed.
R	\underline{R} egister-type error. An invalid use of or reference to a register has been made.
S	<u>Symbol</u> table overflow. When the quantity of user-defined symbols exceeds the allocated space available in the user's symbol table, the assembler outputs the current source line with the S error code, then returns to the initial dialogue.
Т	<u>Truncation error</u> . A number being loaded into a register or storage location is larger than the length the register or location allows.
U	<u>Undefined symbol.</u> An undefined symbol is encountered during the evaluation of an expression. Relative to the expression, the undefined symbol is assigned a value of zero.
v	<u>V</u> alue out of range.
W	Warning. Nonstandard usage or procedure. Processing continues.

B-1

APPENDIX C

TERMS AND SYMBOLS

Associative Processor

Associative Memory	An associative array memory module consists of two basic components: array storage and response store. Each array contains 65,536 bits, organized as a square 256 words by 256 bits of solid state storage. Array input and output may be either 32 bits or 256 bits in parallel. Input data may be stored into the array through a mask contained in the response store.
Array Selector	The Array Select register establishes those associative array memory modules that are to be active for an associative operation. The Array Select register is 32 bits wide. Each bit position controls one array, i.e., bit 0 corresponds to array 0, bit 1 corresponds to array 1, etc. A value of one in an Array Select register bit position will enable the corresponding array number.
AS	Array Select register (32 bits)
ASH	Array Select register, High half (bits 0-15)
ASL	Array Select register, Low half (bits 16-31)
Associative Array	See Associative Memory
BL	Block length counter (16 bits)
Block Length counter	The block length counter is a 16-bit decrementing counter. The block length counter may be sued to control the length of a data block transfer.
Branch and Link registers	A group of registers that occupies dedicated memory locations in the HSDB (addresses 600_{16} to 607_{16}). They are used as linkages to subroutines.
Bulk Core	The bulk core memory is a section of AP control memory used to store instructions or data. In the standard STARAN S configuration it contains 16,384 words (32 bits each). Bulk core addresses range from 8000_{16} to BFFF ₁₆ .

Common register (32 bits)

Control Digit

С

 $^{\rm cd}$

C-1

CDR	Card Reader
СН	Common register, High half (bits 0-15)
CL	Common register, Low half (bits 16-31)
Common Register	The Common register is an AP register that contains 32 bits numbered 0 to 31. Bit 0 is the left-most (most-significant) bit. Bit 31 is the right- most (least-significant) bit. The Common register may contain the argument for a search operation performed upon the associative memory, the input data stored into an associative memory, or the input data received from an associative memory in a load operation. Data from an associative memory is loaded into the Common register through a mask generated by the mask generator.
Control Digit	A control digit permits post-incrementing or post-decrementing of the DP register and/or post-decrementing the BL register. It is implemented in instructions with a Control Memory address, using the DP register as a base register.
Control Memory Address	The Control memory address is a symbolic or absolute address in Bulk Core, Page Memory, or the High Speed Data buffer. Valid address ranges are 000_{16} to $7 {\rm FF}_{16}$ and 8000_{16} to ${\rm BFF}_{16}$.
Data Pointer	The data pointer is a 16-bit register in AP control that may contain the control memory address for block transfers. The data pointer can be stepped with each transfer within a data block.
DMA	Direct Memory Access
DP	Data Pointer (16 bits)
DP0	Data Pointer, byte 0 (bits 0-7)
DP1	Data Pointer, byte 1 (bits 8-15)
EOM	End of Medium
EOT	End of Tape
Execution Location Counter	The Execution Location counter indicates the address of the instruction when it is executed. This will differ from the Load Location Counter only when program segments are moved to a Page Memory for execution.

C-2

FASCII

Formatted ASCII Code

FBIN

Formatted Binary Code

Field Expression There are two ways of denoting a field expression:

l) b±i

where b must be a symbol, and i is an optional constant modifier. b should have been previously defined in a DF instruction. b represents the most-significant bit position and the number of contiguous bits occupied by a field in either the Common register or associative memory. The optional constant modifier, i, modifies only the mostsignificant bit position.

2) (b,i)±j

where b may be a constant or a symbol and represents the most significant bit position of a field. If b was defined as a field via a previous DF instruction, the most-significant bit position is the value used. i must be a constant and represents the number of contiguous bits occupied by the field. j is an optional constant modifying only the mostsignificant bit position of the field.

Field Lengh Counter Field length counters are 8-bit AP control registers used to contain the length of data fields. They may be decremented to allow stepping through the bits of a data field. When the counter's contents equal zero, an indication is sent to the AP control for test purposes. There are two field length counters: FL1 and FL2.

Field Pointers A field pointer is an 8-bit AP control register that generally contains an array bit column or word address. Field pointers may be incremented or decremented to facilitate stepping through data fields. There are four field pointers: FP1, FP2, FP3, and FPE.

FL1	Field Length counter 1 (8 bits)
FL2	Field Length counter 2 (8 bits)
FPE	Field Pointer E (8 bits)
FP1	Field Pointer 1 (8 bits)
FP2	Field Pointer 2 (8 bits)
FP3	Field Pointer 3 (8 bits)

High-Speed Data Buffer The High-Speed Data Buffer is a section of AP Control memory consisting of fast solid state elements. In the standard configuration of STARAN S it contains 512 words with addresses from 600_{16} to $7FF_{16}$. Since the HSDB can be accessed faster than bulk core, it is a convenient place to store data and instructions that require quick access.

Interlocks

The EXF logic contains 64 stored bits called interlocks. These bits have no predetermined meaning. Software may assign a meaning to an interlock and use it for any purpose. Sixteen interlocks (hex addresses 00 through 0F) can be controlled and sensed manually be panel switches and lights to facilitate communication with an operator. The other 48 interlocks (hex addresses 10 through 3F) can only be sensed and controlled by function codes.

Interrupt MASK The program status word in the program control logic contains the interrupt mask for the 15 AP control interrupts. All interrupts with numbers greater than the mask are accepted. The interrupt mask is contained in bits 28 through 31 of the program status word.

Link Pointer

Load

Location

Counter

The link pointer is registers FP1 and FP2 concatenated together. FP1 contains the address of the selected associative array memory module and FP2 contains the array word or bit column address. The link pointer is commonly used to store the address of the first responder of a search operation.

The load location keeps track of the addresses associated with instructions when they are loaded.

LSB

Least Significant Bit (bit 31 of 32-bit word); right-most bit; low order bit.

М

M-Response Store Register; MASK (256 bits)

M Response Store register

MASK

The M response store register (MASK) is a 256-bit register contained in the response store element of each associative array memory module. Its special use is to select associative memory words participating in an associative operation.

M-Response Store Register

Masked Store	Data being stored into associative memory may be stored through a mask which is contained in the M response store register. This is a masked store. Data will be stored only into words that have the corresponding M register bit set. Other words are unchanged.
MDA	Multi-Dimensional Access memory; associative memory
MSB	Most Significant Bit (bit 0 of word); left-most bit; high-order bit.
Page Memory	Three page memories of 512 words each are included in the AP control memory of the standard STARAN S configuration. They are fast memories that should be used for program segments that require frequent usage and/or fast execution. The page memory address ranges are: Page 0 - 000_{16} to $1FF_{16}$; Page 1 - 200_{16} to $3FF_{16}$; Page 2 - 400_{16} to $5FF_{16}$.
Page0	High-speed solid state memory; 512 32-bit words
Pagel	High-speed solid state memory; 512 32-bit words
Page2	High-speed solid state memory; 512 32-bit words
PC	Program Counter (16 bits)

PARALLEL INPUT/ OUTPUT (OPTIONAL FEATURE) Each associative array in STARAN can have up to 256 inputs and 256 outputs into the custom I/O cabinet. The basic width of the parallel input/output (PI/O) is 256 n where n is equal to the number of associative arrays in the system (n can have a maximum value of 32). The custom I/O cabinet is capable of buffering and reformatting the data received from any peripheral device to match the width necessary to communicate with the STARAN associative array.

Program Counter The program counter occupies bits 0-15 of the program status word in AP control. The program counter contains the address of the current instruction being executed. It is normally incremented sequentially through control memory. Its normal sequence may be altered by branch or loop instruction.

Program Status Word	The Program Status Word (PSW) consists of the program counter (PC) (bits 0-15), which contains the address of the current AP control instruction being executed, and the Interrupt Mask (IMASK) (bits 28-31), which contains the current interrupt status.
PSW	Program Status Word
Resolve	The resolver logic in AP control finds the associative array memory module address and word address of the first responder. The array address is loaded into FP1 and the word address into FP2 (see link pointer). This permits subsequent operations to only affect the first responder.
Responder	A responder is a response store element in an enabled associative array memory module whose Y register bit is set. Generally, responders indicate words satisfying some search criteria. The Y register can be tested for a response or a no-response condition.
Response	See responder
RS	Response Store
R0	Branch and Link Register (memory location 600 ₁₆)
R 1	Branch and Link Register (memory location 601 ₁₆)
R2	Branch and Link Register (memory location 601 ₁₆)
R3	Branch and Link Register (memory location 603 ₁₆)
R4	Branch and Link Register (memory location 604 ₁₆)
R5	Branch and Link Register (memory location 605 ₁₆)
R6	Branch and Link Register (memory location 606 ₁₆)
R7	Branch and Link Register (memory location 607 ₁₆)
SPS	STARAN Program Supervisor
SVS	Supervisor call
UBIN	Unformatted Binary
UASCII	Unformatted ASCII

X Response Store Register (256 bits)

The X response store register is a 256-bit register contained in the response store element of each associative array memory module. It may be used as temporary storage of data loaded from the array or stored into the array. It can be combined logically with data from the input network and/or the Y register. It is useful as temporary storage in parallel arithmetic operations or searches.

Y Response Store register (256 bits)

Y Response Store Register

Y

The Y response store register is a 256-bit register contained in the response store element of each associative array memory module. It may be used as temporary storage of data loaded from the array or stored into the array. It can be combined logically with data from the input network. It is useful as temporary storage in parallel arithmetic operations and searches. It is also used as the responder in a resolve operation.

х

Response

Register

Store

х

C-7

APPENDIX D

HEXADECIMAL /DECIMAL TABLE

HEXADECIMAL-DECIMAL NUMBER CONVERSION TABLE

GENERAL

The table provides for direct conversion of hexadecimal and decimal numbers in these ranges:

Hexadecimal 000 to FFF Decimal 0000 to 4095

HEXADECIMAL-DECIMAL NUMBER CONVERSION In the table, the decimal value appears at the intersection of the row representing the most significant hexadecimal digits $(16^2 \text{ and } 16^1)$ and the column representing the least significant hexadecimal digit (16^0) .

Example

21,16	=	³¹⁰⁵ 10	
IEX	0	> 1	2
C0	3072	3073	3074
C1	3088	3089	3090
C2	3104	3105	3106
C3	3120	3121	3122
	HEX C0 C1 C2	HEX 0 C0 3072 C1 3088 C2 3104	HEX 0 1 CO 3072 3073 C1 3088 3089 C2 3104 (3105)

For numbers outside the range of the table, add the following values to the table figures:

Hexadecimal	Decimal	Hexadecimal	Decimal
1000	4,096	C000	49,152
2000	8,192	D000	53,248
3000	12,288	E000	57,344
4000	16,384	F000	61,440
5000	20,480	10000	65,536
6000	24,576	20000	131,072
7000	28,672	30000	196,608
8000	32,768	40000	262,144
.9000	36,864	50000	327,680
A000	40,960	60000	393, 216
B000	45,056	70000	458,752

1C21 ₁₆ =	7201 ₁₀
Hexadecimal	Decimal
C21 +1000	3105 4096
1C21	7201

Example

D-1

		0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F
•	00 01 02 03 04 05 06 07 08 07 08 09 0A 0B 0C 0D 0E 0F	0000 0016 0032 0048 0064 0096 0112 0128 0144 0160 0176 0192 0208 0224 0240	0001 0017 0033 0049 0065 0081 0097 0113 0129 0145 0161 0173 0193 0209 0225 0241	0002 0018 0034 0050 0066 0082 0098 0114 0130 0146 0162 0178 0194 0210 0226 0242	0003 0019 0035 0051 0067 0083 0099 0115 0131 0147 0163 0179 0195 0211 0227 0243	0004 0020 0036 0052 0068 0106 0116 0132 0148 0148 0148 0196 0212 0228 0244	0005 0021 0037 0053 0069 0085 0101 0117 0133 0149 0165 0181 0197 0213 0229 0245	0006 0022 0038 0054 0070 0086 0102 0118 0134 0150 0162 0198 0214 0230 0246	0007 0023 0039 0055 0071 0087 0103 0119 0135 0151 0167 0183 0199 0215 0231 0247	0008 0024 0040 0056 0072 0088 0104 0120 0136 0152 0168 0184 0200 0216 0232 0248	0009 0025 0041 0057 0073 0105 0121 0137 0153 0169 0185 0201 0217 0233 0249	0610 0026 0042 0058 0074 0090 0106 0122 0138 0154 0170 0186 0202 0218 0234 0250	0011 0027 0043 0059 0075 0107 0123 0139 0155 0171 0187 0203 0219 0235 0251	0012 0028 0044 0060 0072 0108 0124 0140 0152 0172 0188 0204 0236 0252	0013 0029 0045 0061 0077 019 0125 0141 0157 0141 0157 0189 0205 0221 0237 0253	0014 0030 0046 0062 0078 0094 0110 0126 0142 0158 0174 0190 0206 0222 0238 0254	0015 0031 0047 0063 0079 0195 0111 0127 0143 0159 0191 0223 0223 0239 0255
		0	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F
	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F	0256 0272 0288 0304 0320 0352 0368 0384 0400 0416 0432 0448 0464 0480 0496	0257 0273 0289 0305 0321 0337 0353 0369 0385 0401 0417 0433 0449 0465 0481 0497	0258 0274 0290 0306 0322 0354 0370 0386 0402 0418 0430 04450 0466 0482 0498	0259 0275 0291 0307 0323 0355 0371 0387 0403 0419 0435 0451 0467 0483 0499	0260 0276 0292 0308 0324 0356 0372 0388 0404 0420 0435 0452 0468 0484 0500	0261 0277 0293 0309 0325 0341 0357 0373 0389 0405 0421 0437 0453 0469 0485 0501	0262 0278 0294 0310 0326 0358 0374 0390 0406 0422 0438 0454 0470 0486 0502	0263 0279 0295 0311 0327 0343 0375 0391 0423 0439 0423 0439 0455 0471 0487 0503	0264 0280 0296 0312 0328 0340 0376 0392 0408 0424 0440 0456 0472 0488 0504	0265 0281 0297 0313 0329 0345 0361 0377 0393 0425 0441 0457 04473 0489 0505	0266 0282 0298 0314 0330 0362 0378 0394 0426 0442 0458 0442 0458 0474 0490 0506	0267 0283 0299 0315 0347 0363 0379 0363 0379 0363 0411 0427 0443 0459 0459 0475 0491 0507	0268 0284 0300 0316 0332 0364 0380 0364 0380 0412 0428 0444 0460 0472 0508	0269 0285 0301 0317 0333 0349 0365 0381 0397 0413 0429 0445 0445 0445 0461 0477 0493 0509	0270 0286 0302 0318 0350 0366 0382 0366 0382 0398 0414 0430 0446 0430 0446 0446 0494 0510	C271 0287 0303 0319 0351 0351 0367 C383 0415 C431 0447 0463 0447 0463 0479 0495 0495
		. 0	1	2	3	4	5	6	7	8	9	A	В	Ĺ	D	E	F
	20 21 22 23 24 25 26 27 28 29 2A 28 20 22 22 22 22 22 22 22 22 22 22 22 22	0512 0528 0544 0560 0576 0592 0608 0624 0656 0672 0688 0704 0720 0736 0752	0513 0529 0545 0561 0577 0593 0625 0625 0625 0657 0673 0673 0705 0721 0737 0753	0514 0530 0546 0562 0578 0610 0626 0642 0642 0642 0642 0674 0690 0706 0726 0738 0754	0515 0531 0547 0563 0579 0611 0627 0643 0675 0691 0707 0723 0739 0755	0516 0532 0548 0564 0596 0612 0628 0628 0660 0676 0692 0708 0724 0724 0740 0756	0517 0533 0549 0565 0581 0597 0613 0629 0645 0661 0677 0693 0709 0725 0741 0757	0518 0534 0550 0566 0582 0614 0630 06642 0678 0694 0726 0726 0742 0758	0519 0535 0551 0567 0583 0599 0615 0647 0663 0647 0663 0647 0663 0679 0695 0711 0727 0743 0759	0520 0536 0552 0568 0584 0600 0616 0632 0648 0680 0696 0712 0742 0744 0760	0521 0537 0553 0569 0585 0601 0617 0633 0649 0665 0681 0697 0713 0745 0745 0761	0522 0538 0554 0570 0602 0618 0634 0634 0666 0682 0698 0714 0730 0746 0762	0523 0539 0555 0571 0603 0619 0635 0651 0683 0699 0715 0731 0747 0763	0524 0540 0556 0572 0588 0604 0620 0636 0652 0688 0684 0700 0716 0732 0748 0764	0525 0541 0557 0573 0589 0621 0637 0653 0669 0665 0701 0717 0733 0749 0765	U526 0542 0558 0574 0590 0606 0622 0638 0654 0638 0654 0686 0702 0686 0702 0718 0734 0750 0766	U527 0543 0559 0575 0607 U623 0607 0665 0671 0687 0703 0719 0735 U751 0767
	21 22 23 24 25 26 27 28 29 2A 29 2A 2D 2C 2D 2E	0528 0544 0560 0576 0608 0624 0640 0656 0672 0688 0704 0720 0736	0529 0545 0561 0577 0593 0609 0625 0641 0657 0673 0673 0705 0721 0737	0530 0546 0562 0578 0610 0626 0642 0658 0674 0658 0674 0679 0706 0702 0738	0531 0547 0563 0579 0611 0627 0643 0659 0675 0691 0707 0723 0739	0532 0548 0564 0580 0596 0612 0628 0644 0660 0676 0692 0708 0724 0740	0533 0549 0565 0581 0597 0613 0629 0645 0661 0677 0693 0709 0725 0741	0534 0550 0566 0582 0598 0614 0630 0646 0662 0678 0694 0710 0726 0742	0535 0551 0567 0583 0599 0615 0647 0663 0647 0663 0679 0695 0711 0727 0743	0536 0552 0568 0584 0600 0616 0632 0648 0664 0680 0696 0712 0728 0744	0537 0553 0569 0585 0601 0617 0633 0649 0665 0649 0665 0681 0697 0713 0729 0745	0538 0554 0570 0586 0602 0618 0634 0650 0666 0682 0698 0714 0730 0746	0539 0555 0571 0587 0603 0619 0635 0651 0667 0683 0699 0715 0731 0747	0540 0556 0572 0588 0604 0620 0636 0652 0668 0684 0700 0716 0732 0748	0541 0557 0573 0589 0605 0621 0637 0653 0669 0685 0701 0717 0733 0749	0542 0558 0574 0590 0606 0622 0638 0654 0670 0686 0702 0718 0734 0734	0543 0559 0575 0591 0607 0623 0639 0655 0671 0687 0703 0719 0735 0751

1

D-2

	0	1	2	3	4	5	6	7	8	9	A	В	С	D	Ε	F
40 41 43 44 45 46 47 48 47 48 49 44 40 42 44 44 44 44 44 44 44 44 44 44 44 44	1024 1040 1056 1072 1088 1104 1120 1136 1152 1168 1184 1200 1216 1232 1248 1264	1025 1041 1057 1073 1089 1105 1121 1137 1153 1169 1185 1201 1217 1233 1249 1265	1026 1042 1058 1074 1090 1106 1122 1138 1154 1170 1186 1202 1218 1234 1250 1266	1027 1043 1059 1075 1091 1107 1123 1139 1155 1171 1185 1219 1235 1219 1235 1251 1267	1028 1044 1060 1076 1092 1108 1124 1140 1156 1172 1188 1204 1220 1236 1252 1268	1029 1045 1061 1077 1093 1109 1125 1141 1157 1173 1189 1205 1221 1237 1253 1269	1030 1046 1062 1078 1094 1110 1126 1142 1158 1174 1190 1206 1222 1238 1254 1270	1031 1047 1063 1079 1095 1111 1127 1143 1159 1175 1191 1207 1223 1239 1255 1271	1032 1048 1064 1080 1012 1128 1144 1160 1172 1208 1224 1208 1224 1240 1256 1272	1033 1049 1065 1081 1097 1113 1129 1145 1161 1177 1193 1209 1225 1241 1257 1273	1034 1050 1066 1082 1098 1114 1130 1146 1162 1178 1194 1210 1226 1242 1258 1274	1035 1051 1067 1083 1099 1115 1131 1147 1163 1179 1211 1227 1243 1259 1275	1036 1052 1068 1084 1106 1116 1132 1148 1164 1186 1212 1228 1244 1260 1276	1037 1053 1069 1085 1101 1117 1133 1149 1165 1181 1197 1213 1229 1245 1261 1277	1038 1054 1070 1085 1102 1118 1134 1150 1166 1182 1198 1214 1230 1246 1262 1278	1039 1055 1071 1087 1103 1119 1135 1151 1167 1183 1199 1215 1231 1247 1263 1279
	U	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F
50 51 52 53 54 55 56 57 58 59 58 59 58 50 55 50 55 55 55 55 55 55 55 55 55 55	1280 1296 1312 1328 1344 1360 1376 1392 1408 1424 1448 1426 1472 1488 1504 1520	1281 1297 1313 1329 1345 1361 1377 1393 1409 1425 1445 1457 1473 1489 1505 1521	1282 1298 1314 1330 1362 1378 1394 1410 1426 1458 1474 1458 1474 1490 1506 1522	1283 1299 1315 1331 1347 1363 1379 1395 1411 1427 1443 1459 1475 1491 1507 1523	1284 1300 1316 1332 1348 1364 1380 1396 1412 1428 1448 1440 1476 1476 1492 1508 1524	1285 1301 1317 1333 1349 1365 1381 1397 1413 1429 1445 1461 1477 1493 1509 1525	1286 1302 1318 1334 1350 1366 1382 1398 1414 1430 1446 1462 1478 1494 1510 1526	1287 1303 1319 1335 1351 1367 1383 1399 1415 1487 1463 1447 1463 1479 1465 1511 1527	1288 1304 1320 1352 1365 1352 1384 1400 1416 1432 1448 1464 1480 1496 1512 1528	1289 1305 1321 1337 1353 1365 1401 1417 1433 1449 1465 1481 1497 1513 1529	1290 1306 1322 1338 1354 1370 1386 1402 1418 1430 1466 1482 1498 1514 1530	1291 1307 1323 1339 1355 1371 1387 1403 1419 1435 1451 1467 1483 1499 1515 1531	1292 1308 1324 1340 1352 1388 1404 1420 1452 1468 1452 1468 1484 1500 1516 1532	1293 1309 1325 1341 1357 1373 1389 1405 1421 1435 1469 1485 1501 1517 1533	1294 1310 1326 1342 1358 1374 1390 1406 1422 1438 1454 1470 1486 1502 1518 1534	1295 1311 1327 1343 1355 1391 1407 1423 1437 1455 1471 1487 1503 1519 1535
	0	1	2	3	4	5	6	7	8	9	A	В	С	D	Е	F۰
60 61 62 63 64 65 66 67 68 67 68 67 68 66 66 66 66 66	1536 1552 1568 1584 1616 1632 1648 1668 1696 1712 1728 1744 176	1537 1553 1569 1585 1601 1617 1633 1649 1665 1681 1697 1713 1729 1745 1761 1777	1538 1554 1570 1586 1602 1618 1634 1650 16682 1698 1714 1730 1746 1762 1778	1539 1555 1571 1587 1603 1619 1635 1651 1663 1699 1715 1731 1743 1779	1540 1556 1572 1588 1604 1620 1636 1652 1668 1684 1700 1716 1732 1748 1768	1541 1557 1573 1589 1605 1621 1637 1653 1665 1701 1717 1733 1749 1765 1781	1542 1558 1574 1590 1606 1622 1638 1654 1670 1702 1718 1734 1734 1750 1760 1782	1543 1559 1575 1591 1607 1623 1639 1655 1671 1703 1719 1735 1751 1767 1783	1544 1560 1576 1608 1624 1640 1656 1672 1688 1704 1720 1736 1758 1784	1545 1561 1577 1593 1609 1625 1641 1657 1673 1689 1705 1721 1737 1753 1769 1785	1546 1562 1578 1594 1610 1626 1642 1658 1642 1658 1690 1706 1722 1738 1754 1774	1547 1563 1575 1611 1627 1643 1659 1659 1691 1707 1723 1739 1755 1771 1787	1548 1564 1580 1612 1628 1644 1660 1676 1676 1708 1724 1740 1756 1772 1788	1549 1565 1581 1613 1629 1645 1661 1673 1709 1725 1741 1753 1789	1550 1566 1582 1614 1630 1646 1662 1678 1694 1710 1726 1742 1758 1774 1790	1551 1567 1583 1599 1615 1631 1647 1663 1675 1711 1727 1743 1759 1775 1791
	Ο	1	2	3	4	5	6	7	8	9	A	В	С	D	E	F
70 71 72 73 74 75 76 77 78 79 78	1792 1808 1824 1840 1856 1872 1888 1904 1920 1936	1793 1809 1825 1841 1857 1873 1889 1905 1921 1937	1794 1810 1826 1842 1858 1874 1890 1906 1922 1938 1954	1795 1811 1827 1843 1859 1875 1891 1907 1923 1939	1796 1812 1828 1844 1860 1876 1892 1908 1924 1940 1956	1797 1813 1829 1845 1861 1877 1893 1909 1925 1941 1957	1798 1814 1830 1846 1862 1878 1894 1910 1926 1942 1958	1799 1815 1831 1847 1863 1879 1895 1911 1927 1943 1959	1800 1816 1832 1848 1864 1880 1896 1912 1928 1944 1960	1801 1817 1833 1849 1865 1881 1897 1913 1929 1945 1961	1802 1818 1834 1850 1866 1882 1898 1914 1930 1946 1962	1803 1819 1835 1851 1867 1883 1899 1915 1931 1947 1963	1804 1820 1836 1852 1868 1884 1900 1916 1932 1948 1964	1805 1821 1837 1853 1869 1885 1901 1917 1933 1949 1965	1806 1822 1838 1854 1870 1886 1902 1918 1934 1950 1966	1807 1823 1839 1855 1871 1887 1903 1919 1935 1951 1967

D-3

82 83	0 2048 2064	1 2049 2065	2 2050 2066	3 2051 2067	20	068	052 2053 068 2069	052 2053 2054 068 2069 2070	052 2053 2054 2055 068 2069 2070 2071	052 2053 2054 2055 2056 068 2069 2070 2071 2072	052 2053 2054 2055 2056 2057 068 2069 2070 2071 2072 2073	052 2053 2054 2055 2056 2057 2058 068 2069 2070 2071 2072 2073 2074	052 2053 2054 2055 2056 2057 2058 2059 068 2069 2070 2071 2072 2073 2074 2075	052 2053 2054 2055 2056 2057 2058 2059 2060 068 2069 2070 2071 2072 2073 2074 2075 2076	052 2053 2054 2055 2056 2057 2058 2059 2060 2061 068 2069 2070 2071 2072 2073 2074 2075 2076 2077	052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078
84 85 86 87 88 89 88 88 88	2080 2096 2112 2128 2144 2160 2176 2192 2208 2224	2081 2097 2113 2129 2145 2161 2177 2193 2209 2225	2082 2098 2114 2130 2146 2162 2178 2194 2210 2226	2083 2099 2115 2131 2147 2163 2179 2195 2211 2227	2084 2100 2116 2132 2148 2164 2180 2196 2212 2228		2101 2117 2133 2149 2165 2181 2197 2213	2101 2102 2117 2118 2133 2134 2149 2150 2165 2166 2181 2182 2197 2198 2213 2214	2101 2102 2103 2117 2118 2119 2133 2134 2135 2149 2150 2151 2165 2162 2163 2181 2182 2183 2197 2198 2199 2213 2214 2215	2101 2102 2103 2104 2117 2118 2119 2120 2133 2134 2135 2136 2149 2150 2151 2152 2165 2166 2167 2168 2181 2182 2183 2184 2197 2198 2199 2200 2213 2214 2215 2216	2101 2102 2103 2104 2105 2117 2118 2119 2120 2121 2133 2134 2135 2136 2137 2149 2150 2151 2152 2153 2165 2166 2167 2168 2169 2181 2182 2183 2184 2185 2197 2198 2199 2200 2201 2213 2214 2215 2216 2217	2101 2102 2103 2104 2105 2106 2117 2118 2119 2120 2121 2122 2133 2134 2135 2136 2137 2138 2149 2150 2151 2152 2153 2154 2165 2166 2167 2168 2169 2170 2181 2182 2183 2184 2185 2186 2197 2198 2199 2200 2201 2202 2213 2214 2215 2216 2217 2218	2101 2102 2103 2104 2105 2106 2107 2117 2118 2119 2120 2121 2122 2123 2133 2134 2135 2136 2137 2138 2139 2149 2150 2151 2152 2153 2154 2157 2165 2166 2167 2168 2169 2170 2171 2181 2182 2183 2184 2185 2186 2187 2197 2198 2199 2200 2201 2202 2203 2213 2214 2215 2216 2217 2218 2219	2101 2102 2103 2104 2105 2106 2107 2108 2117 2118 2119 2120 2121 2122 2123 2124 2133 2134 2135 2136 2137 2138 2139 2140 2149 2150 2151 2152 2153 2154 2155 2156 2165 2166 2167 2168 2169 2170 2171 2172 2181 2182 2183 2184 2185 2186 2187 2188 2197 2198 2199 2200 2201 2203 2204 2213 2214 2215 2216 2217 2218 2219 2220	2101 2102 2103 2104 2105 2106 2107 2108 2109 2117 2118 2119 2120 2121 2122 2123 2124 2125 2133 2134 2135 2136 2137 2138 2139 2140 2141 2149 2150 2151 2152 2153 2154 2155 2156 2157 2165 2166 2167 2168 2169 2170 2171 2172 2173 2181 2182 2183 2184 2185 2186 2187 2188 2189 2197 2198 2199 2200 2201 2202 2204 2205 2213 2214 2215 2216 2217 2218 2219 2220 2221	2101 2102 2103 2104 2105 2106 2107 2108 2109 2110 2117 2118 2119 2120 2121 2122 2123 2124 2125 2126 2133 2134 2135 2136 2137 2138 2139 2140 2141 2142 2149 2150 2151 2152 2153 2154 2155 2156 2157 2158 2165 2166 2167 2168 2189 2170 2171 2172 2173 2174 2181 2182 2183 2184 2185 2186 2187 2188 2189 2190 2197 2198 2190 2201 2202 2203 2204 2205 2206 2213 2214 2215 2216 2217 2218 2219 2220 2221 2222
8C 8D 8E 8F	2224 2240 2256 2272 2288	2223 2241 2257 2273 2289	2220 2242 2258 2274 2290	22243 2259 2275 2291	2228 2244 2260 2276 2292		2225 2245 2261 2277 2293	2245 2246 2261 2262 2277 2278	2245 2246 2247 2261 2262 2263 2277 2278 2279	224522462247224822612262226322642277227822792280	224522462247224822492261226222632264226522772278227922802281	224522462247224822492250226122622263226422652266227722782279228022812282	224522462247224822492250225122612262226322642265226622672277227822792280228122822283	224522462247224822492250225122522261226222632264226522662267226822772278227922802281228222832284	224522462247224822492250225122522253226122622263226422652266226722682269227722782279228022812282228322842285	224522462247224822492250225122522253225422612262226322642265226622672268226922702277227822792280228122822283228422852286
00	U 220 (1	2	3	4		5			-	-	-	-	-		
90 91 93 94 95 96 97 99 99 99 99 99 99 99 99 99 99 99 99	2304 2320 2336 2352 2368 2384 2400 2416 2432 2448 2448 2464 2480 2496 2512 2528 2544	2305 2321 2337 2353 2369 2385 2401 2417 2433 2449 2465 2481 2497 2513 2529 2545	2306 2322 2338 2354 2370 2386 2402 2418 2430 2466 2482 2498 2546 2546	2307 2323 2355 2357 2387 2403 2419 2431 2467 2483 2499 2515 2531 2547	2308 2324 2340 2356 2372 2388 2404 2420 2452 2468 2452 2468 2452 2468 2452 2548		2309 2325 2341 2357 2373 2389 2405 24237 2453 2469 2485 25017 2533 2549	2325 2326 2341 2342 2357 2358 2373 2374 2389 2390 2405 2406 2421 2422 2437 2438 2453 2454 2469 2470 2485 2486 2501 2502 2517 2518 2533 2534 2549 2550	2325 2326 2327 2341 2342 2343 2357 2358 2359 2373 2374 2375 2389 2390 2391 2405 2406 2407 2421 2422 2423 2437 2438 2439 2453 2454 2455 2469 2470 2471 2485 2486 2487 2501 2502 2503 2517 2518 2519 2533 2534 2535 2549 2550 2551	2325 2326 2327 2328 2341 2342 2343 2344 2357 2358 2359 2360 2373 2374 2375 2376 2389 2390 2391 2392 2405 2406 2407 2408 2421 2422 2423 2424 2437 2438 2439 2440 2453 2454 2455 2456 2465 2466 2470 2471 2472 2485 2486 2487 2488 2501 2502 2503 2504 2517 2518 2519 2520 2533 2534 2535 2536 2549 2550 2551 2552 2551 2552	2325 2326 2327 2328 2329 2341 2342 2343 2344 2345 2357 2358 2359 2360 2361 2373 2374 2375 2376 2377 2389 2390 2391 2392 2393 2405 2406 2407 2408 2409 2421 2422 2423 2424 2425 2437 2438 2439 2440 2441 2453 2454 2455 2456 2457 2465 2457 2456 2457 2463 2455 2456 2457 2473 2488 2489 2501 2502 2503 2504 2505 2517 2518 2519 2520 2521 2533 2534 2535 2536 2537 2549 2550 2551 2552 2553	2325 2326 2327 2328 2329 2330 2341 2342 2343 2344 2345 2346 2357 2358 2359 2360 2361 2362 2373 2374 2375 2376 2377 2378 2389 2390 2391 2392 2393 2394 2405 2406 2407 2408 2409 2410 2421 2422 2423 2424 2425 2426 2437 2438 2439 2440 2441 2442 2453 2454 2455 2456 2457 2458 2465 2460 2471 2473 2474 2485 2485 2486 2487 2488 2489 2490 2501 2502 2503 2504 2505 2506 2517 2518 2519 2520 2521 2522 2533 2534 2551 2552 2553 2554	2325 2326 2327 2328 2329 2330 2331 2341 2342 2343 2344 2345 2346 2347 2357 2358 2359 2360 2361 2362 2363 2373 2374 2375 2376 2377 2378 2379 2389 2390 2391 2392 2393 2394 2395 2405 2406 2407 2408 2409 2410 2411 2421 2422 2423 2424 2425 2426 2427 2437 2438 2439 2440 2441 2442 2443 2453 2454 2455 2456 2457 2458 2459 2469 2470 2471 2472 2473 2474 2475 2485 2486 2487 2488 2489 2490 2491 2501 2502 2503 2504 2505 2506 2507 2517 2518 2519 2520 2521 2522 <td>2325 2326 2327 2328 2329 2330 2331 2332 2341 2342 2343 2344 2345 2346 2347 2348 2357 2358 2359 2360 2361 2362 2363 2364 2373 2374 2375 2376 2377 2378 2379 2380 2389 2390 2391 2392 2393 2394 2395 2396 2405 2406 2407 2408 2409 2410 2411 2412 2421 2422 2423 2424 2425 2426 2427 2428 2437 2438 2439 2440 2441 2442 2443 2444 2453 2456 2457 2458 2459 2460 2469 2470 2471 2472 2473 2474 2475 2476 2485 2486 2487 2488 2489 2490 2491 2492 2501 2502 2503 2504 2505<!--</td--><td>2325 2326 2327 2328 2329 2330 2331 2332 2333 2341 2342 2343 2344 2345 2346 2347 2348 2349 2357 2358 2359 2360 2361 2362 2363 2364 2365 2373 2374 2375 2376 2377 2378 2379 2380 2381 2389 2390 2391 2392 2393 2394 2395 2396 2397 2405 2406 2407 2408 2409 2410 2411 2412 2413 2421 2422 2423 2424 2425 2426 2427 2428 2429 2437 2438 2439 2440 2441 2442 2443 2444 2445 2453 2454 2455 2456 2457 2458 2459 2460 2461 2469 2470 2471 2472 2473 2474 2475 2476 2477 2485 2486<</td><td>2325 2326 2327 2328 2329 2330 2331 2332 2333 2334 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2357 2358 2359 2360 2361 2362 2363 2364 2365 2366 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414 2421 2422 2423 2424 2425 2426 2427 2428 2429 2430 2437 2438 2439 2440 2441 2442 2443 2444 2445 2446 2433 2454 2455 2456 2457 2458 2459 2460 2461 2462 2469 2470 2471</td></td>	2325 2326 2327 2328 2329 2330 2331 2332 2341 2342 2343 2344 2345 2346 2347 2348 2357 2358 2359 2360 2361 2362 2363 2364 2373 2374 2375 2376 2377 2378 2379 2380 2389 2390 2391 2392 2393 2394 2395 2396 2405 2406 2407 2408 2409 2410 2411 2412 2421 2422 2423 2424 2425 2426 2427 2428 2437 2438 2439 2440 2441 2442 2443 2444 2453 2456 2457 2458 2459 2460 2469 2470 2471 2472 2473 2474 2475 2476 2485 2486 2487 2488 2489 2490 2491 2492 2501 2502 2503 2504 2505 </td <td>2325 2326 2327 2328 2329 2330 2331 2332 2333 2341 2342 2343 2344 2345 2346 2347 2348 2349 2357 2358 2359 2360 2361 2362 2363 2364 2365 2373 2374 2375 2376 2377 2378 2379 2380 2381 2389 2390 2391 2392 2393 2394 2395 2396 2397 2405 2406 2407 2408 2409 2410 2411 2412 2413 2421 2422 2423 2424 2425 2426 2427 2428 2429 2437 2438 2439 2440 2441 2442 2443 2444 2445 2453 2454 2455 2456 2457 2458 2459 2460 2461 2469 2470 2471 2472 2473 2474 2475 2476 2477 2485 2486<</td> <td>2325 2326 2327 2328 2329 2330 2331 2332 2333 2334 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2357 2358 2359 2360 2361 2362 2363 2364 2365 2366 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414 2421 2422 2423 2424 2425 2426 2427 2428 2429 2430 2437 2438 2439 2440 2441 2442 2443 2444 2445 2446 2433 2454 2455 2456 2457 2458 2459 2460 2461 2462 2469 2470 2471</td>	2325 2326 2327 2328 2329 2330 2331 2332 2333 2341 2342 2343 2344 2345 2346 2347 2348 2349 2357 2358 2359 2360 2361 2362 2363 2364 2365 2373 2374 2375 2376 2377 2378 2379 2380 2381 2389 2390 2391 2392 2393 2394 2395 2396 2397 2405 2406 2407 2408 2409 2410 2411 2412 2413 2421 2422 2423 2424 2425 2426 2427 2428 2429 2437 2438 2439 2440 2441 2442 2443 2444 2445 2453 2454 2455 2456 2457 2458 2459 2460 2461 2469 2470 2471 2472 2473 2474 2475 2476 2477 2485 2486<	2325 2326 2327 2328 2329 2330 2331 2332 2333 2334 2341 2342 2343 2344 2345 2346 2347 2348 2349 2350 2357 2358 2359 2360 2361 2362 2363 2364 2365 2366 2373 2374 2375 2376 2377 2378 2379 2380 2381 2382 2389 2390 2391 2392 2393 2394 2395 2396 2397 2398 2405 2406 2407 2408 2409 2410 2411 2412 2413 2414 2421 2422 2423 2424 2425 2426 2427 2428 2429 2430 2437 2438 2439 2440 2441 2442 2443 2444 2445 2446 2433 2454 2455 2456 2457 2458 2459 2460 2461 2462 2469 2470 2471
	0	1	2	3	4		5									
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9	2560 2576 2592 2608 2624 2640 2656 2672 2688 2704 2720 2736 2752 2768 2784 2800	2561 2577 2593 2693 2625- 2641 2657 2673 2689 2705 2721 2737 2753 2769 2785 2801	2562 2578 2594 2610 2626 2642 2658 2674 2690 2706 2722 2738 2770 2786 2802	2563 2579 2595 2611 2627 2643 2659 2675 2691 2707 2723 2737 2723 2771 2787 2803	2564 2580 2596 2612 2628 2644 2660 2676 2692 2708 2724 2708 2724 2740 2756 2772 2788 2804		2565 2581 2597 2613 2629 2641 2629 2661 26677 26693 2709 2725 2741 773 789 805	2581 2582 2597 2598 2613 2614 1629 2630 2645 2646 2661 2662 2677 2678 2693 2694 709 2710 725 2726 7741 2742 773 2754 789 2790 805 2806	2581 2582 2583 2597 2598 2599 2613 2614 2615 3629 2643 2646 2645 2646 2647 3661 2662 2663 3677 2678 2679 3693 2694 2695 709 2710 2711 725 2726 2727 741 2742 2743 757 2758 2759 805 2806 2807	2581 2582 2583 2584 2597 2598 2599 2600 2613 2614 2615 2616 2623 2644 2645 2646 2645 2664 2663 2664 2677 2678 2679 2680 2693 2694 2695 2696 2709 2710 2711 2712 2725 2726 2727 2728 2741 2742 2743 2744 773 2758 2759 2766 773 2774 2775 2776 789 2790 2791 2792 805 2806 2807 2808	2581 2582 2583 2584 2585 2597 2598 2599 2600 2601 2613 2614 2615 2616 2617 2629 2630 2631 2632 2633 2645 2646 2647 2648 2649 2661 2662 2663 2664 2665 2677 2678 2679 2680 2681 2693 2694 2695 2696 2697 2709 2710 2711 2712 2713 2725 2726 2727 2728 2729 741 2742 2743 2744 2745 2774 2775 2776 2777 2788 2799 2790 2791 2792 2793 805 2806 2807 2808 2809	2581 2582 2583 2584 2585 2586 2597 2598 2599 2600 2601 2602 2613 2614 2615 2616 2617 2618 2629 2630 2631 2632 2633 2634 2645 2646 2647 2648 2649 2650 2661 2662 2663 2664 2665 2666 2677 2678 2679 2680 2681 2682 2693 2694 2695 2696 2697 2698 2709 2710 2711 2712 2713 2714 725 2726 2727 2728 2729 2730 741 2742 2743 2744 2745 2746 773 2758 2759 2760 2761 2772 778 2790 2791 2792 2793 2794 805 2806 2807 2808 2809 2810	2581 2582 2583 2584 2585 2586 2587 2597 2598 2599 2600 2601 2602 2603 2613 2614 2615 2616 2617 2618 2619 2629 2630 2631 2632 2633 2634 2635 2645 2646 2647 2648 2649 2650 2651 2661 2662 2663 2664 2665 2666 2667 2679 2680 2681 2682 2683 2693 2693 2694 2695 2696 2697 2698 2699 2709 2710 2711 2712 2713 2714 2715 2725 2726 2727 2728 2729 2730 2731 2741 2742 2743 2744 2745 2764 2747 2758 2759 2760 2761 2762 2778 2779 2805 2806 2807 2808 2809 2810 2811 <td>2581 2582 2583 2584 2585 2586 2587 2588 2597 2598 2599 2600 2601 2602 2603 2604 2613 2614 2615 2616 2617 2618 2619 2620 2629 2630 2631 2632 2633 2634 2655 2636 2645 2664 2647 2648 2649 2650 2651 2652 2661 2662 2663 2664 2665 2666 2667 2668 2679 2679 2680 2681 2682 2684 2699 2700 709 2710 2711 2712 2713 2714 2715 2716 774 2742 2743 2744 2745 2746 2747 2748 757 2758 2759 2760 2761 2762 2763 2764 7774 2775 2776 2777 2778 2779 2780 789 2790 2791 2792</td> <td>2581 2582 2583 2584 2585 2586 2587 2588 2589 2597 2598 2599 2600 2601 2602 2603 2604 2605 2613 2614 2615 2616 2617 2618 2619 2620 2621 2629 2630 2631 2632 2633 2634 2635 2636 2637 2645 2664 2663 2664 2665 2666 2667 2668 2669 2677 2678 2679 2680 2681 2682 2683 2684 2685 2693 2694 2695 2696 2697 2698 2699 2700 2701 709 2710 2711 2712 2713 2714 2715 2716 2717 725 2726 2727 2728 2729 2730 2731 2732 2733 741 2742 2743 2744 2745 2764 2764 2764 2764 773 2758</td> <td>2581 2582 2583 2584 2585 2586 2587 2588 2589 2590 2597 2598 2599 2600 2601 2602 2603 2604 2605 2606 2613 2614 2615 2616 2617 2618 2619 2620 2621 2622 2629 2630 2631 2632 2633 2634 2635 2636 2637 2638 2645 2664 2663 2664 2665 2666 2667 2668 2669 2670 2677 2678 2679 2680 2681 2682 2683 2684 2685 2686 2693 2694 2695 2696 2697 2698 2699 2700 2701 2702 709 2710 2711 2712 2713 2714 2715 2716 2717 2718 725 2726 2727 2728 2729 2730 2731 2732 2733 2734 741 2742 2743</td>	2581 2582 2583 2584 2585 2586 2587 2588 2597 2598 2599 2600 2601 2602 2603 2604 2613 2614 2615 2616 2617 2618 2619 2620 2629 2630 2631 2632 2633 2634 2655 2636 2645 2664 2647 2648 2649 2650 2651 2652 2661 2662 2663 2664 2665 2666 2667 2668 2679 2679 2680 2681 2682 2684 2699 2700 709 2710 2711 2712 2713 2714 2715 2716 774 2742 2743 2744 2745 2746 2747 2748 757 2758 2759 2760 2761 2762 2763 2764 7774 2775 2776 2777 2778 2779 2780 789 2790 2791 2792	2581 2582 2583 2584 2585 2586 2587 2588 2589 2597 2598 2599 2600 2601 2602 2603 2604 2605 2613 2614 2615 2616 2617 2618 2619 2620 2621 2629 2630 2631 2632 2633 2634 2635 2636 2637 2645 2664 2663 2664 2665 2666 2667 2668 2669 2677 2678 2679 2680 2681 2682 2683 2684 2685 2693 2694 2695 2696 2697 2698 2699 2700 2701 709 2710 2711 2712 2713 2714 2715 2716 2717 725 2726 2727 2728 2729 2730 2731 2732 2733 741 2742 2743 2744 2745 2764 2764 2764 2764 773 2758	2581 2582 2583 2584 2585 2586 2587 2588 2589 2590 2597 2598 2599 2600 2601 2602 2603 2604 2605 2606 2613 2614 2615 2616 2617 2618 2619 2620 2621 2622 2629 2630 2631 2632 2633 2634 2635 2636 2637 2638 2645 2664 2663 2664 2665 2666 2667 2668 2669 2670 2677 2678 2679 2680 2681 2682 2683 2684 2685 2686 2693 2694 2695 2696 2697 2698 2699 2700 2701 2702 709 2710 2711 2712 2713 2714 2715 2716 2717 2718 725 2726 2727 2728 2729 2730 2731 2732 2733 2734 741 2742 2743
AA AB ACO ADO AEO AFO			2	3	4	5		-		-						
AA AB ACO ADO AEO	0	1 2817	-					0000	L 2822 282 3	1 2822 2823 2824	1 2822 2823 2824 2825					1 2822 2823 2824 2825 2826 2827 2828 2829 2830 7 2838 2839 2840 2841 2842 2843 2844 2845 2846

	0	1	2	3	4	5	6	7	8	9	A	В	с	D	E	F
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA	3072 3088 3104 3120 3136 3152 3168 3184 3200 3216 3232	3073 3089 3105 3121 3137 3153 3169 3185 3201 3217 3233	3074 3090 3196 3122 3138 3154 3170 3186 3202 3218 3234	3075 3091 3107 3123 3139 3155 3171 3187 3203 3219 3235	3076 3092 3108 3124 3140 3156 3172 3188 3204 3220 3236	3077 3093 3109 3125 3141 3157 3173 3173 3189 3205 3221 3237	3078 3094 3110 3126 3142 3158 3174 3190 3206 3222 3238	3079 3095 3111 3127 3143 3159 3175 3191 3207 3223 3239	3080 3096 3112 3128 3144 3160 3176 3176 3192 3208 3224 3240	3081 3097 3113 3129 3145 3161 3177 3193 3209 3225 3241	3082 2098 3114 3130 3146 3162 3178 3194 3210 3226 3242	3083 3099 3115 3131 3147 3163 3179 3195 3211 3227 3243	3084 3100 3116 3132 3148 3164 3180 3196 3212 3228 3244	3085 3101 3117 3133 3149 3165 3181 3197 3213 3229 3245	3086 3102 3118 3134 3150 3166 3182 3198 3214 3230 3246	3087 3103 3119 3135 3151 3167 3183 3199 3215 3231 3247
CB CC CD CE CF	3248 3264 3280 3296 3312	3249 3265 3281 3297 3313	3250 3266 3282 3298 3314	3251 3267 3283 3299 3315	3252 3268 3284 3300 3316	3253 3269 3285 3301 3317	3254 3270 3286 3302 3318	3255 3271 3287 3303 3319	3256 3272 3288 3304 3320	3257 3273 3289 3305 3321	3258 3274 3290 3306 3322	3259 3275 3291 3307 3323	3260 3276 3292 3308 3324	3261 3277 3293 3309 3325	3262 3278 3294 3310 3326	3263 3279 3295 3311 3327
	0	1	2	3	4	5	6	7	8	9	А	В	С	D	E	F
D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA D8 D9 DA DB DC DD DE DF	3328 3344 3360 3376 3492 3424 3440 3456 3472 3488 3504 3520 3536 3552 3568	3329 3345 3361 3377 3393 3425 3441 3457 3473 3489 3505 3521 3537 3553 3569	3330 3346 3362 3378 3410 3426 3442 3458 3474 3490 3506 3522 3538 3554 3570	3331 3347 3363 3379 3395 3415 3427 3443 3459 3475 3443 3459 3475 3597 3523 3539 3555 3571	3332 3348 3364 3380 3392 3428 3428 3444 3460 3476 3492 3504 3524 3524 3556 3572	3333 3349 3365 3381 3429 3445 3445 3445 3445 3445 3445 3445 344	3334 3350 3366 3382 3398 3430 3446 3430 3446 3462 3478 3490 3526 3526 3528 3558 3574	3335 3351 3367 3383 3495 3431 3447 3463 3479 3463 3479 3451 3527 3543 3559 3575	3336 3352 3368 3384 3400 3416 3432 3448 3464 3480 3464 3480 3512 3528 3528 3544 3560 3576	3337 3353 3369 3385 3401 3417 3433 3449 3465 3481 3497 35129 3529 3545 3561 3577	3338 3354 3370 3386 3402 3418 3434 3430 3446 3482 3498 3514 3530 3546 3530 3546 3546 3546 3546 3546 3546 3546 3546	3339 3355 3371 3387 3403 3435 3451 3455 3451 3467 3483 3495 3515 3531 3547 3563 3579	3340 3356 3372 3388 3404 3436 3436 3436 3436 3444 3506 3516 3532 3548 3564 3580	3341 3357 3373 3427 3425 3425 3437 3453 3469 3485 3507 3517 3517 3533 3549 3565 3581	3342 3358 3374 3390 3406 3422 3438 3454 3470 3486 3502 3518 3534 3550 3566 3582	3343 3359 3375 3391 3423 3423 3439 3455 3471 3487 3503 3519 3535 3551 3567 3583
	0	1	2	3	4	5	6	7	8	9	А	В	С	D	E	F
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E8 E9 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1 E1	3584 3600 3616 3632 3648 3680 3696 3712 3728 3744 3760 3776 3792 3808 3824	3585 3601 3617 3633 3649 3649 3768 3713 3745 3745 3745 3745 3761 3777 3793 3809 3825	3586 3602 3618 3634 3650 3682 3698 3714 3736 3746 3746 3746 3778 3794 3810 3826	3587 3603 3619 3635 3651 3683 3699 3715 3747 3747 3747 3747 3747 3747 3747 374	3588 3604 3620 3636 3652 3684 3700 3716 3748 3748 3748 3748 3748 3748 3748 3796 3812 3828	3589 3605 3621 3637 3653 3685 3701 3717 3733 3749 3749 3745 3749 3745 3781 3797 3813 3829	3590 3606 3622 3638 3654 3702 3718 3750 3750 3750 3750 3756 3782 3798 3814 3830	3591 3607 3623 3639 3655 3687 3703 3719 3735 3751 3751 3751 3767 3783 3799 3815 3831	3592 3608 3624 3640 3656 3688 3704 3720 3752 3752 3758 3752 3768 3752 3768 3752 3768 3752 3768 3752 3768 3752 3768 3832	3593 3609 3625 3641 3657 3689 3705 3721 3753 3753 3753 3759 3785 3801 3817 3833	3594 3610 3626 3642 3658 3674 3706 3706 3722 3754 3754 3754 3770 3786 3802 3818 3834	3595 3611 3627 3643 3659 3691 3707 3723 3755 3771 3787 3803 3819 3835	3596 3612 3628 3644 3666 3676 3692 3708 3724 3756 3756 3756 3772 3788 3804 3820 3836	3597 3613 3629 3645 3645 3677 3693 3709 3725 3745 3757 3773 3789 3805 3821 3837	3598 3614 3630 3646 3678 3694 3710 3726 3758 3774 3758 3774 3790 3806 3822 3838	3599 3615 3631 3647 3663 3679 3695 3711 3727 3743 3759 3775 3791 3807 3823 3839
	0	1	2	3	4	5	6	7	8	9	А	В	С	D	E	F
F0 F1 F2 F3 F5 F6 F7 F8 F7 F8 F8 F8 F8 F8 F7 F8 F7 F8 F7 F8 F7 F8 F7 F8 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7	3840 3856 3872 3888 3904 3920 3936 3952 3968 4000 4016 4032 4044 4068	3841 3857 3873 3905 3921 3937 3953 3965 4001 4017 4033 40045 4065 4081	3842 3858 3870 3906 3922 3938 3954 4002 4018 4002 4018 4030 4066 4082	3843 3859 3871 3907 3923 3939 3955 3971 3987 4003 4019 4003 4019 4035 4067 4063	3844 3860 3872 3908 3924 3940 3952 3988 4004 4020 40352 4068 4084	3845 3861 3873 3909 3925 3941 3957 3989 4005 4021 4037 4065 4069 4065	3846 3862 3874 3910 3926 3942 3954 3974 3990 4006 4022 4038 4006 4022 4038 4070 4086	3847 3863 3895 3911 3927 3943 3959 3959 4007 4023 4007 4023 4035 4071 4087	3848 3864 3896 3912 3928 3944 3976 3976 3992 4008 4024 4024 4056 4072 4088	3849 3865 3897 3913 3929 3945 3945 3945 3977 3993 4009 4025 4025 4057 4057 4057 4057	3850 3866 3898 3914 3930 3946 3978 3978 3978 4010 4026 4026 4058 4074 4090	3851 3867 3889 3915 3931 3947 3963 3979 3975 4011 4027 4043 4059 4075 4091	3852 3868 3980 3916 3932 3948 3948 3996 4012 4028 4012 4028 4040 4060 4076 4092	3853 3869 3881 3901 3917 3933 3945 3981 3997 4013 4029 4041 4029 4061 4077 4093	3854 3870 3882 3918 3934 3956 3982 3998 4014 4030 4042 4062 4062 4078 4094	3855 3871 3887 3903 3935 3951 3983 3999 4015 4031 4063 4063 4079 4095

APPENDIX E

OCTAL/DECIMAL TABLE

		<u> </u>								ĺ	<u> </u>							7
		0	1	2	3	4	5	6	7	<u> </u>	0	1	2	3	4	5	6	
0000 0000	0000			0002 0010						0410	0264	0265	0258 0266	0267	0268	0269	0270	0271
to to	0020	-		0018 0026				0022 0030					0274 0282					
0777 0511 (Octal) (Decimal)				0034						0440	0288	0289	0290	0291	0232	0293	0294	0295
	0050			0042 0050									0298 0306					
Octal Decimal													0314					
10000 - 4096	0100	0064	0065	0066	0067	0068	0069	0070	0071	0500	0320	0321	0322	0323	0324	0325	0326	0327
20000 8192 30000 12288	0110	0072	0073	0074	0075	0076	0077	0078	0079	0510			0330					
40000 - 16384 50000 - 20480		0080											0338 0346					
60000 · 24576	0140			0098									0354 0362					
70000 - 28672		0104								0560	0368	0369	0370	0371	0372	0373	0374	0375
	0170	0120	0121	0122	0123	0124	0125	0126	0127	0570	0376	0377	0378	0379	0380	0381	0362	0383
	0200	0128											0386					
	0210			01 38 0146									0394 0402					
	0230	0152	0153	0154	0155	0156	0157	0158	0159	0630	0408	0409	0410	0411	0412	0413	0414	0415
	0240	0160		0162									0418 0426					
	0260	0176	0177	0178	0179	0180	0181	0182	0183	0660	0432	0433	0434	0435	0436	04 37	0438	04 39
	0270	0184	0185	0186	0187	0188	0189	0190	0191	0670	0440	0441	0442	0443	0444	0445	0440	0447
	0300			0194									0450					
	0310	0200		0202 0210									0458 0466					
	0330	0216	0217	0218	0219	0220	0221	0222	0223				0474 0482					
	0340	0224		0226									0490					
		0240											0498 0506					
	[0310	0240	1	2	3	4	5	6	7	0.10				3	4	5	6	7
	[<u></u>										0	1	2					
1000 0512	1000			0514 0522						1400	0768	0769	0770 0778	0771	0772	0773	0782	0783
to to		0528								1420	0784	0785	0786	0787	0788	0789	0790	0791
1777 1023 (Octal) (Decimal)	1030			05 38 0546						1430	0792	0801	0794 0802	0795	0804	0805	0806	0807
	1050			0554 0562						1450	0808	0809	0810	0811	0812	0813	0814	0815
	1060			0570						1460	0816	0817	0818 0826	0827	0828	0829	0830	0831
	1100	0576	0577	0578	0579	0580	0581	0582	0581				0834				•	
	1110	0584	0585	0586	0587	0588	0589	0590	0591	1510	0840	0841	0842	0843	0844	0845	0846	0847
		0592																0855 0863
	1140	0608	0609	0610	0611	0612	0613	0614	0615	1540	0864	0865	0866	0867	0868	0869	0870	0871
		0616																0879 0887
	1170			0634														0895
		0640																0903
	1210			0650 0658									0906 0914					0911
	1230	0664	0665	0666	0667	0668	0669	0670	0671	1630	0920	0921	0922	0923	0924	C925	0926	0927
	1240	L (0674 0682						-			0930 0938					
	1260	0688	0689	0690	0691	0692	0693	0694	0695	1660	0944	0945	0946	0947	0948	0949	0950	0951
	1270	0696	0697	0038	0033	0700	0701	0702	0703	1670	0952	0953	0954	0955	J956	0957	0958	095 9
	1.100	0704	0705	0706									0962					
			071 -	0714			0111	0.10	0113	11710	0968	0969	0970	1071		CHO 77 7		0975
	1310	0712							0727	1720	0976		0978					
	1 310 1 320 1 330	0712 0720 0728	0721 0729	0722 0730	0723 0731	0724 07 32	0725 07 33	0726 0734	0735	1730	0984	0977 0985	0978 0986	0979 0 987	0980 0988	0981 0989	0982 0990	0983 0991
	1310 1320 1330 1340 1350	0712 0720 0728 0736 0744	0721 0729 0737 0745	0722 0730 0738 0746	0723 0731 0739 0747	0724 0732 0740 0748	0725 0733 0741 0749	0726 0734 0742 0750	0735 0743 0751	1730 1740 1750	0984 0992 1000	0977 0985 0993 1001	0978 0986 0994 1002	0979 0987 0995 1003	0980 0988 0996 1004	0981 0989 0997 1005	0982 0990 0998 1006	0983 0991 0999 1007
	1310 1320 1330 1340 1350	0712 0720 0728 0736 0744 0752	0721 0729 0737 0745 0753	0722 0730 0738 0746	0723 0731 0739 0747 0755	0724 0732 0740 0748 0756	0725 0733 0741 0749 0757	0726 0734 0742 0759 0758	0735 0743 0751 0759	1730 1740 1750 1760	0984 0992 1000 1008	0977 0985 0993 1001 1009	0978 0986 0994 1002 1010	0979 0987 0995 1003 1011	0980 0988 0996 1004 1012	0981 0989 0997 1005 1013	0982 0990 0998 1006 1014	0983 0991 0999

	<u> </u>								ſ			2	3	4	5	6	7
·	0	1	2	3	4	5	6	7		0	1						
2000 1024 2000	11032	1013	1014	1035	1036	1037	1038	10191	2400 2410	1200	1288	1730	1231	1292	1232	1234	1493
to to 2010 2777 1535 2020	11040	1041	1042	1043	1044	1045	1046	1047	24201	1296	1297	1298	1299	1 300	1301	1304	1303
(Octal) (Decimal) 2030 2040	11056	1057	1058	1059	1050	1061	1062	1061	2430 2440	1312	1313	1314	1315	1310	1317	1710	1218
2050	1064	1065	1066	1067	1068	1069	1070	1071	2450	1320	1 3 2 1	1322	1323	1324	1325	1326	1 3 2 7
Octal Decimal 2050	1072	1073	1074	1075	1076	1077	1078	1079	2460 2470	1328	1329	1330	1331	1332	1333	1342	1343
Octal Decimal 2070 10000 4096									1								
20000 8192 2100 30000 12288 2110	1088	1089	1090	1091	1092	1093	1094	1095	2500 2510	1344	1345	1346	1347	1348	1349	1350	1351
30000 12288 2110 40000 16384 2120	11104	1105	1106	1107	1108	1109	1110	1111	2520	1360	1361	1332	1363	1364	1365	1300	1 301
50000 - 20480 2130	1112	1113	1114	1115	1116	1117	1118	1119	2530	1368	1369	1370	1371	1372	1373	1374	1375
60000 - 24576 2140 70000 - 28672 2150	1128	1129	1130	1131	1132	1133	1134	1135	2550	1 384	1385	1386	1387	1388	1389	1390	1391
2160	11136	1137	1138	1139	1140	1141	1142	1143	2560 2570	1 3 9 2	1393	1394	1395	1 396	1397	1398	1 3 9 9
2170								1	1 1								
2200									2600 2610	1408	1409	1410	1411	1412	1413	1414	1415
2220	1160	1169	1170	1171	1172	1173	1174	1175	2620	1424	1425	1426	1427	1428	1429	1430	1431
2230	1176	1177	1178	1179	1180	1181	1182	1183	2630	1432	1433	1434	1435	1436	1437	1438	1439
	1184	1193	1194	1195	1196	1197	1198	1199	2650	1448	1449	1450	1451	1452	1453	1454	1455
2260	1200	1201	1202	1203	1204	1205	1206	1207	2660	1456	1457	1458	1459	1460	1461	1462	1463
2270	1208	1209	1210	1211	1712	1213	1214	1215	2010	1464	1403	1400	1401	1400	1403	14/0	14/1
2300			1218							1472							
2310	1224		1226							1480							
2330	1240	1241	1242	1243	1244	1245	1246	1247	2730	1496	1497	1498	1499	1500	1501	1502	1503
2340	1248	1249	1250	1251	1252	1253	1254	1255	2740	1504	1505	1506	1507	1508	1509	1510	1511
2360	1264	1265	1266	1267	1268	1269	1270	1271	2760	1520	1521	1522	1523	1524	1525	1526	1527
2370	1272	1273	1274	1275	1276	1277	1278	1279	2770	1528	1529	1530	1531	1532	1533	1534	1535
	0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
	0 1536			16.30			1647	1649	1400	1792	1701	1704	1705	1706	1707	1708	1700
3000 1536 30 to to 30	0 1544	5 15.17	1220	1238	1540	1549	1542	1551	1410	1800	1801	1807	1807	1130	1131	1130	1 +3 3
3777 2047 30			1240	1047	1240				13410			1005	1003	1804	1805	1800	1807
	0 1552	2 1553	1554	1555	1556	1557	1558	1559	3420	1808	1809	1810	1811	1812	1813	1814	1815
(Octal) (Decimal) 303 304	0 1552	2 1553 0 1561	1554	1555	1556 1564	1557 1565	1558 1566	1559 1567	3420 3430	1808	1809 1817	1810 1818	1811 1819	1812 1820	1813 1821	1814 1822	1815 1823
304	0 1552 0 1567 0 1567 0 1576	2 1553 0 1561 1 1569 5 1577	1554 1562 1570 1578	1555 1563 1571 1579	1556 1564 1572 1580	1557 1565 1573 1581	1558 1566 1574 1582	1559 1567 1575 1583	3420 3430 3440 3450	1808 1816 1824 1832	1809 1817 1825 1833	1810 1818 1826 1834	1811 1019 1827 1835	1812 1820 1828 1836	1813 1821 1829 1837	1814 1822 1830 1838	1815 1823 1831 1839
304 303 304	0 1552 0 1567 0 1568 0 1578 0 1584	2 1553 0 1561 1 1569 5 1577 1 1585	1554 1562 1570 1578 1586	1555 1563 1571 1579 1587	1556 1564 1572 1580 1588	1557 1565 1573 1581 1589	1558 1566 1574 1582 1590	1559 1567 1575 1583 1583	3420 3430 3440 3450 3460	1808 1816 1824 1832 1840	1809 1817 1825 1833 1841	1810 1818 1826 1834 1842	1811 1819 1827 1835 1843	1812 1820 1828 1836 1844	1813 1821 1829 1837 1845	1814 1822 1830 1838 1846	1815 1823 1831 1839 1847
304 301 301 301 301	0 1552 0 1567 0 1568 0 1578 0 1584 0 1584 0 1592	2 1553 0 1561 1 1569 5 1577 1 1585 2 1593	1554 1562 1570 1578 1578 1586	1555 1563 1571 1579 1587 1587	1556 1564 1572 1580 1588 1596	1557 1565 1573 1581 1589 1597	1558 1566 1574 1582 1590 1598	1559 1567 1575 1583 1591 1591	3420 3430 3440 3450 3450 3460 3470	1808 1816 1824 1832 1840 1848	1809 1817 1825 1833 1841 1849	1810 1818 1826 1834 1842 1850	1811 1819 1827 1835 1843 1851	1812 1820 1828 1836 1844 1852	1813 1821 1829 1837 1845 1853	1814 1822 1830 1838 1846 1854	1815 1823 1831 1839 1847 1855
304 305 304 305 307 316	0 1552 0 1567 0 1568 0 1578 0 1584	2 1553 0 1561 1 1569 5 1577 1 1585 2 1593 0 1601	1554 1562 1570 1578 1586 1586 1594	1555 1563 1571 1579 1587 1595 1603	1556 1564 1572 1580 1588 1596	1557 1565 1573 1581 1589 1597	1558 1566 1574 1582 1590 1598 1606	1559 1567 1575 1583 1591 1599 1607	3420 3430 3440 3450 3460 3470 3500	1808 1816 1824 1832 1840 1848	1809 1817 1825 1833 1841 1849 1857	1810 1818 1826 1834 1842 1850 1858	1811 1819 1827 1835 1843 1851 1859	1812 1820 1828 1836 1844 1852 1860	1813 1821 1829 1837 1845 1853 1853	1814 1822 1830 1838 1846 1854 1862	1815 1823 1831 1839 1847 1855 1863
304 305 306 307 310 311 311 311 311	0 1552 0 1561 0 1561 0 1576 0 1584 0 1592 0 1600 0 1608 0 1616	2 1553 0 1561 1 1569 5 1577 1 1585 2 1593 0 1601 3 1609 5 1617	1554 1562 1570 1578 1586 1586 1594 1602 1610	1555 1563 1571 1579 1587 1595 1603 1611 1619	1556 1564 1572 1580 1588 1596 1604 1612 1620	1557 1565 1573 1581 1589 1597 1605 1613 1621	1558 1566 1574 1582 1590 1598 1606 1614 1622	1559 1567 1575 1583 1591 1599 1607 1615 1623	3420 3430 3440 3450 3460 3470 3500 3510 3520	1808 1816 1824 1832 1840 1848 1856 1864 1872	1809 1817 1825 1833 1841 1849 1857 1865 1873	1810 1818 1826 1834 1842 1850 1858 1866 1874	1811 1819 1827 1835 1843 1851 1859 1867 1875	1812 1820 1828 1836 1844 1852 1860 1868 1876	1813 1821 1829 1837 1845 1853 1861 1869 1877	1814 1822 1830 1838 1846 1854 1862 1870 1878	1815 1823 1831 1839 1847 1855 1863 1871 1879
304 305 306 307 310 311 311 311 311 311	0 1552 0 1567 0 1568 0 1578 0 1584 0 1592 0 1608 0 1616 0 1624	2 1553 1561 1569 1569 1569 1585 1593 1601 1601 1609 1617 1625	1554 1562 1570 1578 1586 1594 1602 1610 1618 1626	1555 1563 1571 1579 1587 1587 1595 1603 1611 1619 1627	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631	3420 3430 3440 3450 3460 3470 3500 3510 3520 3530	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846	1815 1823 1831 1839 1847 1855 1863 1871 1879 1887
304 305 304 304 304 304 314 315 315 314 315 314 315	0 1552 0 1567 0 1567 0 1584 0 1577 0 1584 0 1592 0 1608 0 1668 0 1616 0 1624 9 1632 0 1640	2 1553 1561 1569 1569 1577 1585 1593 1601 1601 1603 1617 1625 1633 1641	 1554 1562 1570 1578 1586 1594 1602 1610 1618 1626 1634 1642 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647	3420 3430 3440 3450 3460 3470 3510 3510 3520 3530 3540 3550	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902	1815 1823 1831 1839 1847 1855 1863 1871 1879 1887 1895 1903
304 305 304 306 310 311 311 311 311 311 311 311 311 311	0 1552 0 1567 0 1564 0 1576 0 1584 0 1592 0 1600 0 1608 0 1616 0 1624 9 1632	2 1553 1561 1569 1569 1585 1585 1585 1593 1601 1601 1603 1617 1625 1633 1641 1649	 1554 1562 1570 1578 1586 1594 1602 1610 1618 1626 1634 1642 1650 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645 1653	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655	3420 3430 3440 3450 3460 3470 3510 3510 3520 3530 3540 3550 3550	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896 1904	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906	1811 1019 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900 1908	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902 1910	1815 1823 1831 1839 1847 1855 1863 1871 1879 1887 1895 1903 1911
304 305 304 307 314 311 313 314 315 314 315 314 315 314 315	0 1552 0 1561 0 1564 0 1564 0 1564 0 1562 0 1608 0 1616 0 1624 9 1632 0 1646 0 1656	2 1553 1561 1569 1569 1585 1585 1585 1593 1601 1601 1609 1617 1625 1633 1641 1641 1645 1657	1554 1562 1570 1578 1586 1594 1602 1610 1618 1626 1634 1634 1634 1658	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1643 1643 1651 1659	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645 1653 1661	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663	3420 3430 3440 3450 3460 3470 3510 3510 3520 3530 3540 3550 3550 3550 3550	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896 1904 1912	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905 1913	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906 1914	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1899 1807 1915	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900 1908 1916	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 18902 1910 1918	1815 1823 1831 1839 1847 1855 1863 1871 1879 1887 1895 1903 1911 1919
304 303 304 304 305 314 315 314 315 314 315 316 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 316 317 317 317 317 317 317 317 317 317 317	0 1552 0 1567 0 1567 0 1584 0 1592 0 1600 0 1608 0 1616 0 1624 9 1632 0 1640	2 1553 1561 1569 1569 1585 1585 1585 1585 1601 1601 1601 1603 1617 1625 1633 1641 1649 1657 1655	 1554 1562 1570 1578 1586 1594 1602 1610 1618 1626 1634 1642 1658 1666 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645 1653 1661	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1670	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663	3420 3430 3440 3450 3460 3470 3510 3510 3520 3530 3540 3550 3550 3560 3570	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896 1904 1912	1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1889 1897 1905 1913 1921	1810 1818 1826 1834 1842 1850 1858 1858 1858 1856 1874 1898 1898 1906 1914	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1899 1907 1915 1923	1812 1820 1828 1836 1844 1852 1860 1868 1876 1868 1876 1884 1892 1900 1908 1916	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917	1814 1822 1830 1838 1846 1854 1862 1878 1846 1878 1846 1894 1902 1910 1918	1815 1823 1831 1839 1847 1855 1863 1871 1879 1887 1895 1903 1911 1919 1927
304 303 304 304 305 310 311 311 311 311 311 311 311 311 311	0 1552 0 1567 0 1567 0 1584 0 1578 0 1584 0 1608 0 1608 0 1616 0 1624 9 1632 0 1648 0 1656 0 1664 0 1672 0 1580	2 1553 1 1561 1 1569 1 1569 1 1585 1 1585 1 1585 1 1593 1 1601 1 1601 1 1601 1 1601 1 1603 1 1649 1 1649 1 1649 1 1649 1 1645 1 1665 1 1665 1 1665 1 16681 1 1681	 1554 1562 1570 1578 1586 1594 1602 1610 1618 1626 1642 1650 1658 1666 1674 1682 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659 1667 1675 1683	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1684	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645 1653 1661 1669 1677 1685	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1654 1662 1670 1678 1686	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687	3420 3430 3440 3450 3460 3500 3510 3520 3530 3540 3550 3560 3550 3560 3570 3600 3610 3620	1808 1816 1824 1832 1840 1848 1856 1864 1856 1864 1880 1888 1896 1904 1912 1920 1928 1936	1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1889 1897 1905 1913 1921 1929 1937	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906 1914 1922 1930 1938	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1939	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900 1908 1916 1924 1932 1940	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902 1910 1918 1926 1934 1942	1815 1823 1831 1839 1847 1855 1863 1871 1875 1895 1903 1911 1919 1927 1935 1943
304 305 304 304 304 304 304 304 314 314 314 314 314 314 314 314 314 31	0 1552 0 1561 0 1574 0 1574 0 1584 0 1592 0 1600 0 1616 0 1616 0 1624 9 1632 0 1640 0 1656 0 1656 0 1664 0 1672 0 1688 0 1688	2 1553 1 1561 1 1569 1 1569 1 1569 1 1585 2 1593 1 1601 3 1609 1 1625 2 1633 1 1649 5 1657 1 1655 1 1673 1 1673 1 1685 1 1673 1 1685 1 1673 1 1689 1 1673 1 1689 1 1673 1 1689 1 1673 1 1689 1 1673 1 1689 1 1673 1 1689 1 1689 1 1673 1 1689 1 1689 1 1673 1 1689 1 1689 1 1689 1 1673 1 1689 1 1689	 1554 1562 1570 1578 1586 1594 1602 1610 1618 1626 1634 1642 1650 1658 1666 1674 1682 1690 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659 1667 1675 1683 1691	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1688	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1645 1653 1661 1669 1677 1685 1693	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1670 1678 1686	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687	3420 3430 3440 3450 3460 3500 3510 3520 3540 3540 3540 3540 3560 3570 3600 3610 3620 3630	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896 1904 1912 1920 1928 1936	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905 1913 1921 1921 1921 1937 1945	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906 1914 1922 1930 1938 1946	1811 1819 1827 1835 1843 1851 1859 1867 1875 1875 1875 1899 1907 1915 1923 1931 1939 1947	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1890 1908 1916 1924 1932 1948	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941	1814 1822 1830 1838 1846 1854 1862 1870 1878 1866 1894 1902 1910 1918 1926 1934 1945	1815 1823 1839 1847 1855 1863 1871 1879 1877 1895 1903 1911 1919 1927 1927 1935 1943 1951
304 305 304 306 316 311 311 311 311 311 311 311 311 31	0 1552 0 1561 0 1564 0 1574 0 1584 0 1584 0 1592 0 1608 0 16168 0 16169 0 1624 9 1632 0 1648 0 1656 0 1664 0 1672 0 1680 0 1680 0 1680 0 1680 0 1690 0 1690	2 1553 1 1561 1 1569 5 1577 5 1577 5 1577 5 1577 1 1601 3 1609 5 1617 1 1625 2 1633 0 1641 3 1649 5 1657 1 1655 1 1655 1 681 5 1681 5 1693 1 681 5 1693 1 695 5 1673 1 681 5 1693 1 695 5 1673 1 681 5 1693 1 695 5 1673 1 705 5 1673 1 705 5 1705 5 1705	 1554 1562 1570 1578 1586 1584 1602 1610 1610 1634 1626 1634 1658 1666 1674 1682 1690 <li< th=""><th>1555 1563 1571 1579 1587 1595 1603 1611 1611 1657 1635 1643 1651 1659 1667 1675 1683 1691 1699 1707</th><th>1556 1564 1572 1580 1588 1596 1604 1612 1628 1636 1644 1652 1660 1668 1676 1684 1676 1684 1692 1700</th><th>1557 1565 1573 1581 1589 1597 1605 1613 1629 1637 1645 1653 1661 1669 1677 1685 1693 1701</th><th>1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1670 1678 1686 1694 1702 1710</th><th>1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687 1695 1703</th><th>3420 3430 3450 3450 3450 3510 3520 3530 3540 3550 3550 3560 3570 3600 3610 3620 3630 3640 3630 3640 3650</th><th>1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896 1904 1912 1920 1928 1936 1944 1952 1960</th><th>1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905 1913 1921 1929 1937 1945 1951</th><th>1810 1818 1826 1834 1842 1850 1858 1858 1858 1858 1878 1878 1906 1914 1922 1930 1938 1946 1954</th><th>1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1939 1947 1955 1963</th><th>1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900 1908 1916 1924 1932 1940 1948 1956</th><th>1813 1821 1829 1837 1845 1853 1861 1869 1875 1893 1901 1909 1917 1925 1933 1941 1949 1957 1965</th><th>1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902 1910 1918 1926 1934 1942 1950 1956</th><th>1815 1823 1839 1847 1855 1863 1879 1879 1887 1895 1903 1911 1919 1927 1935 1943 1959 1967</th></li<>	1555 1563 1571 1579 1587 1595 1603 1611 1611 1657 1635 1643 1651 1659 1667 1675 1683 1691 1699 1707	1556 1564 1572 1580 1588 1596 1604 1612 1628 1636 1644 1652 1660 1668 1676 1684 1676 1684 1692 1700	1557 1565 1573 1581 1589 1597 1605 1613 1629 1637 1645 1653 1661 1669 1677 1685 1693 1701	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1670 1678 1686 1694 1702 1710	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687 1695 1703	3420 3430 3450 3450 3450 3510 3520 3530 3540 3550 3550 3560 3570 3600 3610 3620 3630 3640 3630 3640 3650	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1888 1896 1904 1912 1920 1928 1936 1944 1952 1960	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905 1913 1921 1929 1937 1945 1951	1810 1818 1826 1834 1842 1850 1858 1858 1858 1858 1878 1878 1906 1914 1922 1930 1938 1946 1954	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1939 1947 1955 1963	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900 1908 1916 1924 1932 1940 1948 1956	1813 1821 1829 1837 1845 1853 1861 1869 1875 1893 1901 1909 1917 1925 1933 1941 1949 1957 1965	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902 1910 1918 1926 1934 1942 1950 1956	1815 1823 1839 1847 1855 1863 1879 1879 1887 1895 1903 1911 1919 1927 1935 1943 1959 1967
304 303 304 304 310 311 311 311 311 314 315 314 317 317 317 317 317 317 317 317 317 317	0 1552 0 1561 0 1564 0 1574 0 1574 0 1584 0 1592 0 1600 0 1600 0 1600 0 1624 0 1624 0 1632 0 1646 0 1656 0 1664 0 1672 0 1680 0 1680 0 1688 0 1696 0 1712	2 1553 3 1561 3 1565 3 1577 4 1585 2 1593 5 1577 4 1585 2 1593 5 1673 5 1673 5 1641 3 1641 3 1645 5 1675 1 1681 4 1689 5 1673 1 1681 4 1687 5 1713	1554 1562 1578 1586 1578 1586 1594 1602 1610 1618 1626 1634 1642 1658 1666 1674 1682 1690 1698 1706 1714	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659 1667 1675 1683 1691 1699 1707 1715	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1684 1692 1700 1708	1557 1565 1573 1581 1597 1605 1613 1621 1629 1637 1645 1653 1661 1669 1677 1685 1693 1701 1709	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1670 1678 1686 1694 1702 1710	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687 1695 1703 1711	3420 3430 3440 3450 3460 3470 3510 3520 3530 3540 3550 3560 3570 3600 3610 3620 3630 3640 3650 3660	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1896 1904 1912 1920 1928 1936 1944 1952 1968	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905 1913 1921 1929 1937 1945 1953 1969	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1914 1922 1930 1938 1946 1954 1952	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1939 1947 1955 1951 1971	1812 1820 1828 1836 1844 1852 1860 1868 1876 1884 1892 1900 1908 1916 1924 1932 1940 1948 1956 1956 1956	1813 1821 1821 1827 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1957 1973	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1878 1846 1894 1900 1918 1926 1934 1942 1950 1958 1958 1956 1974	1815 1823 1839 1847 1855 1863 1871 1879 1871 1879 1887 1995 1903 1911 1919 1927 1935 1943 1954 1959 1967 1975
304 303 304 304 304 314 311 314 314 314 314 314 314 314 31	0 1552 0 1561 0 1574 0 1574 0 1574 0 1574 0 1600 0 1600 0 1600 0 1616 0 1640 0 1656 0 1667 0 1667 0 1667 0 1668 0 1672 0 1680 0 1696 0 1704 0 1712 0 1720	2 1553 1 1561 1 1561 1 1565 1 1565 2 1575 1 1585 2 1593 1 1609 3 1609 3 1609 3 1641 3 1641 3 1645 1 1645 1 1673 9 1665 1 1673 9 1689 1 1721 1 721	 1554 1562 1578 1602 1610 1610<td>1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1659 1667 1675 1663 1691 1699 1707 1715 1723</td><td>1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1688 1676 1682 1700 1708 1716 1724</td><td>1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645 1653 1651 1669 1677 1685 1693 1701 1709 1717 1725</td><td>1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1662 1662 1662 1662 1702 1710 1718 1726</td><td>1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1633 1647 1655 1663 1663 1667 1667 1665 1703 1711 1719 1727</td><td>3420 3430 3440 3450 3460 3510 3510 3520 3530 3540 3550 3560 3560 3660 3650 3660 3650 3660 3670</td><td>1808 1816 1824 1832 1840 1848 1856 1864 1856 1868 1896 1904 1912 1920 1928 1936 1944 1952 1960 1968 1976</td><td>1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1889 1897 1901 1921 1921 1923 1945 1953 1961 1969 1977</td><td>1810 1818 1826 1834 1834 1850 1858 1866 1874 1866 1874 1882 1890 1898 1906 1914 1922 1930 1946 1954 1946 1970 1978</td><td>1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1939 1947 1955 1963 1971 1979</td><td>1812 1820 1828 1844 1852 1860 1868 1868 1868 1876 1884 1892 1900 1908 1916 1916 1948 1946 1948 1946 1948</td><td>1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1965 1973 1981</td><td>1814 1822 1830 1838 1846 1854 1862 1870 1878 1886 1894 1902 1910 1918 1926 1934 1942 1950 1958 1966 1974</td><td>1815 1823 1839 1847 1855 1863 1871 1879 1887 1895 1903 1911 1919 1927 1943 1951 1959 1967 1975 1983</td>	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1659 1667 1675 1663 1691 1699 1707 1715 1723	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1688 1676 1682 1700 1708 1716 1724	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1637 1645 1653 1651 1669 1677 1685 1693 1701 1709 1717 1725	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1662 1662 1662 1662 1702 1710 1718 1726	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1633 1647 1655 1663 1663 1667 1667 1665 1703 1711 1719 1727	3420 3430 3440 3450 3460 3510 3510 3520 3530 3540 3550 3560 3560 3660 3650 3660 3650 3660 3670	1808 1816 1824 1832 1840 1848 1856 1864 1856 1868 1896 1904 1912 1920 1928 1936 1944 1952 1960 1968 1976	1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1889 1897 1901 1921 1921 1923 1945 1953 1961 1969 1977	1810 1818 1826 1834 1834 1850 1858 1866 1874 1866 1874 1882 1890 1898 1906 1914 1922 1930 1946 1954 1946 1970 1978	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1939 1947 1955 1963 1971 1979	1812 1820 1828 1844 1852 1860 1868 1868 1868 1876 1884 1892 1900 1908 1916 1916 1948 1946 1948 1946 1948	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1965 1973 1981	1814 1822 1830 1838 1846 1854 1862 1870 1878 1886 1894 1902 1910 1918 1926 1934 1942 1950 1958 1966 1974	1815 1823 1839 1847 1855 1863 1871 1879 1887 1895 1903 1911 1919 1927 1943 1951 1959 1967 1975 1983
304 303 304 304 307 314 314 314 314 314 314 314 314 314 314	0 1552 0 1561 0 1574 0 1574 0 1574 0 1574 0 1574 0 1606 0 1616 0 1616 0 1624 9 1632 0 1640 0 1656 0 1656 0 1656 0 1688 0 1688 0 1696 0 1712 0 1728	2 1553 3 1561 3 1567 4 1585 2 1593 3 1601 3 1609 5 1617 4 1683 3 1641 3 1649 5 1657 4 1685 4 1689 4 1689 4 1697 5 1713 9 1721 1729	 1554 1554 1562 1578 1586 1578 1586 1594 1602 1610 1610 1618 1626 1634 1642 1650 1658 1666 1674 1682 1698 1706 1714 1722 1730 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659 1667 1675 1663 1691 1707 1715 1723	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1684 1692 1700 1708 1716 1724	1557 1565 1573 1581 1589 1605 1613 1621 1629 1637 1645 1653 1661 1669 1677 1685 1693 1701 1709 1717 1725	1558 1566 1574 1582 1590 1690 1614 1622 1630 1638 1646 1634 1662 1670 1678 1684 1684 1684 1684 1684 1702 1710 1718 1726	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687 1695 1703 1711 1719 1727	3420 3430 3440 3450 3450 3510 3510 3520 3530 3540 3550 3550 3570 3600 3610 3610 3620 3640 3650 3660 3650 3660 3670	1808 1816 1824 1832 1840 1848 1856 1864 1856 1868 1896 1904 1912 1920 1928 1936 1944 1952 1960 1968 1976 1984	1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1889 1897 1905 1913 1921 1929 1935 1961 1969 1977 1985	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906 1914 1922 1930 1946 1944 1945 1946	1811 1819 1827 1835 1843 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1931 1935 1963 1971 1979 1987	1812 1820 1828 1836 1844 1852 1860 1868 1868 1876 1868 1916 1908 1916 1948 1948 1956 1964 1972 1980	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1945 1949	1814 1822 1830 1838 1846 1854 1862 1870 1878 1878 1878 1910 1918 1926 1938 1942 1950 1958 1956 1974 1982	1815 1823 1839 1847 1855 1863 1871 1879 1877 1879 1887 1895 1903 1911 1919 1927 1935 1951 1955 1955 1967 1975 1983 1991
304 303 304 304 305 314 311 311 311 311 314 314 314 314 314	0 1552 0 1561 0 1564 0 1574 0 1574 0 1584 0 1592 0 1600 0 1608 0 1616 0 1624 0 1632 0 1664 0 1656 0 1664 0 1656 0 1664 0 1672 0 1664 0 1672 0 1666 0 1778 0 1728 0 1728 0 1728	2 1553 1 1561 1 1561 1 1565 1 1565 1 1585 2 1593 1 1601 1 1609 5 1617 1 1625 2 1633 1 1641 1 1641 1 1641 1 1641 1 1645 1 1657 1 1657 1 1729 1 729 1 727 1 745	 1554 1562 1578 1578 1586 1594 1602 1618 1626 1634 1626 1634 1658 1666 1674 1698 1714 1722 1730 1736 1746 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1657 1643 1651 1659 1667 1675 1683 1691 1679 1707 1715 1723	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1684 1692 1700 1708 1716 1724 1732 1740 1748	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1633 1653 1653 1661 16693 1701 1709 1717 1725 1733 1741	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1644 1662 1650 1658 1644 1702 1710 1718 1726 1774 1726	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1635 1663 1663 1671 1655 1663 1701 1719 1727 1735 1743	3420 3430 3440 3450 3460 3510 3520 3530 3540 3550 3560 3550 3660 3660 3660 3660 366	1808 1816 1824 1832 1840 1848 1856 1864 1872 1880 1804 1904 1912 1920 1928 1936 1944 1952 1960 1968 1976 1984 1976	1809 1817 1825 1833 1841 1849 1857 1865 1873 1865 1873 1881 1889 1897 1913 1921 1929 1937 1945 1953 1961 1969 1977 1985 1993 2001	1810 1818 1826 1834 1842 1850 1858 1866 1874 1860 1914 1906 1914 1922 1930 1938 1946 1954 1954 1970 1978	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1939 1947 1955 1951 1979 1987 1995 2003	1812 1820 1828 1836 1844 1852 1860 1868 1868 1868 1868 1908 1908 1916 1924 1932 1940 1948 1956 1964 1972 1980 1988	1813 1821 1827 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1957 1973 1981	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902 1910 1918 1926 1934 1942 1950 1958 1966 1974 1982 2006	1815 1823 1839 1847 1855 1863 1871 1879 1871 1879 1887 1895 1903 1911 1919 1927 1935 1954 1956 1957 1958 1951 1991 1991 1991
304 305 304 306 307 316 317 317 317 317 317 317 317 317 317 317	0 1552 0 1561 0 1574 0 1574 0 1574 0 1584 0 1592 0 1600 0 1616 0 1616 0 1624 9 1632 0 1660 0 1624 9 1632 0 1660 0 1656 0 1664 0 1656 0 1667 0 1664 0 1674 0 1774 0 1728 0 1728	2 1553 1 1561 1 1565 1 1565 1 1565 2 1593 1 1609 1 1609 1 1609 1 1617 1 1625 2 1633 1 1641 1 1711 1 1721 1 1725 1 1745 1 1755 1 1755	 1554 1562 1570 1578 1578 1594 1602 1618 1626 1634 1642 1658 1666 1658 1668 1698 1714 1722 1730 1738 1746 1754 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1659 1667 1663 1669 1707 1715 1723 1723 1731 1739	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1668 1668 1668 1668 1668 1770 1708 1716 1724 1732 1740 1745	1557 1565 1573 1589 1589 1597 1605 1613 1621 1629 1637 1645 1653 1661 1669 1677 1685 1693 1701 1709 1717 1725 1743 1741	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1638 1646 1658 1658 1658 1658 1658 1659 1718 1726 1718 1726	1559 1567 1575 1503 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1661 1671 1687 1695 1703 1711 1719 1727 1743 1751	3420 3430 3440 3450 3460 3510 3510 3520 3530 3540 3550 3560 3560 3660 3660 3660 3660 3670 3710 3710 3710	1808 1816 1816 1824 1832 1840 1848 1856 1864 1856 1868 1896 1904 1912 1920 1928 1944 1952 1960 1968 1976 1984 1992 2000 2008	1809 1817 1825 1833 1841 1849 1857 1865 1873 1881 1889 1897 1905 1913 1921 1921 1953 1961 1969 1977 1985 1993 2001	1810 1818 1826 1834 1842 1850 1858 1866 1874 1866 1874 1880 1914 1922 1930 1938 1946 1954 1954 1978 1986 1994 2002	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1947 1955 1963 1971 1979 1987 1995 2003 2011	1812 1820 1828 1836 1844 1852 1860 1868 1868 1868 1868 1884 1876 1900 1908 1916 1948 1956 1964 1972 1980 1988 1996 2004	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1907 1925 1933 1941 1949 1957 1957 1958 1973 1981	1814 1822 1830 1838 1846 1854 1862 1870 1878 1846 1894 1902 1918 1926 1934 1942 1950 1958 1965 1974 1982 1990 1998	1815 1823 1839 1847 1855 1863 1871 1879 1877 1879 1887 1895 1901 1919 1927 1943 1951 1955 1967 1975 1983 1991 1999 2007 2015
304 303 304 304 304 304 304 304 304 314 314 314 314 314 314 314 314 314 31	0 1552 0 1561 0 1574 0 1574 0 1574 0 1574 0 1574 0 1606 0 1608 0 1616 0 1624 9 1632 0 1640 0 1656 0 1656 0 1664 0 1656 0 1666 0 1672 0 1688 0 1688 0 1688 0 1696 0 1774 0 1728 0 1728 0 1744 0 1752 0 1768	2 1553 3 1561 3 1567 4 1585 2 1593 3 1601 3 1609 3 1609 3 1609 4 1633 3 1641 3 1649 5 1657 4 1649 5 1657 4 1655 1 1657 5 1673 1 1705 1 1721 1 729 1 737 1 745 1 769 1 7	 1554 1554 1562 1578 1578 1578 1580 1594 1602 1610 1610 1610 1614 1626 1634 1650 1658 1666 1674 1698 1706 1714 1722 1730 1738 1746 1754 1752 1770 	1555 1563 1571 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659 1667 1675 1663 1669 1707 1715 1723 1771 1739 1747 1755 1763	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1644 1652 1660 1668 1676 1668 1676 1676 1708 1716 1724 1732 1740 1748 1756 1764 1772	1557 1565 1573 1581 1589 1605 1613 1621 1629 1637 1645 1653 1661 1669 1677 1685 1693 1701 1709 1717 1725	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1638 1646 1634 1662 1670 1678 1684 1684 1684 1684 1684 1702 1710 1718 1726	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1639 1647 1655 1663 1671 1679 1687 1703 1711 1719 1727 1735 1743 1751 1759	3420 3430 3440 3450 3450 3510 3520 3530 3540 3550 3550 3550 3570 3600 3610 3620 3640 3650 3660 3650 3660 3670 3710 3710 3710 3730 3730	1808 1816 1816 1824 1830 1848 1856 1864 1856 1868 1896 1904 1912 1920 1928 1936 1944 1952 1960 1968 1976 1984 1992 2000 2008 2016	1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1869 1897 1913 1921 1929 1937 1945 1953 1961 1969 1977 1985 1993 2001 2009 2017 2025	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906 1914 1922 1930 1941 1944 1942 1970 1978 1946 1994 2002 2018 2002	1811 1819 1827 1835 1843 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1931 1935 1963 1971 1979 1987 1987 1995 2003 2011 2019 2027	1812 1820 1828 1836 1844 1852 1860 1868 1868 1868 1876 1884 1892 1900 1908 1916 1948 1948 1956 1948 1956 1948 1956 2004 2012 2028	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1955 1973 1981 1989 1997 2005 2013 2021	1814 1822 1830 1838 1846 1854 1862 1870 1878 1878 1878 1878 1910 1918 1926 1934 1942 1950 1958 1966 1974 1942 1950 1958 2006 2014 2022 2039	1815 1823 1839 1847 1855 1863 1871 1879 1877 1879 1877 1895 1903 1911 1919 1927 1935 1951 1955 1967 1975 1983 1991 1999 2007 2015 2031
304 303 304 304 305 314 311 311 311 311 311 311 311 311 311	0 1552 0 1561 0 1574 0 1574 0 1584 0 1592 0 1600 0 1608 0 1608 0 1624 9 1632 0 1666 0 1624 9 1632 0 1666 0 1656 0 1666 0 1666 0 1686 0 1686 0 1686 0 1686 0 1686 0 1704 0 1772 0 1728 0 1736 0 1744 0 1750	2 1553 3 1561 3 1565 3 1577 4 1585 2 1593 3 1601 3 1609 5 1617 4 1625 2 1633 0 1641 3 1641 3 1641 3 1645 1 1675 1 1729 1 729 1 777 1 729 1 777 1 729 1 729 1 777 1 729 1 729 1 777 1 729 1 729 1 777 1 729 1 729	 1554 1562 1578 1578 1578 1594 1602 1618 1626 1634 1626 1634 1658 1666 1674 1698 1706 1738 1746 1754 1762 1778 	1555 1563 1579 1587 1595 1603 1611 1619 1627 1635 1643 1651 1659 1667 1663 1669 1707 1715 1707 1715 1723 1731 1747 1755	1556 1564 1572 1580 1588 1596 1604 1612 1620 1628 1636 1648 1636 1668 1670 1708 1770 1708 1770 1748 1756 1764 1772 1780	1557 1565 1573 1581 1589 1597 1605 1613 1621 1629 1633 1661 1653 1661 1653 1661 1709 1701 1705 1773 1741 1749 1757 1755	1558 1566 1574 1582 1590 1598 1606 1614 1622 1630 1634 1646 1654 1654 1670 1768 1774 1718 1726 1734 1742 1758 1766	1559 1567 1575 1583 1591 1599 1607 1615 1623 1631 1635 1663 1663 1667 1655 1663 1701 1679 1687 1687 1695 1701 1719 1727 1735 1743 1751 1751	3420 3430 3440 3450 3450 3510 3510 3520 3530 3540 3550 3560 3550 3660 3660 3660 3660 366	1808 1816 1816 1824 1830 1848 1856 1864 1856 1868 1896 1904 1912 1920 1928 1936 1944 1952 1960 1968 1976 1984 1992 2000 2008 2016	1809 1817 1825 1833 1841 1849 1857 1865 1873 1861 1889 1897 1905 1913 1921 1929 1937 1945 1953 1961 1969 1977 1985 1993 2001 2009 2017 2025 2033	1810 1818 1826 1834 1842 1850 1858 1866 1874 1882 1890 1898 1906 1914 1922 1930 1944 1954 1945 1946 1954 1946 2010 2018 2026 2034	1811 1819 1827 1835 1843 1851 1859 1867 1875 1883 1891 1899 1907 1915 1923 1931 1931 1979 1947 1955 2003 2011 2019 2027 2035	1812 1820 1828 1836 1844 1852 1860 1868 1868 1868 1876 1884 1990 1908 1916 1948 1948 1948 1946 1948 1948 1946 2004 2012 2020 2028 2036	1813 1821 1829 1837 1845 1853 1861 1869 1877 1885 1893 1901 1909 1917 1925 1933 1941 1949 1957 1945 1943 1949 1957 1981	1814 1822 1830 1838 1846 1854 1862 1870 1878 1866 1894 1910 1918 1926 1934 1942 1950 1958 1966 1974 1982 2006 2014 2022 2038	1815 1823 1839 1847 1855 1863 1871 1879 1877 1879 1877 1895 1903 1911 1919 1927 1935 1951 1955 1967 1975 1983 1991 1999 2007 2015 2031 2039

E-2

		0	1	2	3	4	5	6	٦	1	0	1	2	3	4	5	6	7
4000 2048	4000	2048	2049	2050	2051	2052	2053	2054	2055	4400	2304	2305	2306	2307	2 3 0 8	2 309	2310	2311
to to	4010									4410								
4777 2559 (Octal) (Decimal)	4030	2072	2073	2074	2075	2076	2077	2078	2079	4430	2328	2329	2330	2331	2332	2333	2334	2335
	4040									4440 4450								
Octal Decimal 10000 - 4096	4060	2096	2097	2098	20 99	2100	2101	2102	2103	4460	2352	2353	2354	2355	2356	2357	2358	2359
20000 · 8192	4070	2104	2105	2106	2107	2108	2109	2110	2111	4470	2360	2361	2362	2363	2364	2365	2366	2367
30000 - 12288 40000 - 16384										4500								
50000 - 20480										4510								
60000 · 24576 70000 · 28672	4130	2136	2137	2138	2139	2140	2141	2142	2143	4530	2392	2393	2394	2395	2396	2397	2398	2 3 9 9
		2144								4540					2404 2412			
	4160	2160	2161	2162	2163	2164	2165	2166	2167	4560	2416	2417	2418	2419	2420	2421	2422	2423
	4170	2168	2169	2170	2171	2172	2173	2174	2175	4570	2424	2425	2420	242(2428	2429	2430	2431
										4600								
										4610 4620								
·.	4230	2200	2201	2202	2203	2204	2205	2206	2207	4630	2456	2457	2458	2459	2460	2461	2462	2463
	4250	2216	2217	2218	2219	2220	2221	2222	2223	4640 4650	2472	2473	2474	2475	2476	2477	2478	2479
										4660 4670								
		2240 2248													2500			
	4320	2256	2257	2258	2259	2260	2261	2262	2263	4720	2512	2513	2514	2515	2516	2517	2518	2519
										4730								
	4350	2280	2281	2282	2283	2284	2285	2286	2287	4750	2536	2537	2538	2539	2540	2541	2542	2543
		2288 2296								4760					2548 2556			
					_				_	and the second second		والمستعدان فالبلغان					_	
		0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
	6000	0	1	2			5				0	1	2	3		5		
5000 2560		2560	2561	2562	2563	2564	2565	256Ġ	2567	5400 5410	2816	2817	2818	2819	2820	2821	2822	2823
to to 5777 3071	5010 5020	2560 2568 2576	2561 2569 2577	2562 2570 2578	2563 2571 2579	2564 2572 2580	2565 2573 2581	2566 2574 2582	2567 2575 2583	5410 5420	2816 2824 2832	2817 2825 2833	2818 2826 2834	2819 2827 2835	2820 2828 2836	2821 2829 2837	2822 2830 2838	2823 2831 2839
to to	5010 5020 5030 5040	2560 2568 2576 2584 2592	2561 2569 2577 2585 2593	2562 2570 2578 2588 2594	2563 2571 2579 2587 2595	2564 2572 2580 2588 2596	2565 2573 2581 2581 2597	2566 2574 2582 2590 2598	2567 2575 2583 2591 2599	5410 5420 5430 5440	2816 2824 2832 2840 2848	2817 2825 2833 2841 2849	2818 2826 2834 2842 2850	2819 2827 2835 2843 2851	2820 2828 2836 2844 2852	2821 2829 2837 2845 2853	2822 2830 2838 2846 2854	2823 2831 2839 2847 2855
to to 5777 3071	5010 5020 5030 5040 5050	2560 2568 2576 2584 2592 2600	2561 2569 2577 2585 2593 2601	2562 2570 2578 2588 2594 2602	2563 2571 2579 2587 2595 2603	2564 2572 2580 2588 2596 2604	2565 2573 2581 2589 2597 2605	2566 2574 2582 2590 2598 2606	2567 2575 2583 2591 2599 2607	5410 5420 5430 5440 5450	2816 2824 2832 2840 2848 2856	2817 2825 2833 2841 2849 2857	2818 2826 2834 2842 2850 2858	2819 2827 2835 2843 2851 2859	2820 2828 2836 2844 2852 2860	2821 2829 2837 2845 2853 2861	2822 2830 2838 2846 2854 2854 2862	2823 2831 2839 2847 2855 2863
to to 5777 3071	5010 5020 5030 5040 5050 5060	2560 2568 2576 2584 2592 2600 2608	2561 2569 2577 2585 2593 2601 2609	2562 2570 2578 2588 2594 2602 2610	2563 2571 2579 2587 2595 2603 2611	2564 2572 2580 2588 2596 2604 2612	2565 2573 2581 2589 2597 2605 2613	2566 2574 2582 2590 2598 2606 2614	2567 2575 2583 2591 2599 2607 2615	5410 5420 5430 5440	2816 2824 2832 2840 2848 2856 2864	2817 2825 2833 2841 2849 2857 2865	2818 2826 2834 2842 2850 2858 2866	2819 2827 2835 2843 2851 2859 2867	2820 2828 2836 2844 2852 2860 2868	2821 2829 2837 2845 2853 2861 2869	2822 2830 2838 2846 2854 2854 2862 2870	2823 2831 2839 2847 2855 2863 2863 2871
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070	2560 2568 2576 2584 2592 2600 2608 2616	2561 2569 2577 2585 2593 2601 2609 2617	2562 2570 2578 2588 2594 2602 2610 2618	2563 2571 2579 2587 2595 2603 2611 2619	2564 2572 2580 2588 2596 2604 2612 2620	2565 2573 2581 2589 2597 2605 2613 2621	2566 2574 2582 2590 2598 2606 2614 2622	2567 2575 2583 2591 2599 2607 2615 2623	5410 5420 5430 5440 5450 5460 5470	2816 2824 2832 2840 2848 2856 2864 2872	2817 2825 2833 2841 2849 2857 2865 2873	2818 2826 2834 2842 2850 2858 2866 2874	2819 2827 2835 2843 2851 2859 2867 2875	2820 2828 2836 2844 2852 2860 2868 2876	2821 2829 2837 2845 2853 2861 2869 2877	2822 2830 2838 2846 2854 2854 2862 2870 2878	2823 2831 2839 2847 2855 2863 2863 2879
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070 5100 5110	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633	2562 2570 2578 2586 2594 2602 2610 2618 2626 2634	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635	2564 2572 2580 2588 2596 2604 2612 2620 2628 2628 2636	2565 2573 2581 2597 2605 2613 2621 2629 2637	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638	2567 2575 2583 2591 2599 2607 2615 2623 2631 2631	5410 5420 5430 5440 5450 5460 5470 5500 5510	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888	2817 2825 2833 2841 2849 2857 2865 2873 2881 2881 2889	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893	2822 2830 2838 2846 2854 2862 2870 2878 2886 2886 2894	2823 2831 2839 2847 2855 2863 2863 2871 2879 2887 2887 2895
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070 5100 5110 5120	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641	2562 2570 2578 2586 2594 2602 2610 2618 2626 2634 2642	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2643	2564 2572 2580 2588 2596 2604 2612 2620 2628 2636 2644	2565 2573 2581 2597 2605 2613 2621 2629 2637 2645	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646	2567 2575 2583 2591 2599 2607 2615 2623 2631 2631 2639 2647	5410 5420 5430 5440 5450 5460 5470 5500 5510 5520	28.16 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901	2822 2830 2838 2846 2854 2862 2870 2878 2886 2894 2902	2823 2831 2839 2847 2855 2863 2879 2887 2895 2903
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070 5100 5110 5120 5130 5140	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657	2562 2570 2578 2586 2594 2602 2610 2618 2626 2634 2642 2650 2658	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2643 2651 2659	2564 2572 2580 2588 2596 2604 2612 2620 2628 2636 2644 2652 2660	2565 2573 2581 2597 2605 2613 2621 2629 2637 2645 2653 2661	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2654 2662	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2663	5410 5420 5430 5440 5450 5460 5470 5500 5510 5520 5530 5540	2P.16 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2904 2912	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2907 2915	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917	2822 2830 2838 2846 2854 2862 2870 2878 2886 2894 2902 2910 2918	2823 2831 2839 2847 2855 2863 2871 2879 2887 2895 2903 2911 2919
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070 5110 5120 5130 5140 5150	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2646 2648	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665	2562 2570 2578 2586 2594 2602 2610 2618 2626 2634 2642 2650 2658 2658 2666	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2643 2651 2659 2667	2564 2572 2580 2588 2596 2604 2612 2620 2628 2636 2644 2652 2660 2668	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2653 2653 2661 2669	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2654 2662 2670	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2663 2671	5410 5420 5430 5440 5450 5460 5470 5500 5510 5520 5530	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924	2821 2829 2837 2845 2853 286L 2869 2877 2885 2893 2901 2909 2917 2925	2822 2830 2838 2846 2854 2862 2870 2878 2886 2894 2902 2910 2918 2926	2823 2831 2839 2847 2855 2863 2871 2879 2887 2895 2903 2911 2919 2927
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070 5110 5120 5130 5140 5150 5160	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2672	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665 2673	2562 2570 2578 2586 2594 2602 2610 2618 2626 2634 2642 2642 2650 2658 2666 2674	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2643 2651 2659 2667 2675	2564 2572 2580 2588 2596 2604 2612 2620 2628 2636 2644 2652 2644 2652 2660 2668 2676	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2645 2653 2661 2669 2677	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2662 2654 2662 2670 2678	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2663 2671 2679	5410 5420 5430 5440 5450 5460 5470 5500 5510 5520 5530 5540 5550	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932	2821 2829 2837 2845 2853 286L 2869 2877 2885 2893 2901 2909 2917 2925 2933	2822 2830 2838 2846 2854 2862 2870 2878 2886 2894 2902 2910 2918 2926 2934	2823 2831 2839 2847 2855 2863 2671 2879 2887 2895 2901 2911 2919 2927 2935
to to 5777 3071	5010 5020 5030 5040 5060 5070 5110 5120 5130 5140 5140 5160 5170 5200	2560 2568 2576 2592 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2656 2664 2656 2680 2688	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665 2673 2681 2689	2562 2570 2578 2598 2594 2602 2610 2618 2626 2634 2642 2658 2658 2658 2666 2674 2682 2690	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2643 2651 2659 2667 2675 2683 2691	2564 2572 2580 2588 2596 2604 2612 2620 2628 2636 2644 2652 2660 2668 2676 2684 2672	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2661 2669 2677 2685 2693	2566 2574 2582 2598 2606 2614 2622 2630 2638 2646 2654 2662 2670 2678 2686 2686 2694	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2663 2671 2679 2687 2695	5410 5420 5430 5440 5450 5460 5470 5500 5510 5520 5530 5520 5530 5550 5550 5550 555	2816 2824 2832 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2920 2928 2936 2936	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945	2818 2826 2834 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2938 2938 2946	2819 2827 2835 2845 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2931	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917 2925 2933 2941 2949	2822 2830 2838 2846 2854 2862 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950	2823 2831 2839 2847 2855 2863 2879 2887 2895 2903 2911 2919 2927 2935 2943 2951
to to 5777 3071	5010 5020 5030 5040 5060 5060 5070 5110 5120 5130 5140 5140 5160 5170 5200 5210	2560 2568 2576 2592 2592 2600 2608 2616 2624 2640 2640 2648 2656 2664 2656 2664 2672 2680 2688 2696	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665 2673 2681 2689 2697	2562 2570 2578 2594 2602 2610 2618 2626 2634 2642 2642 2642 2650 2658 2666 2674 2682 2690 2698	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2643 2651 2659 2667 2659 2667 2675 2683 2691 2699	2564 2572 2580 2586 2596 2604 2612 2620 2628 2636 2644 2652 2644 2652 2660 2648 2676 2684 2692 2700	2565 2573 2581 2597 2605 2613 2621 2629 2637 2645 2653 2661 2669 2677 2685 2693 2701	2566 2574 2582 2598 2698 2698 2614 2622 2630 2638 2646 2654 2646 2654 2662 2670 2678 2686 2694 2702	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2663 2647 2655 2663 2671 2679 2687 2695 2703	5410 5420 5430 5440 5460 5460 5470 5500 5510 5520 5530 5520 5530 5550 5550 5550 555	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2936	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945 2953	2818 2826 2834 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930 2938 2946 2954	2819 2827 2835 2845 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2947 2955	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2916 2924 2932 2940 2948 2956	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917 2925 2933 2941 2949 2957	2822 2830 2838 2846 2854 2854 2854 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950 2958	2823 2831 2839 2847 2855 2863 2871 2895 2903 2911 2919 2927 2935 2943 2951 2959
to to 5777 3071	5010 5020 5030 5040 5050 5060 5070 5100 5110 5130 5140 5130 5140 5130 5140 5130 5120 5210 5210 5220 5230	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2656 2664 2672 2680 2688 2696 2704 2712	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665 2665 2665 2673 2681 2689 2697 2705 2705	2562 2570 2578 2594 2602 2610 2618 2626 2634 2642 2650 2658 2666 2674 2682 2690 2698 2706 2714	2563 2571 2579 2603 2611 2619 2627 2635 2643 2651 2659 2667 2667 2667 2667 2669 2667 2669 2667 2669 2707 2715	2564 2572 2580 2596 2604 2612 2628 2636 2644 2652 2668 2668 2668 2668 2668 2668 2668	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2669 2677 2685 2693 2701 2709 2717	2566 2574 2582 2590 2614 2622 2630 2638 2646 2654 2662 2670 2678 2686 2694 2694 2694 2710 2718	2567 2575 2583 2599 2607 2615 2623 2631 2639 2647 2655 2663 2663 2663 2667 2669 2687 2695 2703 2711 2719	5410 5420 5430 5450 5460 5470 5500 5510 5520 5530 5530 5530 5550 5550 5550 555	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2936 2944	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2915 2929 2937 2929 2937 2945 2953 2961 2969	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2916 2922 2930 2938 2946 2954 2954	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2947 2955 2955 2955 2955	2820 2828 2834 2852 2860 2868 2876 2884 2892 2900 2908 2916 2916 2924 2932 2940 2948 2956 2954 2954	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917 2925 2933 2941 2949 2957 2957 2957 2965 2973	2822 2830 2838 2854 2854 2854 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2934 2950 2958 2956 2956 2956	2823 2831 2839 2847 2855 2863 2671 2879 2887 2895 2903 2911 2919 2927 2935 2943 2951 2959 2957 2955
to to 5777 3071	5010 5020 5040 5050 5060 5070 5100 5110 5120 5130 5140 5150 5160 5170 5200 5220 5220 5220 5220	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2648 2656 2656 2656 2656 2688 2696 2704 2712 2720	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665 2673 2681 2689 2697 2705 2713 2721	2562 2570 2578 2586 2594 2602 2610 2618 2626 2634 2642 2650 2658 2664 2674 2682 2690 2698 2706 2714 2722	2563 2571 2577 2587 2595 2603 2611 2619 2627 2635 2643 2651 2657 2667 2667 2667 2667 2667 2667 2667	2564 2572 2586 2596 2604 2612 2620 2628 2644 2652 2668 2668 2668 2668 2668 2668 2668	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2645 2653 2677 2685 2693 2701 2709 2717 2725	2566 2574 2582 2598 2606 2614 2622 2630 2638 2646 2654 2662 2670 2678 2666 2674 2702 2710 2718 2718	2567 2575 2583 2599 2607 2615 2623 2639 2647 2655 2663 2671 2679 2687 2695 2703 2711 2719 2711 2719	5410 5420 5430 5450 5460 5470 5500 5510 5520 5530 5540 5550 5550 5570 5560 5570 5600 5610 5620 5630 5640	28.16 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2944 2952 2958 2952 2968 2976	2817 2825 2833 2849 2857 2865 2873 2881 2889 2897 2905 2913 2929 2937 2929 2937 2945 2953 2969 2969 2969	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 29906 2914 2922 2930 2938 2946 2954 2954 2970 2978	2819 2827 2835 2851 2859 2867 2875 2883 2891 2899 2907 2915 29207 2915 2931 2931 2939 2931 2939 2947 2955 2957 2979	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2940 2948 2956 2954 2954	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917 2929 2933 2941 2949 2957 2965 2973 2981	2822 2830 2838 2854 2854 2854 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950 2958 2956 2974 2982	2823 2831 2839 2847 2855 2863 2671 2879 2887 2895 2903 2911 2919 2927 2935 2943 2951 2951 2957 2957 2953
to to 5777 3071	5010 5020 5040 5050 5060 5070 5110 5120 5130 5140 5150 5160 5170 5220 5210 5220 5220 5220 5220 5220 522	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2656 2664 2656 2680 2688 2696 2704 2712 2728 2736	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2657 2657 2655 2673 2681 2689 2697 2705 2713 2729 2737	2562 2578 2578 2578 2594 2602 2618 2626 2634 2642 2658 2666 2658 2666 2658 2666 2658 2690 2698 2706 2714 2730 2738	2563 2571 2579 2687 2695 2603 2611 2619 2627 2643 2659 2667 2669 2667 2669 2669 2669 2707 2715 2723 2731 2731 2731	2564 2572 2580 2596 2604 2612 2620 2628 2644 2650 2668 2660 2668 2666 2668 2666 2684 2692 2700 2708 2702 2702 2772 2772 2772	2565 2573 2581 2597 2605 2613 2629 2637 2645 2669 2667 2669 2677 2685 2661 2669 2677 2685 2693 2701 2709 2717 2725 2733 2741	25566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2662 2662 2662 2662 2662 2662	2567 2575 2583 2591 2599 2607 2615 2623 2647 2655 2663 2663 2663 2667 2687 2687 2687 2687 2687 2687 2687	5410 5420 5430 5440 5450 5460 5470 5510 5510 5520 5530 5550 5550 5550 5550 5550 555	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2912 2920 2928 2916 2912 2920 2928 2936 2944 2952 2960 2968 2976 2984 2992	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945 2953 2961 2965 2961 2965 2985 2993	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930 2914 2923 2938 2946 2954 2954 2970 2986 2994	2819 2827 2835 2835 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2931 2939 2947 2955 2963 2971 2995	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948 2956 2956 2964 2972 2988 2996	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2901 2901 2901 2917 2925 2933 2941 2957 2957 2957 2965 2973 2989 2997	2822 2830 2838 2846 2854 2854 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950 2958 2958 2958 2958 2958 2958	2823 2831 2839 2847 2855 2863 2871 2895 2903 2911 2919 2927 2935 2943 2951 2959 2957 2959 2967 2975 2983 2991 2999
to to 5777 3071	5010 5020 5040 5050 5060 5070 5110 5120 5130 5140 5150 5160 5170 5220 5210 5220 5220 5220 5220 5220 522	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2656 2664 2656 2680 2688 2696 2704 2712 2728 2736	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2657 2657 2655 2673 2681 2689 2697 2705 2713 2729 2737	2562 2578 2578 2578 2594 2602 2618 2626 2634 2642 2658 2666 2658 2666 2658 2666 2658 2690 2698 2706 2714 2730 2738	2563 2571 2579 2687 2695 2603 2611 2619 2627 2643 2659 2667 2669 2667 2669 2669 2669 2707 2715 2723 2731 2731 2731	2564 2572 2580 2596 2604 2612 2620 2628 2644 2650 2668 2660 2668 2666 2668 2666 2684 2692 2700 2708 2702 2702 2772 2772 2772	2565 2573 2581 2597 2605 2613 2629 2637 2645 2669 2667 2669 2677 2685 2661 2669 2677 2685 2693 2701 2709 2717 2725 2733 2741	25566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2662 2662 2662 2662 2662 2662	2567 2575 2583 2591 2599 2607 2615 2623 2647 2655 2663 2663 2663 2667 2687 2687 2687 2687 2687 2687 2687	5410 5420 5430 5440 5450 5460 5470 5500 5510 5520 5530 5550 5550 5550 5550 5570 5600 5610 5620 5640 5620 5640 5650	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2912 2920 2928 2916 2912 2920 2928 2936 2944 2952 2960 2968 2976 2984 2992	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945 2953 2961 2965 2961 2965 2985 2993	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930 2914 2923 2938 2946 2954 2954 2970 2986 2994	2819 2827 2835 2835 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2931 2939 2947 2955 2963 2971 2995	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948 2956 2956 2964 2972 2988 2996	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2901 2901 2901 2917 2925 2933 2941 2957 2957 2957 2965 2973 2989 2997	2822 2830 2838 2846 2854 2854 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950 2958 2958 2958 2958 2958 2958	2823 2831 2839 2847 2855 2863 2871 2895 2903 2911 2919 2927 2935 2943 2951 2959 2957 2959 2967 2975 2983 2991 2999
to to 5777 3071	5010 5020 5040 5050 5060 5070 5100 5110 5120 5130 5140 5150 5160 5220 5220 5220 5220 5220 5220 5220 52	2560 2568 2576 2592 2600 2608 2616 2624 2632 2640 2648 2646 2646 2646 2646 2646 2646 2648 2696 2704 2712 2720 2728 2736 2744 2752	2561 2563 2577 2585 2593 2601 2609 2617 2625 2633 2649 2657 2665 2673 2681 2689 2697 2705 2713 2721 2729 2737 2745	2562 2578 2578 2594 2602 2618 2626 2618 2626 2634 2650 2658 2656 2658 2656 2674 2682 2690 2698 2706 2714 2722 2730 2738 2746	2563 2579 2587 2595 2603 2611 2619 2627 2635 2643 2651 2659 2667 2675 2683 2691 2699 2707 2715 2713 2713 2713 2731 2731 2731 2731 2731	2564 2572 2588 2596 2604 2612 2620 2628 2634 2652 2652 2652 2652 2652 2652 2652 265	2565 2573 2581 2597 2605 2613 2621 2629 2637 2645 2653 2653 2661 2669 2677 2685 2653 2661 2669 2677 2685 2693 2701 2701 2701 2701 2717 2725 2733 2741 2749 2757	25566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2654 2654 2654 2654 2710 2718 2718 2718 2718 2718 2718 27158	2567 2575 2583 2591 2599 2607 2615 2623 2623 2631 2639 2647 2655 2663 2671 2679 2687 2695 2703 2711 2727 2719 2727 2715 2743 2751	5410 5420 5430 5450 5460 5470 5500 5510 5520 5530 5540 5550 5540 5550 5570 5600 5610 5660 5630 5660 5630 5630 5630 5630 563	28.16 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2944 2952 2952 2954 2956 2954 2952 2968 2976 2984 2992 3000 3008	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2973 2905 2913 2921 2929 2937 2945 2953 2953 2953 2953 2977 2985 2993 3001 3009	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2990 2914 2922 2930 2914 29254 2954 2954 2954 2954 2970 2978 2978 2978 2978 2978 2974 3002	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2897 2907 2915 2923 2931 2939 2947 2955 2953 2971 2979 2987 2975 3003 3011	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2908 2916 2924 2932 2940 2948 2956 2956 2956 2956 2988 2996 3004	2821 2823 2845 2853 2861 2869 2877 2885 2893 2909 2917 2925 2933 2941 2949 2957 2957 2957 2965 2973 2981 2989 2997 3005 3013	2822 2830 2838 2846 2854 2854 2862 2870 2878 2886 2894 2910 2918 2926 2934 2942 2950 2958 2958 2958 2958 2996 2998 3006 3014	2823 2831 2839 2847 2855 2863 2879 2887 2895 2903 2911 2919 2927 2935 2943 2951 2959 2959 2963 2991 2999 3007 3015
to 5777 3071 (Octal) (Decimal)	5010 5020 5040 5050 5060 5070 5100 5110 5120 5130 5130 5150 5160 5170 5200 5210 5220 5230 5240 5220 5240 5220 5260 5270 5260 5270 5260 5270 5260 5270	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2672 2680 2688 2696 2704 2712 2720 2728 2736 2736 2736 2736	2561 2569 2577 2585 2593 2601 2609 2617 2625 2633 2641 2657 2657 2657 2657 2657 2657 2657 2681 2689 2697 2705 2713 2721 2721 2737 2745 2753 2753	2562 2570 2578 2594 2602 2618 2626 2634 2642 2658 2658 2658 2658 2658 2658 2690 2698 2706 2714 2722 2730 2738 2746	2563 2571 2577 2587 2603 2611 2619 2627 2659 2667 2659 2667 2659 2667 2659 2669 2707 2715 2723 2731 2731 2731 2739 2731 2739	2564 2572 2588 2596 2604 2612 2620 2628 2636 2644 2652 2660 2668 2668 2668 2668 2668 2668 2714 2732 2770 2772 2756 2774 2756	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2669 2677 2685 2693 2701 2709 2717 2725 2733 2741 2749 2757 2765 2773	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2662 2662 2662 2662 2662 2710 2718 2726 2714 2724 2734 2758 27758	2567 2575 2583 2599 2607 2615 2623 2631 2639 2647 2655 2663 2663 2663 2671 2655 2703 2711 2719 2727 2735 2743 2751 2759	5410 5420 5430 5440 5450 5460 5470 5510 5520 5520 5520 5520 5520 5520 552	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2944 2952 2960 2944 2952 2960 2958 2976 2958 2976 2988 2976 2988 2976 2988 2976 2988 2976 2988 2976 2988 2976 2988 2976 2988 2976 2976 2976 2976 2976 2976 2976 2976	2817 2823 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2929 2937 2929 2937 2945 2953 2961 2953 2961 2953 2961 2953 2961 2953 2961 2953 2961 2953 2961 2953 2961 2953 2961 2965 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2985 2977 2977 2985 2977 2985 2977 2977 2977 2977 2977 2977 2977 297	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930 2938 2946 2954 2954 2954 2954 2954 2970 2978 2986 2994 3002 3010 3018 3026	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2931 2939 2947 2955 2963 2971 2975 2975 2975 3003 3011 3027	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948 2956 2956 2956 2956 2956 2956 2996 3004 3012 3020	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917 2925 2933 2941 2949 2957 2957 2965 2973 2981 2989 2987 3005 3013 3029	2822 2830 2838 2846 2854 2854 2854 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950 2958 2958 2958 2958 2958 2995 3006 3014 3022 3030	2823 2831 2839 2847 2855 2863 2871 2879 2887 2895 2903 2911 2927 2927 2927 2925 2943 2951 2959 2967 2959 2967 2991 2999 3007 3015 3023 3031
to 5777 3071 (Octal) (Decimal)	5010 5020 5030 5040 5050 5060 5070 5100 5120 5130 5140 5140 5140 5140 5170 5200 5210 5220 5230 5240 5220 5230 5240 5230 5240 5230 5240 5230 5330	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2648 2656 2664 2672 2680 2688 2696 2704 2712 2720 2728 2736 2744 2752 2768 2776	2561 2565 2593 2601 2609 2617 2625 2633 2641 2649 2657 2665 2673 2681 2689 2657 2705 2713 2721 2721 2721 2745 2745 2753 2761 2761 2777	2562 2570 2578 2594 2602 2610 2618 2626 2634 2642 2650 2658 2666 2674 2682 2690 2698 2706 2714 2722 2738 2746 2754 2754	2563 2571 2579 2603 2611 2619 2627 2635 2643 2651 2659 2667 2659 2667 2659 2667 2667 2669 2707 2715 2723 2731 2739 2737 2739 2747 2755 2765	2564 2572 2580 2596 2604 2612 2620 2628 2668 2668 2668 2668 2668 266	2565 2573 2581 2597 2605 2613 2629 2637 2645 2653 2669 2677 2685 2693 2701 2709 2717 2725 2731 2749 2749 2757 2757 2757 2757 2757 2757	2566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2662 2670 2678 2686 2694 2734 2710 2718 2742 2742 2742 2750 2758 2758 2758	2567 2575 2583 2599 2607 2615 2623 2631 2639 2647 2655 2663 2663 2663 2663 2663 2663 2671 2679 2687 2695 2711 2719 2727 2735 2743 2751 2759 2767 2775 2775 2775 2775	5410 5420 5430 5450 5460 5470 5500 5510 5520 5530 5520 5530 5530 5550 5550 555	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2944 2952 2960 2968 2976 2968 2976 2984 2992 3000 3008 3018 3024 3032	2817 2823 2831 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945 2953 2961 2969 2977 2985 2961 2969 2977 2985 3001 3009 3017 3025 3033	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930 2938 2946 2954 2954 2954 2954 2954 29878 2986 29970 2978 29878 2984 3002 3010 3016 3034	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2907 2915 2923 2931 2939 2947 2955 2971 2955 2963 2971 2979 2987 2955 3003 3011 3011 30127 3035	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948 2956 2954 2956 2964 2972 2980 2988 3004 3012 3028 3036	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2909 2917 2925 2933 2941 2949 2957 2955 2973 2965 2973 2981 2989 2997 3005 3013 3021 3029	2822 2830 2838 2846 2854 2854 2862 2870 2878 2886 2894 2902 2910 2918 2910 2918 2942 2950 2958 2956 2974 2982 2990 2998 2996 3006 3014 3020 3038	2823 2831 2839 2847 2855 2863 2671 2879 2887 2895 2903 2911 2919 2927 2935 2943 2951 2959 2967 2959 2967 2959 2967 2999 3007 3015 3031 3039
to 5777 3071 (Octal) (Decimal)	5010 5020 5030 5040 5050 5060 5070 5100 5120 5130 5140 5130 5140 5170 5200 5210 5220 5230 5240 5250 5260 5270 5310 5310 5310 5310 5310 5310	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2640 2640 2640 2656 2664 2656 2664 2672 2680 2688 2696 2704 2712 2720 2728 2770 2728 2736 2744 2752 2760 2768 2776	2561 2563 2577 2585 2593 2601 2609 2617 2625 2633 2641 2657 2665 2673 2681 2689 2697 2705 2713 2721 2729 2737 2745 2753 2761 2769 2777 2785	2562 2578 2578 2594 2602 2618 2626 2618 2626 2634 2642 2658 2658 2658 2658 2658 2658 2658 265	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2651 2659 2667 2659 2667 2659 2667 2659 2667 2659 2669 2707 2715 2773 2773 2773 27747 2755 2763 2771 2775	2564 2572 2588 2596 2604 2612 2620 2628 2644 2652 2660 2668 2676 2684 2692 2700 2708 2716 2714 2732 2740 2742 2740 2748 2756 2764 2772 2788 2796	2565 2573 2581 2597 2605 2613 2621 2629 2637 2645 2663 2661 2669 2677 2685 2661 2669 2677 2685 2693 2701 2709 2717 2725 2733 2741 2749 2757 2755 2773 2781 2789 2797	25566 2574 2582 2590 2598 2606 2614 2632 2642 2642 2654 2654 2654 2654 2654 2702 2718 2776 2718 2776 2718 2776 2774 2775 2758 2758	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2663 2671 2679 2687 2695 2703 2711 2719 2727 2735 2743 2751 2759 2767 2775 2775 2775 2779 2779	5410 5420 5430 5450 5460 5470 5500 5510 5520 5530 5540 5550 5570 5570 5600 5610 5660 5660 5660 5660 5660 566	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2944 2952 2960 2952 2960 2958 2976 2984 2976 2984 2976 2984 2976 2984 2976 3000 3008 3016 3024	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945 2953 2961 2953 2961 2953 2977 2985 2993 3001 3009 3017 3025 3031 3049	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2914 2922 2930 2914 2922 2930 2914 29254 2954 2954 2954 2954 2970 2978 2986 2994 3002 3010 3018 3026 3034 3050	2819 2827 2835 2843 2851 2859 2867 2875 2883 2891 2899 2915 2923 2931 2939 2915 2923 2931 2939 2947 2955 2963 2971 2975 3003 3011 3019 3027 3035 3051	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948 2956 2956 2954 2956 2980 2980 2980 2988 2996 3004 3012 3020 3028 3036 3036	2821 2823 2837 2845 2853 2861 2869 2877 2885 2893 2901 2901 2901 2917 2925 2933 2941 2949 2957 2957 2957 2957 2989 2997 3005 3013 3021 3029 3037 3045	2822 2830 2838 2846 2854 2854 2856 2870 2878 2886 2894 2902 2918 2926 2934 2942 2950 2958 2958 2958 2958 2958 2958 2958 2990 2998 3006 3014 3022 3030 3038 3054	2823 2831 2839 2847 2855 2863 2879 2887 2895 2901 2911 2919 2927 2935 2943 2951 2959 2967 2975 2983 2991 2999 3007 3015 3023 3031 3039 3047
to 5777 3071 (Octal) (Decimal)	5010 5020 5040 5050 5060 5070 5100 5120 5120 5130 5140 5170 5200 5210 5220 5230 5240 5250 5260 5270 5310 5310 5310 5310 5310 5310 5310 5320 5330	2560 2568 2576 2584 2592 2600 2608 2616 2624 2632 2640 2640 2640 2640 2656 2664 2656 2664 2656 2664 2672 2680 2704 2712 2720 2728 2770 2728 2736 2744 2752 2760 2768 2776 2768	2561 2563 2577 2585 2593 2601 2609 2617 2625 2633 2641 2657 2665 2673 2681 2689 2697 2705 2713 2721 2729 2737 2745 2753 2761 2769 2777 2785 2793 2801	2562 2578 2578 2578 2594 2602 2618 2626 2634 2642 2658 2658 2658 2658 2658 2658 2658 265	2563 2571 2579 2587 2595 2603 2611 2619 2627 2635 2651 2659 2667 2659 2667 2659 2667 2659 2667 2659 2669 2707 2715 2773 2773 2773 2773 27747 2775 2785 2779 2785 2779 2785 2779 2785 2779 2785 2779 2785 2779 2785 2779 2785 2779 2779 2779 2779 2779 2779 2779 277	2564 2572 2588 2596 2604 2612 2620 2628 2644 2652 2660 2668 2668 2666 2668 2666 2668 2666 2668 2676 2708 2716 2700 2708 2714 2712 2740 2742 2740 2748 2756 2764 2772 2780 2786 2786 2786 2786 2786 2764 2772 2786 2786 2786 2786 2786 2786 2786	2565 2573 2581 2597 2605 2613 2621 2629 2637 2645 2663 2661 2669 2677 2685 2661 2669 2677 2685 2693 2701 2709 2717 2725 2733 2741 2749 2757 2765 2773 2781 2797 2805	25566 2574 2582 2590 2598 2606 2614 2622 2630 2638 2646 2654 2662 2654 2662 2654 2662 2710 2718 2726 2718 2776 2718 2775 2758 2758 2758 2758 2758 2758 275	2567 2575 2583 2591 2599 2607 2615 2623 2631 2639 2647 2655 2655 2655 2663 2671 2695 2703 2711 2719 2727 2735 2743 2751 2759 2767 2759 2767 2759 2767 2759 2767 2775 2779 2779 2779 2779 2779 2779 277	5410 5420 5430 5450 5460 5470 5500 5510 5520 5530 5520 5530 5550 5550 5550 555	2816 2824 2832 2840 2848 2856 2864 2872 2880 2888 2896 2904 2912 2920 2928 2936 2944 2952 2960 2952 2960 2958 2976 2984 2992 3000 3008 3016 3024 3032 3048 3056	2817 2825 2833 2841 2849 2857 2865 2873 2881 2889 2897 2905 2913 2921 2929 2937 2945 2953 2961 2953 2961 2953 2977 2985 2993 3001 3009 3017 3025 3033 3041 3049 3057	2818 2826 2834 2842 2850 2858 2866 2874 2882 2890 2898 2906 2914 2922 2930 2938 2946 2954 2954 2954 2954 2954 2954 2986 2994 3002 3010 3018 3026 3058	2819 2827 2835 2835 2851 2859 2867 2875 2883 2891 2899 2915 2923 2931 2939 2915 2923 2931 2939 2947 2955 2963 2971 2975 3003 3011 3019 3027 3035 3059	2820 2828 2836 2844 2852 2860 2868 2876 2884 2892 2900 2908 2916 2924 2932 2940 2948 2956 2954 2956 2964 2956 2980 2988 2996 3004 3012 3020 3028 3036 30452 3052	2821 2829 2837 2845 2853 2861 2869 2877 2885 2893 2901 2901 2901 2901 2917 2925 2933 2941 2949 2957 2957 2957 2957 2965 2973 2981 2989 2997 3005 3013 3021 3029 3037 3045 3051	2822 2830 2838 2846 2854 2852 2870 2878 2886 2894 2902 2910 2918 2926 2934 2942 2950 2958 2958 2958 2958 2958 2958 2958 2998 3006 3014 3022 3030 3038 3046 3054 3054	2823 2831 2839 2847 2855 2863 2879 2887 2895 2901 2911 2919 2927 2935 2943 2951 2959 2967 2975 2983 2991 2999 3007 3015 3023 3031 3039 3047

	1	0		2	3	4	5	6	7		0	1	2		4	5	6	7
		3072								.			3330	· · · ·	1117	1111	1114	1115
6000 3072 to to	6010	3080	3081	3082	3083	3084	3085	3086	3087	6410	3336	3337	3338	3339	3340	3341	3342	3343
6777 3583	6020	3088 3096	3089 3097	3090 3098	3091 3099	3092 3100	3093 3101	3094 3102	3095 3103				3346 3354					
(Octal) (Decimal)	6040	3104	3105	3106	3107	3108	3109	3110	3111	6440	3360	3361	·3362	3363	3364	3365	3366	3367 3375
	6060	3112 3120	3121	3122	3123	3124	3125	3126	3127	6460	3376	3377	3370 3378	3379	3380	3381	3382	3383
Octal Decimal 10000 4096	6070	3128	3129	3130	3131	3132	3133	3134	3135	6470	3384	3385	3386	3387	3388	3389	3390	3391
20000 8192		3136											3394					
30000 - 12288 40000 - 16384		3144 3152											3402 3410					
50000 20480 60000 24576	6130	3160	3161	3162	3163	3164	3165	3166	3167	6530	3416	3417	3418	3419	3420	3421	3422	3423
70000 - 28672		3168 3176											3426 3434					3439
•		3184 3192											3442 3450					
	6200 6210	3200			3203 3211								3458 3466					
		3216								6620			3474 3482					
	6240	3232	3233	3234	3235	3236	3237	3238	3239		3488	3489	3490	3491	3492	3493	3494	3495
		3240 3248								6650 6660			3498 3506					
		3256											3514					
		3264											3522					
	6310 6320	3272 3280	3273	3274	3275	3276	3285	3286	3287	6710 6720			3530 3538					
	6330	3288 3296	3289	3290	3291	3292	3293	3294	3295	6730 6740			3546 3554				3550 3558	
	6350	3304	3305	3306	3307	3308	3309	3310	3311	6750	3560	3561	3562	3563	3564	3565	3566	3567
		3312									3568 3576		3570 3578				3574 3582	
	_	Leans	فتقريب وسنعد						and the second secon	Concession of the local division of the loca								
		0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
7000 3584	7000	-	1	2						7400								7 3847
7000 3584 to to		3584	3585	3586 3594	3587 3595	3588 3596	3589 3597	3590 3598	3591 3599	7410	3840 3848	3841 3849	3842 3850	3843 3851	3844 3852	3845 3853	3846 3854	3847 3855
	7010 7020	3584 3592 3600	3585 3593 3601	3586 3594 3602	3587 3595 3603	3588 3596 3604	3589 3597	3590 3598 3606	3591 3599 3607	7410 7420	3840 3848 3856	3841 3849 3857 3865	3842 3850 3858 3866	3843 3851 3859 3867	3844 3852 3860 3868	3845 3853 3861 3869	3846 3854 3862 3870	3847 3855 3863 3871
to to 7777 4095	7010 7020 7030 7040	3584 3592 3600 3608 3616	3585 3593 3601 3609 3617	3586 3594 3602 3610 3618	3587 3595 2603 3611 3619	3588 3596 3604 3612 3620	3589 3597 3605 3613 3621	3590 3598 3606 3614 3622	3591 3599 3607 3615 3623	7410 7420 7430 7440	3840 3848 3856 3864 3872	3841 3849 3857 3865 3873	3842 3850 3858 3866 3874	3843 3851 3859 3867 3875	3844 3852 3860 3868 3876	3845 3853 3861 3869 3877	3846 3854 3862 3870 3878	3847 3855 3863 3871 3879
to to 7777 4095	7010 7020 7030	3584 3592 3600 3608 3616 3624 3632	3585 3593 3601 3609 3617 3625 3633	3586 3594 3602 3610 3618 3626 3634	3587 3595 3603 3611 3619 3627 3635	3588 3596 3604 3612 3620 3628 3636	3589 3597 3605 3613 3621 3629 3637	3590 3598 3606 3614 3622 3630 3638	3591 3599 3607 3615 3623 3631 3639	7410 7420 7430 7440 7450 7460	3840 3848 3856 3864 3872 3880 3888	3841 3849 3857 3865 3873 3881 3889	3842 3850 3858 3866 3874 3882 3890	3843 3851 3859 3867 3875 3883 3891	3844 3852 3860 3868 3876 3884 3892	3845 3853 3861 3869 3877 3885 3893	3846 3854 3862 3870 3878 3886 3894	3847 3855 3863 3871 3879 3887 3895
to to 7777 4095	7010 7020 7030 7040 7050	3584 3592 3600 3608 3616 3624 3632 3640	3585 3593 3601 3609 3617 3625 3633 3641	3586 3594 3602 3610 3618 3626 3634 3642	3587 3595 3603 3611 3619 3627 3635 3643	3588 3596 3604 3612 3620 3628 3636 3644	3589 3597 3605 3613 3621 3629 3637 3645	3590 3598 3606 3614 3622 3630 3638 3646	3591 3599 3607 3615 3623 3631 3639 3647	7410 7420 7430 7440 7450 7460 7460 7470	3840 3848 3856 3864 3872 3880 3888 3896	3841 3849 3857 3865 3873 3881 3889 3897	3842 3850 3858 3866 3874 3882 3890 3898	3843 3851 3859 3867 3875 3883 3891 3899	3844 3852 3860 3868 3876 3884 3892 3900	3845 3853 3861 3869 3877 3885 3893 3901	3846 3854 3862 3870 3878 3886 3894 3902	3847 3855 3863 3871 3879 3887 3895 3903
to to 7777 4095	7010 7020 7030 7040 7050 7060 7070 7100	3584 3592 3600 3608 3616 3624 3632 3640 3648	3585 3593 3601 3609 3617 3625 3633 3641 3649	3586 3594 3602 3610 3618 3626 3634 3642 3650	3587 3595 3603 3611 3619 3627 3635 3643 3651	3588 3596 3604 3612 3620 3628 3636 3644 3652	3589 3597 3605 3613 3621 3629 3637 3645 3653	3590 3598 3606 3614 3622 3630 3638 3646 3654	3591 3599 3607 3615 3623 3631 3639 3647 3655	7410 7420 7430 7440 7450 7460 7470 7500	3840 3848 3856 3864 3872 3880 3888 3896 3904	3841 3849 3857 3865 3873 3881 3889 3897 3905	3842 3850 3858 3866 3874 3882 3890 3898 3906	3843 3851 3859 3867 3875 3883 3891 3899 3907	3844 3852 3860 3868 3876 3884 3892 3900 3908	3845 3853 3861 3869 3877 3885 3893 3901 3909	3846 3854 3862 3870 3878 3886 3894 3902 3910	3847 3855 3863 3871 3879 3887 3895 3903 3911
to to 7777 4095	7010 7020 7030 7040 7050 7050 7060 7070 7100 7110 7120	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666	3587 3595 2603 3611 3619 3627 3635 3643 3651 3659 3667	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668	3589 3597 3605 3613 3621 3629 3637 3645 3653 3665 3661 3669	3590 3598 3606 3614 3622 3630 3638 3646 3654 3654 3652 3670	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671	7410 7420 7430 7440 7450 7460 7470 7500 7510 7520	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927
to to 7777 4095	7010 7020 7030 7040 7050 7060 7070 7110 7110 7110 7120 7130 7140	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3652 3680	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684	3589 3597 3605 3613 3621 3629 3637 3645 3665 3665 3669 3677 3685	3590 3598 3606 3614 3622 3630 3638 3646 3654 3654 3654 3670 3678 3686	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3679 3687	7410 7420 7430 7440 7450 7460 7470 7500 7510 7520 7530 7540	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937	3842 3850 3858 3865 3874 3882 3890 3898 3906 3914 3922 3930 3938	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3939	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926 3934 3942	3847 3855 3863 3871 3879 3887 3895 3963 3911 3919 3927 3935 3942
to to 7777 4095	7010 7020 7030 7040 7050 7060 7070 7110 7110 7120 7130 7140 7150	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3656 3664 3672 3680 3688	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690	3587 3595 2603 3611 3619 3627 3635 3643 3651 3653 3667 3675 3683 3691	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3692	3589 3597 3605 3613 3621 3629 3637 3645 3653 3661 3669 3677 3685 3693	3590 3598 3606 3614 3622 3630 3638 3646 3654 3654 3654 3670 3678 3686 3694	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3679 3687 3695	7410 7420 7430 7440 7450 7460 7470 7500 7510 7520 7530 7530 7540 7550	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951
to to 7777 4095	7010 7020 7030 7040 7050 7060 7070 7110 7110 7110 7110 7110 7130 7140 7150 7160	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3672 3680 3688 3696	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3689 3689 3697	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698	3587 3595 3603 3611 3619 3627 3635 3643 3651 3651 3655 3663 3675 3683 3691 3699	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3692 3700	3589 3597 3605 3613 3621 3629 3637 3645 3653 3661 3663 3661 3665 3677 3685 3693 3701	3590 3598 3606 3614 3622 3630 3638 3646 3654 3654 3678 3678 3686 3694 3702	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3679 3687	7410 7420 7430 7440 7450 7460 7470 7500 7510 7520 7530 7530 7530 7530 7530 7550 7550	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3934 3952	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953	3842 3850 3858 3865 3874 3882 3890 3898 3906 3914 3922 3930 3938	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947 3955	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3958	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959
to to 7777 4095	7010 7020 7030 7040 7050 7050 7070 7110 7110 7110 7110 7130 7140 7150 7170 7170	3584 3592 3600 3616 3624 3632 3640 3648 3656 3664 3672 3680 3688 3696 3704 3712	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3687 3705 3713	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3650 3658 3666 3674 3682 3690 3698 3706	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683 3691 3707 3715	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3692 3700 3708	3589 3597 3605 3613 3621 3629 3637 3645 3663 3661 3669 3677 3685 3693 3701 3709	3590 3598 3606 3614 3622 3630 3638 3646 3654 3654 3654 3686 3694 3710 3718	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3679 3687 3695 3703 3711 3719	7410 7420 7430 7440 7450 7460 7470 7500 7510 7520 7530 7540 7550 7560 7570 7600	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3968	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3961	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3970	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947 3955 3963 3971	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3972	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3965 3973	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3958 3966 3974	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975
to to 7777 4095	7010 7020 7030 7040 7050 7060 7070 7110 7110 7110 7110 7110 7150 7160 7170 7170 71200 7210	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3672 3680 3688 3696 3704 3712 3720	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3697 3705 3713 3721	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3656 3674 3682 3690 3698 3706 3714 3722	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683 3691 3699 3707 3715 3723	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3692 3700 3708 3716 3716	3589 3597 3605 3613 3629 3637 3645 3653 3661 3669 3677 3685 3693 3701 3709 3717 3725	3590 3598 3606 3614 3620 3638 3646 3654 3654 3678 3678 3686 3694 3702 3710 3718 3726	3591 3599 3607 3615 3623 3631 3639 3647 3655 3667 3679 3687 3695 3703 3711 3719 3727	7410 7420 7430 7440 7450 7460 7460 7510 7550 7550 7550 7550 7550 7550 7560 7570 7600 7610	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3950 3952 3960	3841 3849 3857 3865 3873 3889 3897 3905 3913 3929 3937 3945 3953 3961 3969 3977	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3952 3970 3978	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947 3955 3963 3971 3979	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3956	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3965 3973 3981	3846 3854 3862 3878 3878 3894 3902 3910 3918 3926 3934 3942 3950 3958 3966 3974 3982	3847 3855 3863 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975 3983
to to 7777 4095	7010 7020 7030 7040 7050 7060 7070 7110 7110 7120 7110 7120 7150 7160 7170 7200 7210 7220 7230	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3656 3664 3672 3680 3688 3696 3704 3712 3720 3728	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3697 3705 3713 3721 3723	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730	3587 3595 2603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683 3691 3699 3707 3715 3723 3731	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3684 3676 3684 3676 3700 3708 3716 3720 3718	3589 3597 3605 3613 3621 3629 3637 3645 3653 3661 3669 3677 3685 3693 3701 3709 3717 3725 3733 3741	3590 3598 3606 3614 3622 3630 3638 3646 3654 3662 3670 3678 3686 3694 3702 3710 3718 3726 3734 3742	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3675 3703 3711 3719 3727 3735 3743	7410 7420 7430 7450 7450 7450 7450 7510 7510 7520 7530 7550 7550 7550 7550 7570 7560 7570 7600 7610 7620 7630	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3968 3976 3984 3976	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3969 3977 3985 3993	3842 3850 3858 3864 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3956 3970 3978 3986 3994	3843 3851 3859 38675 3883 3891 3899 3907 3915 3923 3931 3939 3947 3953 3971 3971 3971	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3972 3980 3988 3996	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3965 3973 3981 39897	3846 3854 3862 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3958 3966 3974 3966 3974	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975 3987 3991 3999
to to 7777 4095	7010 7020 7030 7050 7050 7050 7070 7100 7110 7120 7130 7140 7150 7170 71200 7210 7220 7220 7220 7220 72	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3648 3656 3664 3672 3688 3696 3704 3712 3720 3728 3736	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3687 3705 3713 3721 3721 3721 3737 3745	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3738 3746	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683 3691 3699 3707 3715 3723 3731 3739 3747	3588 3596 3604 3612 3620 3632 3636 3644 3652 3684 3676 3684 3676 3708 3716 3716 3716 3724 3740 3748	3589 3597 3605 3613 3621 3629 3637 3645 3665 3665 3665 3665 3665 3665 3667 3665 3677 3685 3685 3701 3709 3717 3725 3733 3741	3590 3598 3606 3614 3622 3630 3646 3654 3662 3670 3678 3686 3694 3702 3710 3718 3726 3734 3742 3750	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3679 3687 3695 3703 3711 3719 3727 3735 3743	7410 7420 7430 7440 7450 7450 7450 7550 7550 7550 755	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3968 3976 3984 3992	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3969 3977 3985 3993	3842 3850 3858 3864 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3970 3978 3996	3843 3851 3859 3867 3875 3883 3891 3999 3915 3923 3931 3939 3947 3955 3963 3971 3979 3987 3997 3987 3995	3844 3852 3860 3868 3876 39884 3990 3908 3916 3924 3932 3940 3948 3956 3964 3956 3964 3972 3980 3988 3996	3845 3853 3861 3867 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3965 3973 3981 3981 3987 4005	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3958 3966 3974 3982 3990 3998	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3951 3951 3957 3967 3975 3983 3991 3991
to to 7777 4095	7010 7020 7030 7040 7050 7050 7050 7050 7050 7110 7120 7140 7150 7150 7150 7150 7150 7150 7150 7210 7220 7210 7220 7220 7250 7250 7250	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3656 3664 3672 3680 3688 3696 3704 3712 3720 3728 3736 3744 3752 3760	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3689 3687 3705 3713 3721 3729 3737 3745 3753 3761	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3738 3746 3754 3762	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3663 3691 3699 3707 3715 3723 3731 3739 3747 3755 3763	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3672 3700 3708 3770 3716 3724 3740 3740 3740	3589 3597 3605 3613 3621 3629 3637 3645 3665 3665 3665 3665 3665 3673 3665 3701 3709 3717 3709 3717 3741 3749 3757 3765	3590 3598 3606 3614 3622 3630 3638 3646 3638 3646 3654 3662 3670 3678 3702 3710 3718 3742 3750 3758 3766	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3695 3703 3711 3719 3727 3735 3743 3751 3759 3767	7410 7420 7430 7450 7450 7450 7450 7510 7510 7520 7530 7540 7550 7560 7570 7660 7610 7620 7630 7640 7640 7660 7660 7660 7660	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3968 3976 3984 3976 3984 3992 4000 4008	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3927 3945 3953 3961 3969 3977 3965 3993 4001 4009 4017	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3954 3970 3978 3986 3994 4002 4010	3843 3851 3857 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947 3955 3963 3971 3979 3987 3995 4003 4011 4019	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3972 3980 3988 3996 4004 4012	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3931 3941 3949 3957 3965 3973 3981 3989 3997 4005 4013	3846 3854 3862 3878 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3958 3966 3974 3958 3966 3974 3998 4008 4014 1022	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3945 3951 3959 3967 3975 3987 3991 3999 4007 4015
to to 7777 4095	7010 7020 7030 7050 7050 7050 7050 7050 7050 7110 7120 7110 7110 7110 7110 7110 711	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3672 3688 3696 3704 3712 3720 3728 3736 3744 3752 3766 3768	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3697 3705 3713 3721 3729 3737 3745 3753 3761 3769	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3738 3746 3754 3762 3770	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3683 3691 3699 3707 3715 3723 3731 3739 3747 3755 3763 3771	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3700 3708 3716 3732 3740 3778 3754 3772	3589 3597 3605 3613 3621 3629 3637 3645 3653 3661 3669 3677 3685 3693 3701 3709 3717 3725 3733 3741 3749 3757 3765 3773	3590 3598 3606 3614 3622 3630 3638 3646 3654 3654 3654 3654 3694 3702 3710 3718 3726 3734 3734 3734 3750 3758 3766 3774	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3695 3703 3711 3719 3727 3735 3743 3751 3759 3767 3775	7410 7420 7430 7450 7450 7450 7510 7510 7520 7530 7540 7550 7560 7560 7560 7660 7650 7660 7650 7660 7670	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3968 3976 3984 3976 3984 4000 4008 4016 4024	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3969 3977 3985 3993 4001 4009 4017 4025	3842 3850 3858 3864 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3970 3978 3976 3978 3994 4002 4010 4018 4026	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3934 3955 3963 3971 3975 3963 3971 3975 4003 4011 4019 4027	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3972 3980 3988 3996 4004 4012 4020 4028	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3965 3973 3981 3989 3997 4005 4013 4021 4029	3846 3854 3862 3878 3886 3894 3902 3910 3918 3926 3934 3926 3934 3950 3958 3966 3974 3950 3998 4006 4014 1022 4030	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975 3987 3991 3999 4007 4015 4023 4031
to to 7777 4095	7010 7020 7030 7050 7050 7050 7050 7050 7050 705	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3672 3688 3696 3704 3712 3720 3728 3736 3744 3752 3768 3776	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3687 3705 3713 3721 3725 3733 3745 3753 3761 3769 3777	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3738 3746 3754 3754 3770 3778	3587 3595 2603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683 3691 3699 3707 3715 3723 3731 3739 3747 3755 3763 3771 3779	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3700 3708 3716 3720 3740 3748 3756 3772 3780	3589 3597 3605 3613 3621 3629 3637 3645 3653 3661 3669 3677 3765 3773 3725 3733 3741 3749 3757 3775 3773 3781	3590 3598 3606 3614 3622 3630 3638 3646 3654 3662 3670 3678 3782 3710 3718 3726 3734 3742 3750 3758 3774 37782	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3695 3703 3711 3719 3727 3735 3743 3751 3759 3767 3775 3783	7410 7420 7430 7450 7450 7450 7500 7510 7500 7510 7500 7550 7550 75	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3934 3952 3950 3956 3968 3976 3968 3976 3988 4000 4008 4016 4024	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3969 3977 3969 3977 3969 3977 3985 3993 4001 4009 4017 4025 4033	3842 3850 3858 3864 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3954 3970 3978 3986 3994 4002 4010 4018 4026 4034	3843 3851 3859 38675 3883 3891 3899 3907 3915 3923 3931 3939 3945 3953 3951 3971 3975 3963 3971 3975 4003 4011 4019 4027 4035	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3972 3964 3972 3980 3988 3996 4004 4012 4020 4028 4036	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3955 3973 3981 3989 3997 4005 4013 4021 4029 4037	3846 3854 3862 3870 3878 3886 3894 3902 3910 3918 3926 3934 3950 3958 3966 3974 3950 3998 4006 4014 1022 4030 4038	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975 3987 3997 3997 4007 4015 4023 4031 4035
to to 7777 4095	7010 7020 7030 7040 7050 7050 7050 7050 7050 7110 7110 711	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3656 3664 3696 3704 3712 3720 3728 3736 3744 3752 3768 3776 3778	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3685 3687 3765 3705 3713 3721 3729 3737 3745 3753 3761 3769 3777 3765 3793	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3738 3746 3754 3762 3770	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3665 3683 3691 3699 3707 3715 3723 3731 3739 3747 3755 3763 3771	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3672 3700 3708 3716 3772 3740 3778 3756 3756 3758 3776	3589 3597 3605 3613 3621 3629 3637 3645 3665 3665 3665 3665 3665 3665 3665	3590 3598 3606 3614 3622 3630 3638 3646 3638 3646 3654 3662 3670 3678 3768 3770 3718 3748 3710 3718 3745 3758 3758 3758 3774 3782 3790 3798	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3695 3703 3711 3719 3727 3735 3743 3751 3753 3767 3775 3767 3775 3783 3799	7410 7420 7430 7450 7450 7450 7550 7550 7550 7550 755	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3944 3952 3960 3968 3976 3984 3992 4000 4008 4016 4024	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3921 3921 3921 3923 3945 3953 3961 3969 3977 3985 3993 4001 4009 4017 4025 4033	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 39346 3954 3954 3954 3954 3954 3954 3970 3978 3986 3994 4002 4010 4018 4026 4034	3843 3851 3857 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947 3955 3963 3971 3979 3987 3995 4003 4011 4019 4027 4035	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3956 3964 3972 3980 3988 3996 4004 40120 4028 4036	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3931 3941 3949 3957 3965 3973 3981 3989 3997 4005 4013 4021 4029 4037 4045	3846 3854 3862 3878 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3958 3942 3950 3958 3966 3974 3982 3990 3998 4006 4014 4039 4039	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3959 3967 3975 3983 3991 3999 4007 4015
to to 7777 4095	7010 7020 7030 7050 7050 7050 7050 7050 7050 705	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3656 3664 3672 3688 3696 3704 3712 3720 3728 3744 3752 3766 3768 3776 3778 3778	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3697 3705 3713 3745 3773 3745 3753 3761 3769 3777 3785 3793 3801 3809	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3658 3666 3698 3706 3714 3722 3730 3738 3746 3754 3770 3778 3776 3778 3778 3778	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3683 3693 3707 3715 3723 3731 3739 3747 3755 3763 3771 3779 3787 3795 3803 3811	3588 3596 3604 3612 3620 3628 3636 3644 3652 3660 3668 3676 3684 3672 3700 3708 3716 3772 3740 3772 3748 3772 3772 3788 3776	3589 3597 3605 3613 3621 3629 3637 3645 3653 3661 3669 3677 3685 3693 3701 3709 3717 3725 3733 3741 3749 3755 3773 3775 3773 3765 3773	3590 3598 3606 3614 3622 3630 3638 3646 3654 3662 3670 3678 3702 3710 3718 3726 3734 3742 3750 3758 3774 3778 3776 3778 3778	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3703 3711 3719 3727 3735 3743 3751 3759 3767 3775 3783 3791 3791 3797 3815	7410 7420 7430 7450 7450 7450 7450 7510 7520 7530 7540 7550 7550 7560 7550 7560 7560 7660 7670 7660 7670 767	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3968 3976 3988 3976 3988 3976 3988 4000 4008 4016 4024 4032 4040 4048 4056	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3969 3977 3985 3993 4001 4009 4017 4025 4033 4041 4045	3842 3850 3858 3864 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3954 3954 3970 3978 3970 3978 3994 4002 4010 4018 4026 4034 4042 4058 4066	3843 3851 3859 3867 3875 3883 3891 3899 3907 3915 3923 3931 3939 3947 3955 3963 3971 3975 3963 3971 3975 3963 4003 4001 4019 4027 4035 4063	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3972 3980 3988 3996 4004 4012 4020 4028 4036 4044 4052 4068	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3965 3973 3965 3973 3981 3989 3997 4005 4013 4021 4029 4037 4045 4061	3846 3854 3854 3878 3878 3878 3894 3902 3910 3918 3926 3934 3926 3934 3958 3958 3958 3958 3958 3966 3974 3998 4006 4014 1022 4030 4038 4046 4052	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975 3987 3997 3997 4007 4015 4023 4031 4035 4047 4055 4063
to to 7777 4095	7010 7020 7030 7050 7050 7050 7050 7050 7050 705	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3672 3680 3688 3696 3704 3712 3720 3728 3736 3744 3752 3760 3768 3776 3784 3778 3784 3792 3800 3806 3816	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3665 3673 3681 3689 3697 3705 3713 3721 3725 3753 3761 3769 3777 3785 3793 3801 3809 3817	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3718 3746 3754 3754 3776 3778 3778 3778 3786 3794 3810 3818	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3663 3691 3699 3707 3715 3723 3731 3735 3747 3755 3763 3771 3779 3787 3795 3803 3811	3588 3596 3604 3612 3620 3628 3636 3644 3652 3664 3668 3676 3668 3700 3708 3716 3737 3748 3748 3748 3746 3748 3748 3748 3748 3748 3748 3748 3748	3589 3597 3605 3613 3621 3629 3637 3645 3665 3665 3665 3665 3665 3665 3665	3590 3598 3606 3614 3622 3630 3646 3654 3670 3678 3766 3774 3718 3726 3737 3758 3776 3778 3756 3774 3758 3774 3782 3790 3798 3798	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3703 3711 3719 3725 3743 3751 3759 3767 3755 3743 3751 3759 3767 3775 3783 3791 3799 3805 3823	7410 7420 7430 7440 7450 7450 7450 7510 7510 7520 7530 7530 7540 7550 7550 7570 7560 7570 7660 7670 7660 7670 7660 7670 7710 7720 7710 7720 7730 7740 7750	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3936 3944 3952 3960 3968 3976 3988 3976 3988 3976 3988 4000 4008 4016 4024 4032 4040 4048 4056 4072	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3929 3937 3945 3953 3961 3969 3977 3969 3977 3969 3977 3969 3977 3969 3977 4001 4009 4017 4025 4033 4041 4049 4057 4065 4073	3842 3850 3858 3864 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3954 3954 3970 3978 3986 3994 4002 4010 4018 4026 4034 4042 4050 4058 4056	3843 3851 3859 38675 3883 3891 3899 3907 3915 3923 3931 3939 3945 3953 3953 3963 3971 3975 3963 3971 3975 4003 4011 4019 4027 4035 4043 4051 4057	3844 3852 3860 3868 3876 3990 3908 3916 3924 3932 3940 3948 3956 3964 3972 3964 3972 3980 3988 3996 4004 4012 4020 4028 4036 4044 4052 4066 4076	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3933 3941 3949 3957 3955 3973 3965 3973 3981 3989 3997 4005 4013 4021 4029 4037 4045 4053 4061 4067	3846 3854 3862 3878 3886 3894 3902 3910 3918 3926 3934 3942 3950 3934 3950 3958 3966 3974 3950 3998 4006 4014 1022 4030 4038 4046 4054 4062 4078	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3975 3987 3997 3997 4007 4015 4023 4031 4035 4047 4055 4063 4071
to to 7777 4095	7010 7020 7030 7040 7050 7050 7050 7050 7050 7110 7110 711	3584 3592 3600 3608 3616 3624 3632 3640 3648 3656 3664 3656 3686 3696 3704 3712 3720 3728 3776 3728 3776 3768 3776 3776 3778 3778	3585 3593 3601 3609 3617 3625 3633 3641 3649 3657 3685 3673 3681 3689 3687 3705 3713 3721 3725 3753 3753 3753 3761 3769 3777 3785 3793 3801 3809 3817 3825	3586 3594 3602 3610 3618 3626 3634 3642 3650 3658 3666 3674 3682 3690 3698 3706 3714 3722 3730 3738 3746 3754 3754 3776 3778 3778 3778 3786 3794 3810 3818 3826	3587 3595 3603 3611 3619 3627 3635 3643 3651 3659 3667 3675 3683 3691 3699 3707 3715 3723 3731 3739 3747 3755 3763 3771 3779 3787 3795 3803 3811 3819 3827	3588 3596 3604 3612 3620 3612 3628 3636 3644 3652 3684 3676 3700 3708 3716 3772 3740 3776 3776 3776 3776 3776 3778 3778 3778	3589 3597 3605 3613 3621 3629 3637 3645 3661 3665 3661 3665 3665 3665 3665 366	3590 3598 3606 3614 3622 3630 3646 3646 3646 3670 3678 3766 3718 3726 3718 3726 3718 3758 3758 3758 3758 3758 3774 3758 3758 3758 3774 3758 3758 3758 3758 3758 3758 3758 3758	3591 3599 3607 3615 3623 3631 3639 3647 3655 3663 3671 3675 3687 3695 3703 3711 3719 3727 3735 3743 3751 3759 3767 3755 3767 3775 3783 3791 3799 3807 3815 3831	7410 7420 7430 7440 7450 7450 7450 7550 7550 7550 755	3840 3848 3856 3864 3872 3880 3888 3896 3904 3912 3920 3928 3936 3944 3952 3960 3944 3952 3960 3944 3952 3960 3976 3984 3976 3984 3992 4000 4008 4016 4024 4032 4040 4048 4056	3841 3849 3857 3865 3873 3881 3889 3897 3905 3913 3921 3921 3921 3921 3927 3945 3953 3961 3969 3977 3985 3993 4001 4009 4017 4025 4033 4041 4049 4057 4065 40731	3842 3850 3858 3866 3874 3882 3890 3898 3906 3914 3922 3930 3938 3946 3954 3954 3954 3954 3954 3970 3978 3986 3994 4002 4010 4018 4026 4034 4050 4058 4058 4058	3843 3851 3857 3875 3883 3891 3899 3907 3915 3923 3931 3932 3931 3939 3947 3955 3963 3971 3979 3987 3995 4003 4011 4019 4027 4035 4067 4051 4059	3844 3852 3860 3868 3876 3884 3892 3900 3908 3916 3924 3932 3940 3948 3956 3964 3956 3964 3972 3980 3988 3996 4004 4012 4020 4028 4036 4044 4052 4060 4068 4076	3845 3853 3861 3869 3877 3885 3893 3901 3909 3917 3925 3931 3949 3957 3965 3933 3941 3949 3957 3965 3973 3981 3989 3997 4005 4013 4021 4029 4037 4045	3846 3854 3862 3878 3878 3886 3894 3902 3910 3918 3926 3934 3926 3934 3942 3950 3958 3966 3934 3966 3974 3982 3990 3998 4004 1022 4030 4039 4038 4062 4054 4062 4075 4086	3847 3855 3863 3871 3879 3887 3895 3903 3911 3919 3927 3935 3942 3951 3959 3967 3959 3967 3975 3987 3991 3999 4007 4015 4023 4031 4039 4047 4055 4063 4071 4079

APPENDIX F POWERS OF TWO TABLE

POWERS OF TWO TABLE

				2 ⁿ	n	2 ⁻ⁿ													
				1	0	1.0													
				2	1	0.5													
			•	4 8	2 3	0.25 0.125													
					5														
				16 32	4 5	0.062													
				64	6	0.015													
				128	7	0.007		5											
				256	8	0.003	906	25											
				512	9	0.001													
				024	10	0.000													
			2	048	11	0.000	488	281	25										
				096	12	0.000	244	140	625										
				192	13	0.000													
				384 768	14	0.000													
			52	/00	15	0.000	030	517	578	125									
				536	16	0.000													
				072 144	17	0.000													
				288	18 19	0.000						5							
			048		20	0.000													
			097 194		21 22	0.000							c						
			388		22	0.000													
			777		24	0.000								-					
			554 108	•	25 26	0.000													
			217		27	0.000													
			435		28	0.000													
	1	073	870		29 30	0.000													
		147			31	0.000										5			
		20%	067	206	20														
		294 589			32 33	0.000 0.000													
		179			34	0.000											5		
		359			35	0.000													
	68	719	476	736	36	0.000	000	000	014	551	915	228	366	851	806	660	625		
	137	438	953	472	37	0.000	000	000	007	275	957	614	183	425	903	320	312	5	
		877			38	0.000	000	000	003	637	978	807	091	712	951	660	156	25	
	549	755	813	888	39	0.000	000	000	001	818	989	403	545	856	475	830	078	125	
1	099	511	627	776	40	0.000	000	000	000	909	494	701	772	928	237	915	020	062	5
															-37		557	002	2

APPENDIX G

PROGRAM EXAMPLES

				* *	OUTPUT A FIELD	D OF DATA IF FLAG IS SET
				*		
			9000		DRG	X190001
				OUTBUF	DS,200	+200 WORD OUTPUT RUFFER
			8020	DATA	DF	128,32 *MSB=128,32 BIT FIFLD
			0000	FLAG *	EQU	0 *BIT COLUMN ZERO
			8010	-	DRG	BULKC
	8010	8010	36600000		LI,2	AS, XICODOI +SELECT ARRAY #0 8 #1
	8711	8011	02008845		L	Y, FLAG +RESPONSE = WORDS 10 REAF
	8012	8012	34400008		LI	BL,200 +MAX BUFF SIZE COUNTER
	8013	8013	32809001		LI	DP,OUTBUF+1 +INDEX TO OUTBUF
	8914	8014	28E18018		BNR	DONE *EXIT IF NO FLAG SET
	8015	8015	03F86640	LOOP	STEP	*SELECT & CLEAR 1ST Y
	8016	8016	36000000		LC.	DATA *C REG. = VALUE OF SELECTED
	8017	8017	27C4A0FD		-	•
				*	WORD VIA THE L	INK POINTER REG.
	8018	8018	30050000		SR	C, (DP), 3 +STORE COMMON REGISTER
		-		*	INTO MEM ADDRE	ESS CONTAINED IN DP, THEN DP=DP+1, 8
				*	BL = BL-1	
	8019	8019	2901801B		BZ,BL	DONE *EXIT IF OUTPUT 200 HORDS
	801A	801A	28F18015		BRS	LOOP *LOOP IF MORE Y'S SET
W			30819000	DONE	SR	BL, OUTBUE *STORE COUNT IN 1ST
				*	SLOT. SEQ. PRO	DC. MUST CUMPLITE 200-COUNT FOR
				*		S IN THE OUTPUT BUFFER.
	801C	801C	38012000			12 *INTERRUPT SED. PROC.
			38402404		WAIT	*HALT AP

	*	THIS ROUTINE WIL TO ALL ZERDES.	L CLEAR TWO ARRAY MEMORIFS
8000	*	ORG	X ' 8000 '
	*	NOTE THAT WHENEV	ER THE AP GOES FROM THE INACTIVE
		STATE TO THE ACT LOCATION X'8000'	IVE STATE, THE INSTRUCTION AT IS EXECUTED.
8000 8000 28018010		8	PROGCLEAR
8010 8010 8010 3650C000 6		LI,2	BULKC AS,X'CMMM! +SELECT ARRAY #0 & #1
8011 8011 33900000 8012 8012 00007741			FP1,0 *START WITH BIT COLLMN 0 Y *CLEAR ALL Y
8013 8013 3EFF8014		RP1,256	*EXECUTE NEXT INSTRUCTION 256
1	* .		*TIMES
8014 8014 16300002		S OF MEMORY REFERE	Y,FP1+ *STORE Y INTO BIT COLUMN NCED BY FP1,THEN FP1 = FP1+1
	*	-	
8015 8015 38002000		WAIT	HALT AP

CHECK FOR MULTIPLE RESPONSE

*

. 🛨		
8010 8010 000088A2	L L	X,Y +SAVE RESPONSE IN X
8011 8011 03F86640	STEP	+CLEAR FIRST RESPONDER
8012 8012 28E18013	BNR	ONLYONE + BRANCH IF NO Y'S SET
•	-	
•	MULTTPLE RESPO	NSE PROCESSING FOLLOWS
-	•	
	•	
	•	
	•	
*		
*	SINGLE RESPONS	E PROCESSING FOLLOWS
•		
8013 8013 00008843 ONLY	DNE L	Y,X *RESTORE THE SINGLE RESPONDER
	•	
	•	

			*	EXAMPLE OF PAGE	NG ALTERNATING PROGRAM
			*	SEGMENTS INTO P	AGE 1 AND PAGE 2
			*		
		0000	*	ORG	XI80001 +BEGIN BY BOOTLOADING
0000	0040	8000 28018010		B	BULKC +PAGE 1 WITH THE INITIAL
8000	ONGN	8010		ORG	BULKC +PROGRAM SEGMENT
0010	8010	38008006		MVSG, PAGE1	PG1PROG +THIS INSTRUCTION WILL
		28008010	UULKU	HTOOT HOLI	
		380007F0			
		38048015			
04/40		0000000000	*	TEST THE PAGER	UNITL IT IS NOT BUSY, THEN WILL
			*	COMMAND THE PAG	ER TO MOVE A PROGAM SEGMENT
			*	INTO PAGE 1 ACC	ORDING TO THE SPECIFICATIONS
			*	GIVEN AT PG1PRO	G .
			*		
8014	8014	28010200		B	BEGIN *AP CONTROL WILL WAIT
			*	HERE UNTIL THE	PAGER HAS COMPLETED LOADING
			*	PAGE1 ACCORDING	TO THE ABOVE MOVE COMMAND.
			*		
			*		ARTING A DAGE A BROCDAN
8015	8015	C0060200	PG1PROG	STRTSG, PAGE1	*DEFINE A PAGE 1 PROGRAM
			*	SEGMENT ASSEMBL	ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION OF
			*		NING AT THE FIRST LOCATION OF
		30000000	*	OF PAGE 1	PG2PROG *BEGIN LOADING PAGE 2
		38008006 28000200	DEGIN	MV30/FAULZ	FOZERUG POLOIN LONDING FACE L
		38000BF0			
		38048020			
0.012	02.00	00040020			
			*	WHILE PROGRAM E	XECUTION OCCURS HERE IN PAGE 1
801A	0204	36000000	*		XECUTION OCCURS HERE IN PAGE 1 C,0 *CLEAR COMMON REGISTER
801A	0204	36000000	*	WHILE PROGRAM E LI	XECUTION OCCURS HERE IN PAGE 1 C,0 +CLEAR COMMON REGISTER
801A	0204	36000000	*		XECUTION OCCURS HERE IN PAGE 1 C,0 *CLEAR COMMON REGISTER
801A	0204	36000000	*		XECUTION OCCURS HERE IN PAGE 1 C,0 *CLEAR COMMON REGISTER
801A	0204	36000000	•		C,0 *CLEAR COMMON REGISTER
		36000000 28010400	•	LI B	C,0 *CLEAR COMMON REGISTER
			•	LI B FINISHED LOADIN	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH
			*	LI B FINISHED LOADIN IS ENCOUNTERED,	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B	0205	28910400	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B 801C	0205 801C	28010400 Nonoo20	* * *	LI B FINISHED LOADIN IS ENCOUNTERED,	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B 801C 801C	0205 801C 801D	28010400 00000020 00000420	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B 801C 801C 801D 801E	0205 801C 801D 801E	28010400 00000020 0000420 0000820	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B 801C 801C 801D 801E	0205 801C 801D 801E	28010400 00000020 00000420	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B 801C 801C 801D 801E	0205 801C 801D 801E	28010400 00000020 00000420 0000820	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING
801B 801C 801C 801D 801E	0205 801C 801D 801E	28010400 00000020 00000420 0000820	* * * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING.
801B 801C 801D 801E 801F	0205 801C 801D 801E 801F	28710400 00000420 0000420 0000820 000820 0008000	* * * * PG2PR0G	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING.
801B 801C 801D 801E 801F	0205 801C 801D 801E 801F	28010400 00000020 00000420 0000820	* * * * PG2PR0G	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY
801B 801C 801D 801E 801F	0205 801C 801D 801E 801F	28710400 00000420 0000420 0000820 000820 0008000	* * * PG2PROG *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING.
801B 801C 801D 801E 801F 8020	0205 801C 801D 801E 801F 8020	28710400 00000420 00000420 0000820 00008200 00008000	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION
801B 801C 801D 801E 801F 8020	0205 801C 801D 801E 801F 8020	28710400 00000420 00000420 0000820 00008200 00008000	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY
801B 801C 801D 801E 801F 8020 8021 8021	0205 801C 801D 801E 801F 8020 0400 0401	28010400 00000420 00000820 0000820 00008000 C0060400 38008006 28000400	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION
801B 801C 801D 801E 801F 8020 8021 8022 8023	0205 801C 801D 801E 801F 8020 0400 0401 0402	28010400 00000420 00000820 0000820 00008000 C0060400 38008006 28000400 38008006	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION
801B 801C 801D 801E 801F 8020 8021 8022 8023	0205 801C 801D 801E 801F 8020 0400 0401 0402	28010400 00000420 00000820 0000820 00008000 C0060400 38008006 28000400	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2 MVSG, PAGE1	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION NEXTPAGE1 *THIS INSTRUCTION
801B 801C 801D 801E 801F 8020 8021 8022 8023	0205 801C 801D 801E 801F 8020 0400 0401 0402	28010400 00000420 00000820 0000820 00008000 C0060400 38008006 28000400 38008006	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2 MVSG, PAGE1 WILL TEST THE P	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION NEXTPAGE1 *THIS INSTRUCTION AGER UNTIL IT IS NOT BUSY,
801B 801C 801D 801E 801F 8020 8021 8022 8023	0205 801C 801D 801E 801F 8020 0400 0401 0402	28010400 00000420 00000820 0000820 00008000 C0060400 38008006 28000400 38008006	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2 MVSG, PAGE1 WILL TEST THE P THEN WILL COMMA	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION NEXTPAGE1 *THIS INSTRUCTION AGER UNTIL IT IS NOT BUSY, ND THE PAGER TO MOVE A PROGRAM
801B 801C 801D 801E 801F 8020 8021 8022 8023	0205 801C 801D 801E 801F 8020 0400 0401 0402	28010400 00000420 00000820 0000820 00008000 C0060400 38008006 28000400 38008006	* * *	LI B FINISHED LOADIN IS ENCOUNTERED, IS COMPLETE BEF ENDSG STRTSG, PAGE2 SEGMENT ASSEMBL IF LOADED BEGIN OF PAGE 2 MVSG, PAGE1 WILL TEST THE P THEN WILL COMMA	C,0 *CLEAR COMMON REGISTER INTOPAGE2 *IF THE PAGER HASN'T G PAGE 2 BY THE TIME THIS BRANCH THE AP WILL WAIT UNTIL PAGING ORE BRANCHING. *DEFINE A PAGE 2 PROGRAM ED TO EXECUTE PROPERLY ONLY NING AT THE FIRST LOCATION NEXTPAGE1 *THIS INSTRUCTION AGER UNTIL IT IS NOT BUSY,

			*	SPECIFICATIONS AT NEXTPAGE1.
8025	0404	30800018	*	LRR C, (BL, DP) +COMMON = BL & DP
				•
8026	0105	28010200		B INTOPAGE1 +AP CONTROL WILL
0020	0400	20010200	*	WAIT HERE UNTIL THE PAGER HAS COMPLETED LOADING
				PAGE1 ACCORDING TO THE ABOVE MOVE COMMAND.
8027	8027	00000020		ENDSG
8028	8028	00000420		
		00000820		
8Ø2A	802A	00008000		
			*	
			*	
8028	8Ø2B	C0040200	NEXTPAGE1	STRTSG, PAGE1 +DEFINE A PAGE 1 PROGRAM
			*	SEGMENT ASSEMBLED TO EXECUTE PROPERLY ONLY
			*	IF LOADED BEGINNING AT THE FIRST LOCATION OF
			*	PAGE 1
			INTOPAGE1	MVSG,PAGE2 NEXTPAGE2 *THIS INSTRUCTION
		28000200		
		38000BF0		
802r	0203	38040000		WILL TEST THE PAGER UNTIL IT IS NOT BUSY,
				THEN WILL COMMAND THE PAGER TO MOVE A PROGRAM
			*	SEGMENT INTO PAGE 2 ACCORDING TO THE
			•	SPECIFICATIONS GIVEN AT NEXTPAGE2.FTC.
			-	di Loti Ichiteno otten di muxii docetetes
				•
				•
8030	8030	00000020		ENDSG
8031	8031	00000420		
		00000820		
8033	8033	00008000		
				e de la constante de
				•

G-5

.	* INSPECT	AND CHANGE A	N ARRAY FIELD	****
*	1			
*	USAGE DI	RECTIONS:		
*				
*	1. STORE	THE ARRAY N	UMBER INTO APR	AYNLIM
*			RAY MEMORY MOD	
*	•		MBER INTO WORD	
*	· · · · · · · · · · · · · · · · · · ·		GNIFICANT BIT	PUSITION
*		MSBPOSITN		0 7 1
*		IGNS INTO NU	OF CONTIGUOUS	011
*			PECT FUNCTION,	EXECUTE A
*			ING RØ) TO THE	
*			TO INSPECT. TH	
*			THE INSPECTION	
*	FROM	RESULT.		
*			IGE FUNCTION, S	
*	CHAN	SE INTO RESUL	T, THEN EXECUT	E A RETURN
*			TO THE BULK C	ORE ADDRESS
*	ASSIC	SNED TO CHANG	E.	
*				
*				
	START	א וווס	- ·	
8010	ORG	BULK	+ARRAY NUMBER	STODACE
8010 8010 0000000 AR			*ARRAY NUMBER	
8011 8011 00000000 WOF 8012 8012 00000000 MSE			*MSB OF FIELD	
8012 8012 80000000 MSt			*FIELD LENGTH	
8014 8014 00000000 PES	-		*THE ANSWER	
*				
*				
*				
***	***** STARTING	FPLACE FOR A	N INSPECT FUNC	TION
*				
•				
*				DECISTED
8015 8015 36000000 INS			+CLEAR COMMON ARRAYNUM +FP1	
8016 8016 33918010	LR,1		IF ARRAYNUM, SH	
*		ZERO OR ONE		
8017 8017 33418011	LR		WORDNUM +THE	LINK POINTER
4			IOW REFERENCES	
*		WORD WITHIN		
8018 8018 35A18012	LR,2		MSBPOSITN *FI	ELD POINTER,
*	LOAD FPT	S WITH LEAST	SIGNIFICANT 8	BIT POSITIONS
*	OF MSBPC	DSITN.		
8019 8019 35718013	LR,3	FL1,	NUMBITS *FIEL	D WIDTH COUNTER
*			SIGNIFICANT 8	BIT POSITIONS
*	OF NUMBI			
801A 801A 3660C000 LOC			100001 *SELEC	T BOTH ARRAYS
801B 801B 00007741	CLR		CLEAR ALL Y	
801C 801C 2821801F	BZ,FP1		RRAY	T OND ADDAY
801D 801D 36604000	LI,2		140001 *SELEC	I ZNU AKKAT
801E 801E 28018020		\$+2	190001 +00100	T IST ADEAV
801F 801F 36608000 1ST			180001 *SELEC 3 * Y = BIT C	
8020 8020 03808845	L USTNC TH		OGIC, GENERATE	
*	D2TNP 11	IC REQUERER L	UGILA GENERALE	

				*	INPUT WITH A ONE IN THE BIT POSITION REFERENCED
				*	BY FP2 AND 255 ZEROES ELSEWHERE, LOGICAL AND THIS
	8421	8021	03082240		GEN, 32 X'03C82240' *VALUE WITH Y, I.E.,
			n-cold in	•	RESET ALL Y'S EXCEPT IN THE SELECTED WORD
	0000	8000	28E18024		BNR \$+2 +JUMP IF SELECTED BIT = 0
			00008841		SET Y *SET ALL Y IF BIT = 1
			001F88B8		GEN,32 X'001F88B8' *LEFT SHIFT ARG ONE
	8025	8025	22FFAØFB		GEN,32 X'22FFA0FB' +BIT POSITION
	8026	8026	21C0BFFA		GEN,32 X'21CØBFFA' *LOAD BIT #31 OF
				*	COMMON REG. WITH SELECTED BIT NUMBER VALUE IN Y
W	8027	8027	01310001		INCR FP1,FL1 *FP3=FP3+1,8 FL1=FL1=1
			2891801A		BNZ, FL1 LOOP *LOOP ON REST OF FTELD WIDTH
			30018014		
	802A	802A	28080000		B (RØ) *RETURN TO CALLING PROGRAM
				*	
				🛨 1	
				******	STARTING PLACE FOR A CHANGE FUNCTION
				- -	
	9000	8000	36018014	CHANGE	LR C.RESULT *ARG = NEW VALUE
				CHANGE	
			36600000		LI,2 AS,X'COUN' *SELECT BOTH ARRAYS
			35718013		LR, 3 FL1, NUMBITS *OBTAIN FIELD WIDTH
	802E	802E	01010001		DECR FL1 *REPEAT COUNT FOR LOOP
	802F	802F	33900020		LI FP1,32 *PREPARE TO COMPUTE THE
	8030	8030	30008031		RPT +MSB POSITION ON THE FIELD IN
			01340001		DECR FP1 +THE COMMON REGISTER
			35718013		LR,3 FL1, NUMBITS +RELOAD FIELD WIDTH
			35A18012		LR,2 FP3,MSBPOSITN +LOAD ARRAY FIELD
	00000	00000	22-tokts		POINTER, FP1 IS THE COMMON REG. FIELD FOINTER.
				*	
W			31818043		SR FP1, SAVEFP1 +TEMP SAVE POINTER
		-	33418011		LR FP2,WORDNUM *INIT LINK POINTER
	8136	8036	33818043	REPEAT	LR FP1,SAVEFP1 *GET COM REG FIELD
				*	POINTER, NEXT LOAD THE X RS WITH THE VALUE OF THE
	8037	8037	0030B7A0		GEN,32 X1003087A01 *COMMON REG BIT
				•	REFERENCED BY FP1, THEN INCREMENT FP1
ш	80.48	8038	31818043	-	SR FP1, SAVEFP1 *TEMP SAVE POINTER
n			-		
			33918010		
			00007741		
	8038	803B	03C88840		GEN, 32 X'03C88840' +USING THE RESOLVER
				*	LOGIC, GENERATE A 256 BIT INPUT WITH A ONE IN THE
				*	BIT POSITION REFERENCED BY FP2 AND 255 ZEROES
				*	ELSEWHERE, LOAD THIS VALUE INTO Y, I.E., THE Y RS
				*	CONTAINS A ONE FOR ONLY THE REFERENCED WORD.
	8930	8030	08000002		L M,Y *SET UP FOR A MASKED WRITE
			00008843		$L \qquad Y, X + Y = COMMON REG(FP1) VALUE$
					-
			08400001		SM Y,FP3+ *STORE THE COMMON REG
	BUSE	ONSF	13400002		
				*	BIT VALUE REFERENCED BY FP1 INTO THE WORD OF
				*	MEMORY SPECIFIED BY FP2 AND THE BIT COLUMN IN
				*	THE SELECTED MEMORY WORD REFERENCED BY FP3,
				*	THEN INCREMENT FP3.
	8040	8040	01010001		DECR FL1 *DECREMENT FIELD WIDTH
			28918036		BNZ, FL1 REPEAT *STORE OTHER BITS
			28080000		· · ·
	8043	0043	0000000	SAVERP1	
			0000		END

G-7

А

Addressing	2-7
APPLE	1-2
APPLE features	1-2
APPLE language structure	2-1
Argument field	2 - 1
Arithmetic	2-138
Array co-ordinates	3-10
Assembler directives	1-2, 2-9
Assigning slots	3-1, -5
Associative instructions	2-55
Associative memory device code	3-8
Associative Memory or Common Register field expression	2-7
Attach (SVC)	3-5

в

Branch instructions	2 - 17
Buffer format	3-7, -9, -13, -19
Buffer format for device -1 (Control memory)	3-7
Buffer format for device -2 (Associative memory)	3-9
Buffer format for device -3 (Registers)	3-13
BUFFER pseudo-op	3-3, -6, -8, -12, -17
Byte count	3-9, -18, -19,
Byte count update	3-20

С

Character set	2-4
Command field	2 - 1
Command summary	A-i
Comment field	2-3
Comment statements	1-2
Common Register field	2-7
Constants	2-5
Control and test	2 - 154
Control Digit	2-7, C-2
Control Memory address	2-7, C-2
Control Memory device code	3-6

D

Decimal constants	2 - 5
Device Assignment Table (DAT)	3-2
Device codes	3-5
Devices	3-5
Done bit	3-10, -19

E

Echo Bit	3-20
End-of-medium bit (EOM)	3-9, -19
Error codes	B-i

Error codes (SVC)	3-10, -19
Execution location counter	2-6, C-2
Exit (SVC)	3-24
Expressions	2-6
F	
Field expression	2-7
Formatted ASCII	3-18
Formatted binary	3-18

н

Free device (SVC) for new task

Hexadecimal constants	2-5
Hexadecimal/decimal table	D-i

I

I setup	3-27
Immediate value	2-5, 2-39, C-4
Improper mode	3-9, -19
In line	1-2
Interrupt setup (SVC)	3-26, -27
Interrupt signal (SVC)	3-26, -27

3-23

L

Label field	2 - 1
Language elements	2-4
Load Location counter	2-6, C-5
Loads	2-55
Location counter symbol (\$)	2-6
Location counters	2-6

М

Max size	3-9, -17
Mnemonic summary	A-i
Mode byte	3-20
Moves	2 - 125

0

Octal constants	2-5
Octal/decimal	E-i
One-to-many	1-2
One-to-one	1-2

Ρ

Pager control (SVC)	3-28
Pager instructions	2 - 159
PI/O Control (SVC)	3-29
Powers of two table	F-i
Program Counter (location counter)	2-6, C-5
Program examples	G-i

R

Read (SVC)	3 - 15
READ BUFFER Pseudo-op	3-17
Register device code	3-13
Register instructions	2-35
Required entries	2-3
Reset peripheral device (SVC)	3-22
Restart program (SVC)	3-21

s

Searches	2-108
Slot numbers	3 - 1
Source statements	2-1
STARAN S registers (device code -3)	3 - 13
Status byte	3-7, -9, -19
Stores	2-79
Subroutine call sequence	1-2
Summary of APPLE mnemonics ans instruction formats	A-i
Supervisor calls (SVC)	3-1
Supervisor services	3-4
Symbol table	2-4
Symbols	2-4

. **T**

.

Terms and symbols

C-i

т

Timer (SVC)

U

Unformatted ASCII

Unformatted binary

w

Write (SVC)3-16WRITE BUFFER Pseudo-op3-17

3-18

3-18

3-25

.

•

.

.



GOODYEAR AEROSPACE