Alfaskop System 41

Technical Description

Text Section

E9000 3062E-3
1984-12-14

ERICSSON

Preface

This is the text section of the Technical Description for Alfaskop System 41. Its objective is to describe the main functions carried out by the logic of the display units, the flexible disk units and the communication processors as realized by the printed circuit boards.

Chapter 1 serves as general information on the system. Functions or circuits common for two or more units are described in chapter 2 and 3. The other chapters deal with specific units or logic boards.

Certain information is quoted from Motorola Inc. publications.

For information on printers and flexible disk drives, please refer to separate documents. Titles and Printed matter numbers can be found in the Service Information binder under Index of Current Literature.

Tips for the reader:

- A byte consists of eight bits if nothing else is said.
- Bit 0 always designates the least significant bit of a word. E.g. bit 7 (for data) or bit 15 (for addresses) is the most significant bit. Bit is often written b, e.g. b7 means bit 7.
- X or – in register descriptions normally means don't care bits.
- A pin of an integrated circuit or of a connector and sometimes a bit in a register is indicated as a number after a colon. E.g. pin 12 of IC25 will be written as 25:12, pin 2 of connector P1 could be indicated as P1:2.
- A bar over a signal name or a circle on a drawing of a wire or before a signal name indicates that low level is active (or negative logic for buses). E.g. °Bus addressing means that addressing from bus takes place when the signal °Bus addressing is low.

The specifications in this publication are subject to change and supplementation without notice.

It is possible that this manual contains information on Alfaskop products or functions no longer or not yet available in your country. This shall not be construed to mean that an announcement of such Alfaskop products or functions will be made in your country.

This is a revised edition of publication E9000 3062E-2. A new chapter Display Unit, DU 4113, has been added. The chapters 7, 9, 13, 15 – 16, 19 – 20 and 22 – 23 will be updated as soon as possible but until then, some of the references to the Microcomputer chapter are actually found in the Communication or Power Supplies chapters.

Data Terminals
Documentation
S-175 86 Järfälla, Sweden

ERICSSON

Reader comments

You can help improve future editions of this document by answering the following questions and sending us your comments.

1. **Document name and number:** _____

2. **My job:** _____

3. **I have used this document**
   - ☐ To acquire general information (for _____)
   - ☐ As a technical manual (reference book)
   - ☐ As a textbook in a course (trainee)
   - ☐ As source material for teaching a course (instructor)

   Comments: _____

4. **I think this document is**
   - ☐ Easy to find things in
   - ☐ Easy to find things in with some reservations
   - ☐ Difficult to find things in

   Comments: _____

5. **I think this document is**
   - ☐ Easy to understand
   - ☐ Easy to understand with some reservations
   - ☐ Difficult to understand

   Comments: _____

6. **I think this document is**
   - ☐ Well illustrated
   - ☐ Poorly illustrated

   Comments: _____

7. **I think that this document provides**
   - ☐ Full coverage of the subject at hand
   - ☐ Poor coverage (essential parts are lacking)

   Comments: _____

8. **I think that this document is**
   - ☐ Well adapted to my skills and knowledge
   - ☐ Poorly adapted to my skills and knowledge

   Comments: _____

9. **Other comments:** _____

We appreciate your cooperation.
Please send your answers and comments to

Ericsson Information Systems AB
Alfaskop
Documentation Department
S-175 86 Järfälla, Sweden

General Concepts

Contents

Introduction

This chapter contains a brief presentation of the hardware components of Alfaskop System 41. For detailed information on combination possibilities for different emulations (IBM 3270, UTS 400 etc.) look in the respective reference manual.

System Architecture

Some Configuration Examples

Fig. 1 shows a single display unit connected to a computer via modem and telephone lines (remote connection). Descriptions of modem interface logic boards are found in the SCA and ACA chapters.

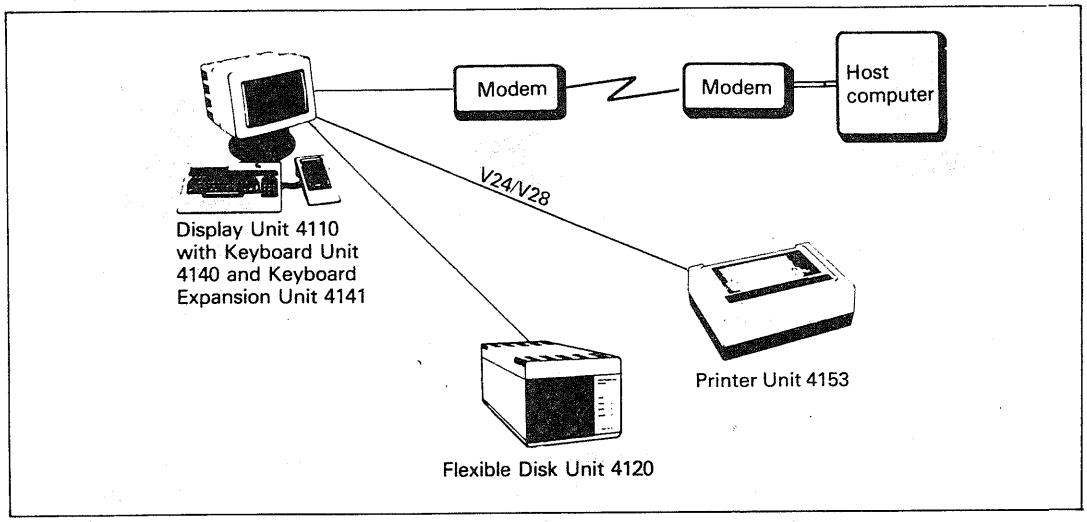


Fig. 1. Example of single display unit configuration

A flexible disk unit is connected to the display unit via a two-wire cable. A coarse description of the two-wire interface of the system units can be found in the Microcomputer chapter. A printer is connected to the display unit via a V24/V28 cable.

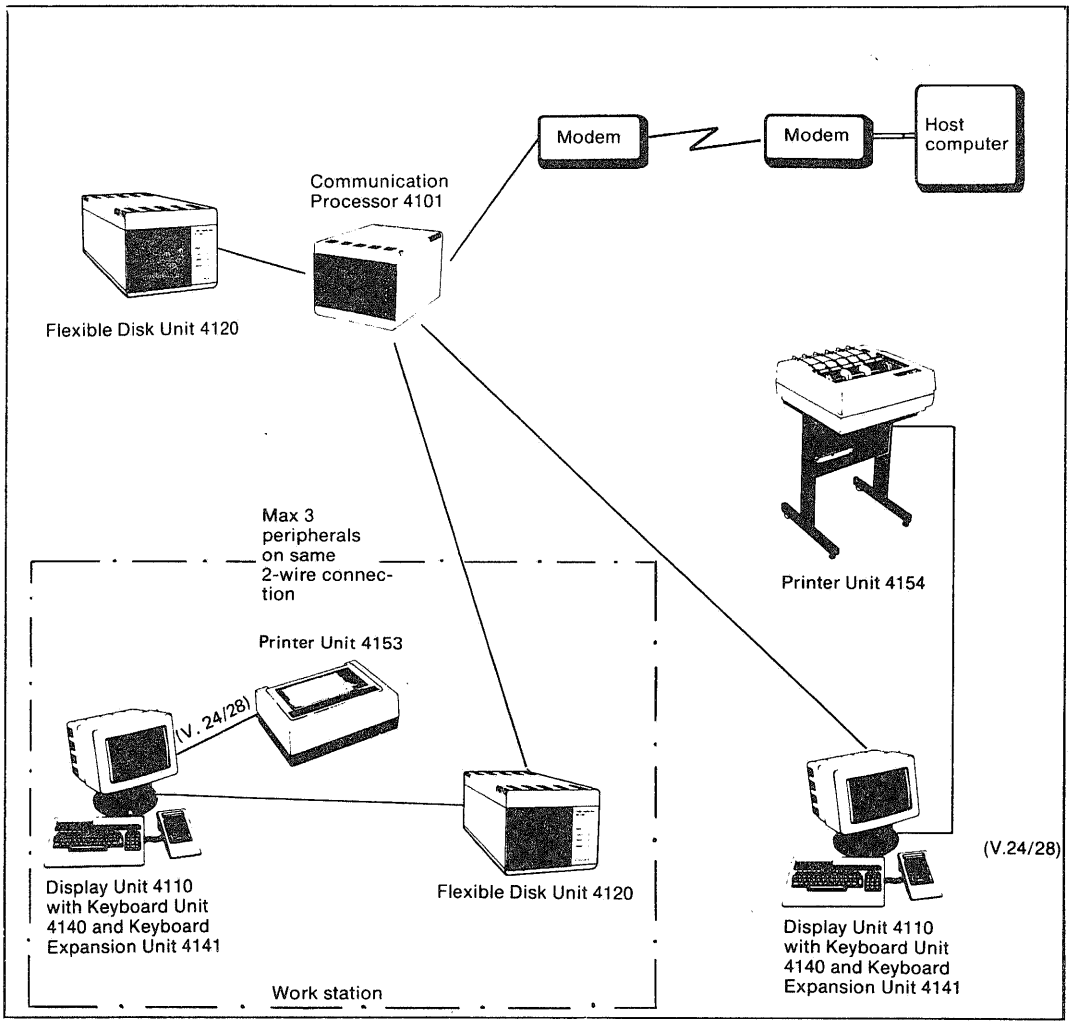


Fig. 2. Example of terminal cluster connected remotely to host computer

Fig. 2 shows an example of a cluster configuration (more than one terminal using the same line to communicate with a computer). The communication processor communicates with display units and flexible disk units via two-wire cables. The printers are connected to display units via V24/V28 cables.

Fig. 3 shows some of the components of units used in single configurations.

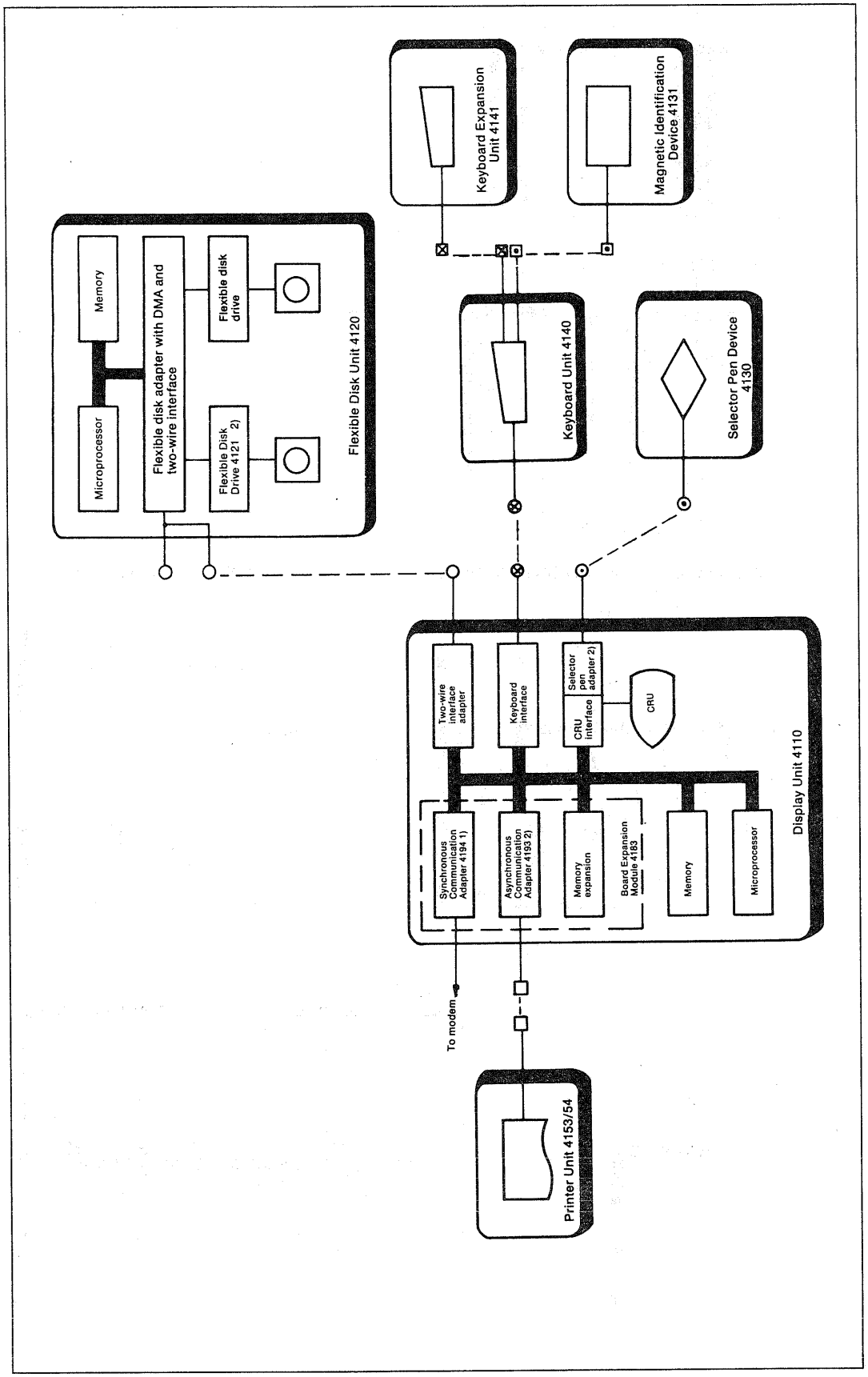


Fig. 3. Single display unit configuration details

Fig. 4 shows the components of units used in remote cluster configurations. Note that a communication processor for remote connection to a host computer (CPR) is shown.

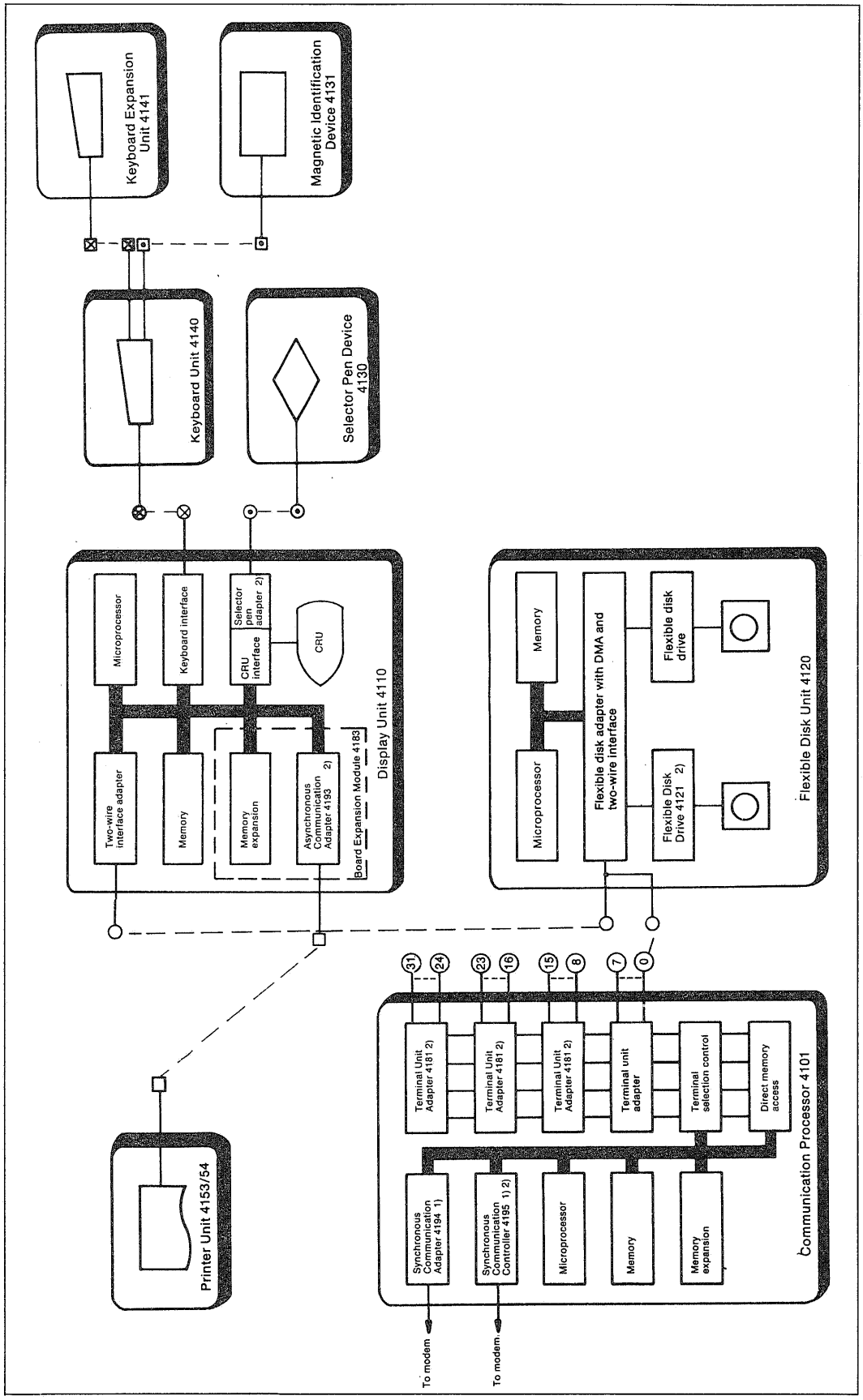


Fig. 4 Remote cluster configuration details

Fig. 5 shows the components of units used in local cluster configurations.

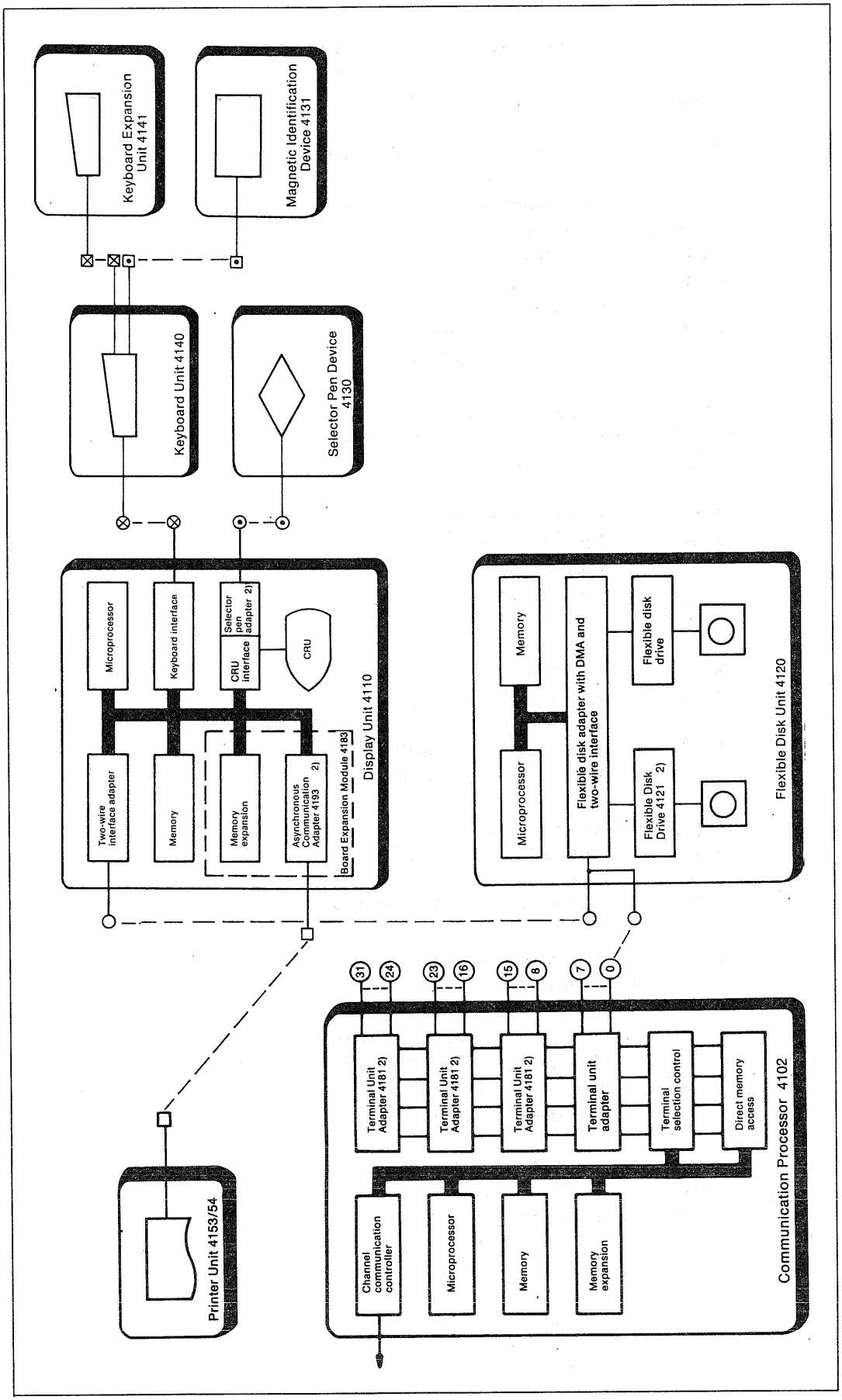


Fig. 5 Local cluster configuration details

Abbreviation List

ACA = Asynchronous communication adapter

BXM = Board expansion module

CCC = Channel communication controller
CIB = CCC interconnection board
CP = Communication processor
CPB = CP board
CPL = CP, local
CPR = CP, remote
CPS = CP power supply
CRU = Cathode ray tube unit
CTF = CP and terminal fan

DPS = Display terminal power supply
DTC = Display terminal controller
DU = Display unit

FD = Flexible disk unit
FDA = Flexible disk adapter
FDD = Flexible disk drive
FDP = Flexible disk processor
FPS = Flexible disk power supply

KB = Keyboard
KBC = Keyboard controller
KBU = Keyboard unit
KXB = Keyboard expansion board
KXU = Keyboard expansion unit

MIA = MID adapter
MID = Magnetic identification device
MRO = Memory board, RO memory
MRW = Memory board, R/W memory

PTC = Printer terminal controller
PU = Printer unit

SCA = Synchronous communication adapter
SCC = Synchronous communication controller
SPA = Selector pen adapter
SPD = Selector pen device

TAB = TUA interconnection board
TIA = Two-wire interface adapter
TUA = Terminal unit adapter

UPS = Universal power supply

Microcomputer

Contents

General

All main units of Alfaskop System 41 include a microcomputer. The purpose of this chapter is to describe the common attributes of the different M6800 family based microcomputers. Thus e.g. the most common LSI circuits of the M6800 family, the bus system, the addressing modes, the interrupt handling and the employed direct memory access method are described in this chapter. Even more detailed information on the M6800 family can be found in Motorola Semiconductors: Microcomputer components.

This chapter is directly applicable for the microcomputers in DU 4110, CPR 4101, CFU 4103 , and in the communication processor parts of CPL 4102. (The reader ought also to be familiar with this chapter before studying the chapter on FD 4120.) The main functional blocks of the microcomputer are shown in Fig. 1.

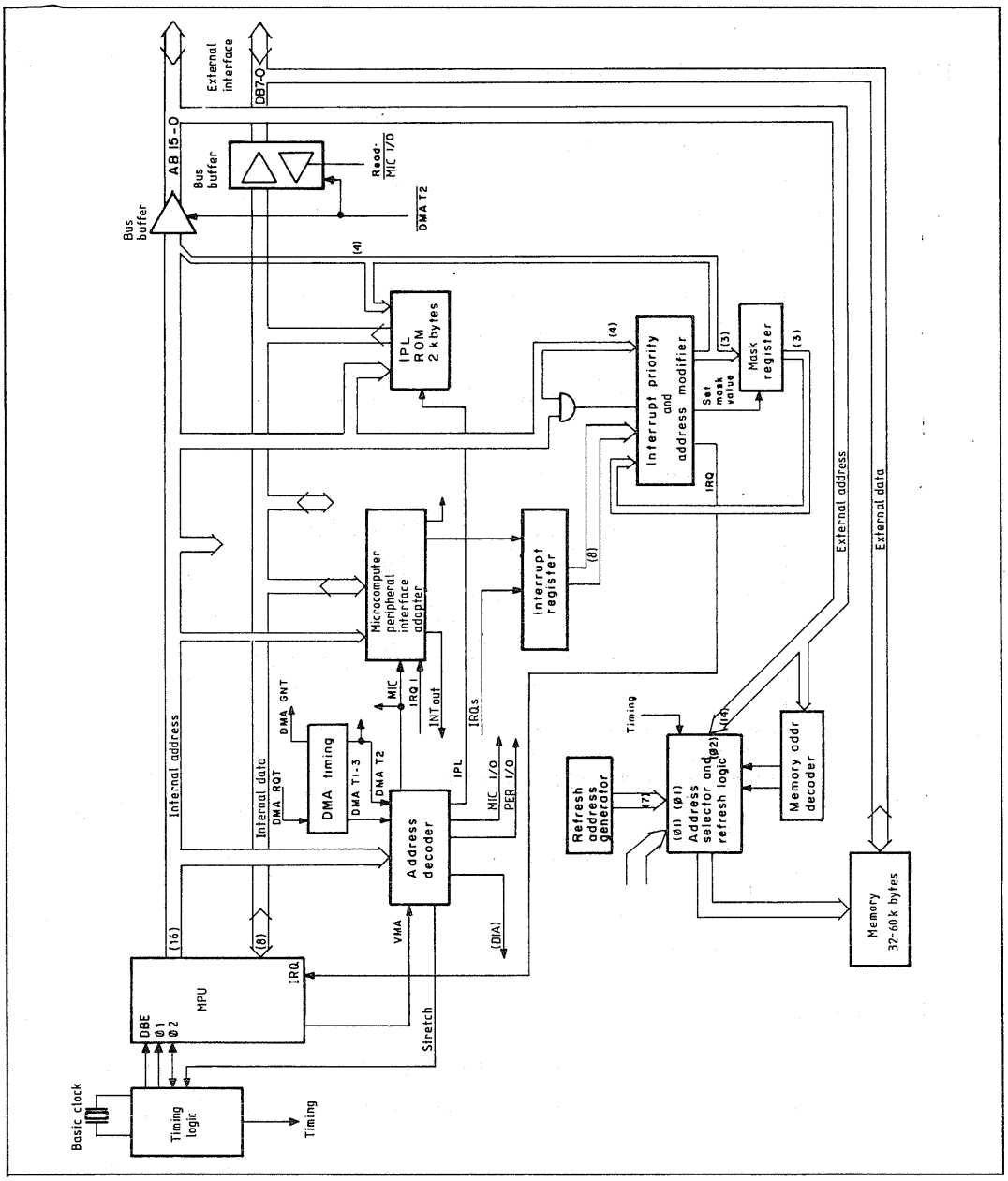


Fig. 1. Microcomputer, basic block diagram

These are:

- Microprocessing unit, MPU
- Timing logic
- Memory
- Address decode, bus timing and parts of memory access multiplexing logic
- Interrupt control logic
- IPL ROM, i.e. a read only memory containing the necessary program for initial program loading

The microprocessing unit used is of the M6800 type or similar, thus with an 8-bit data bus and a 16-bit address bus. No specific I/O instructions are used, but registers of peripheral circuits like parallel interface and serial interface adapters are addressed just as memory locations. The MPU is described in detail below.

Microprocessing Unit, MPU

In this paragraph the hardware and software attributes of the MPU are dealt with. The MPU is an 8-bit parallel three-state device. It has 16 address bits and is thus capable of addressing 65,536 memory locations. See Fig. 2.

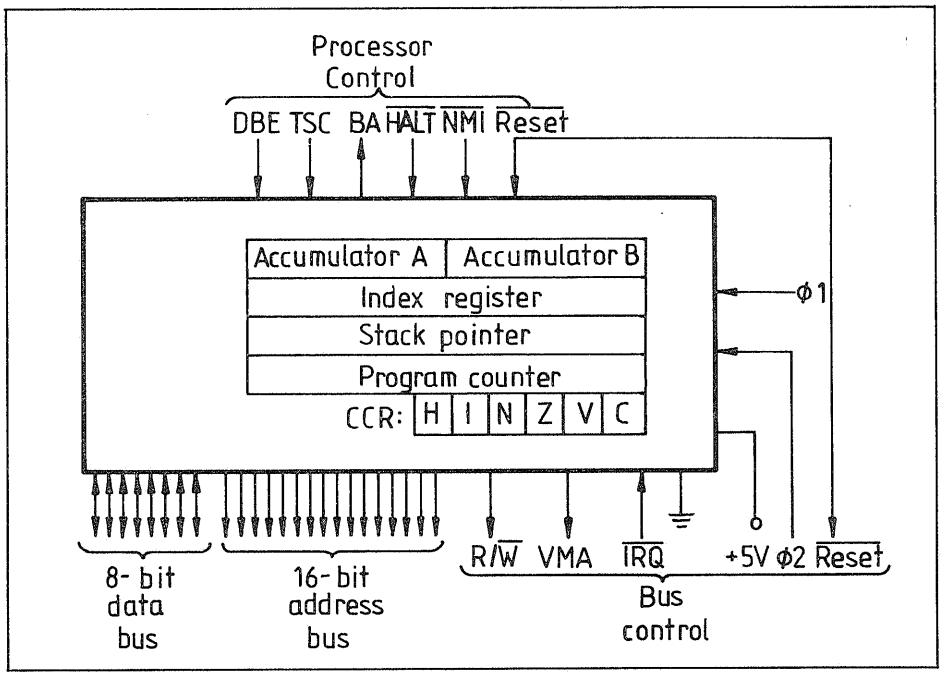


Fig. 2. MPU registers and signals

MPU Registers

Seen from the outside, the MPU contains six registers; three one byte registers and three double byte registers. The one byte registers are:

- Two accumulators named ACCA and ACCB used for storing operands for and results from operations.

- A condition code register, CCR, with 6 flag bits (b7 and b6 are always ones) signalling the result of a previous operation. The meaning of the different CCR bits, when set, are:

b5	H	A preceding operation (e.g. ADD, SUB, CMP) resulted in a half-carry from bit 3 to bit 4.
4	I	When this bit is reset, the MPU will service a maskable interrupt request (IRQ). It will be set when an interrupt occurs, thus hindering or "masking" further interrupts (IRQs).
3	N	The result of a preceding operation is negative (bit 7 of result = 1).
2	Z	The result of a preceding operation = 0.
1	V	A previous operation resulted in 2's complement overflow, i.e. a limit of the number area [− 128, 127] was passed.
0	C	A carry from bit 7 was produced by a preceding operation or a borrow to bit 7 was needed to make the operation (SUB or CMP).

  Detailed information on which CCR bits that are affected by a certain instruction is found in Fig. 4.

The 16-bit double registers, mostly used for addressing, are:

- Index register, which reduces program memory requirements as it can be loaded with a different memory address from the one contained in the program counter and stack pointer (see addressing modes below)
- Stack pointer, which should be initialized to point to the highest address of the read/write memory area to which the stack function has been assigned. At a Push data instruction (PSH) the contents of one of the accumulators (A or B) is stored in the stack and the stack pointer is automatically decremented. When data from the stack is wanted back to an accumulator a Pull data instruction (PUL) is used, resulting in an automatic incrementation of the stack pointer and a following loading of an accumulator from the memory cell pointed out by the stack pointer. The stack thus functions as a LIFO (last-in, first-out memory). The stack pointer is also automatically used at interrupts and at Jump to subroutine (JSR) and Branch to subroutine (BSR) instructions. In the interrupt case the contents of the MPU registers (but the stack pointer) are pushed into the stack and brought back at the Return from interrupt instruction (RTI). At JSR and BSR only the value of the program counter (i.e. the return address) is stacked away and brought back at the Return from subroutine instruction (RTS).

  Both the index register and the stack pointer contents can be loaded from or stored into two consecutive memory cells using one instruction.
- Program counter. At reset and different interrupts, this counter is loaded with program addresses (interrupt vectors) stored at specific memory locations in the highest part of the memory area. The program counter is then automatically stepped through the program and also changed at jump or branch instructions. Otherwise it cannot be written or read.

MPU Signals

Apart from the 16 address bus lines and the 8 bidirectional data bus lines there are several other timing and control signals in the system. The clock inputs to the MPU are:

- Ø1 and Ø2, complementary, non-overlapping clocks. The cycle time mentioned in this text is the clock cycle time. During Ø1 high time the address is set up and during Ø2 high time data is set up and read or written. (On Ø2 going low.) See Timing paragraph for details.

Normal program execution can be stopped by four external signals, namely Interrupt request (IRQ), Non-maskable interrupt (NMI), Reset (R) and HALT.

- IRQ input being low will make the processor jump to an interrupt routine if the interrupt mask bit (I) in the MPU condition code register is reset.
- NMI input going low will always make the MPU jump to an interrupt routine after the present instruction has been completed.

  An incoming interrupt (IRQ, NMI or Software interrupt instruction) or Reset will set the I-bit to prevent further interrupts (IRQs) from interfering with the execution of the interrupt routine.
- Reset being low will make the processor stop (if it has started). When Reset goes high again the MPU will fetch the address of the restart routine from the two highest memory positions ($FFFE_{(16)}$ and $FFFF_{(16)}$), load this address into the program counter and begin the restart sequence. The interrupt mask bit in the CCR will be set during Reset and must be cleared by the program if service of an IRQ is wanted.
- HALT being low will stop all program execution. Interrupts coming during HALT condition will be preserved and serviced after HALT has gone high again. If Reset is low during HALT condition, the restart routine will be entered when HALT goes high again. The MPU will signal that it is halted with a BA signal (see below).
- BA, Bus available, is high when the MPU is halted. BA is also high when the MPU is waiting for an interrupt as a result of a Wait for interrupt instruction. The address and the data bus as well as the R/W signal (see below) are then in high impedance state.
- R/W, Read/Write, signals the peripherals and memories whether the MPU is in read (high) or write (low) state. R/W is in high impedance state when the MPU is halted and its buses are available for another device (e.g. another MPU) to access the memories of the system (DMA, direct memory access).
- VMA, Valid memory address, is low or non-active during HALT. This signal is active only during the cycles when the memory address from the MPU should select a specific circuit.
- TSC, Three-state control, being high puts the R/W and address bus lines into high impedance state. The clocks for the MPU, Ø1 and Ø2, should be stopped to prevent false program execution. Ø1 should be held high.
- DBE, Data bus enable, which is generally tied to Ø2, low, puts the data bus input/output in high impedance state (allowing DMA).

The drawing symbol is shown in Fig. 3.

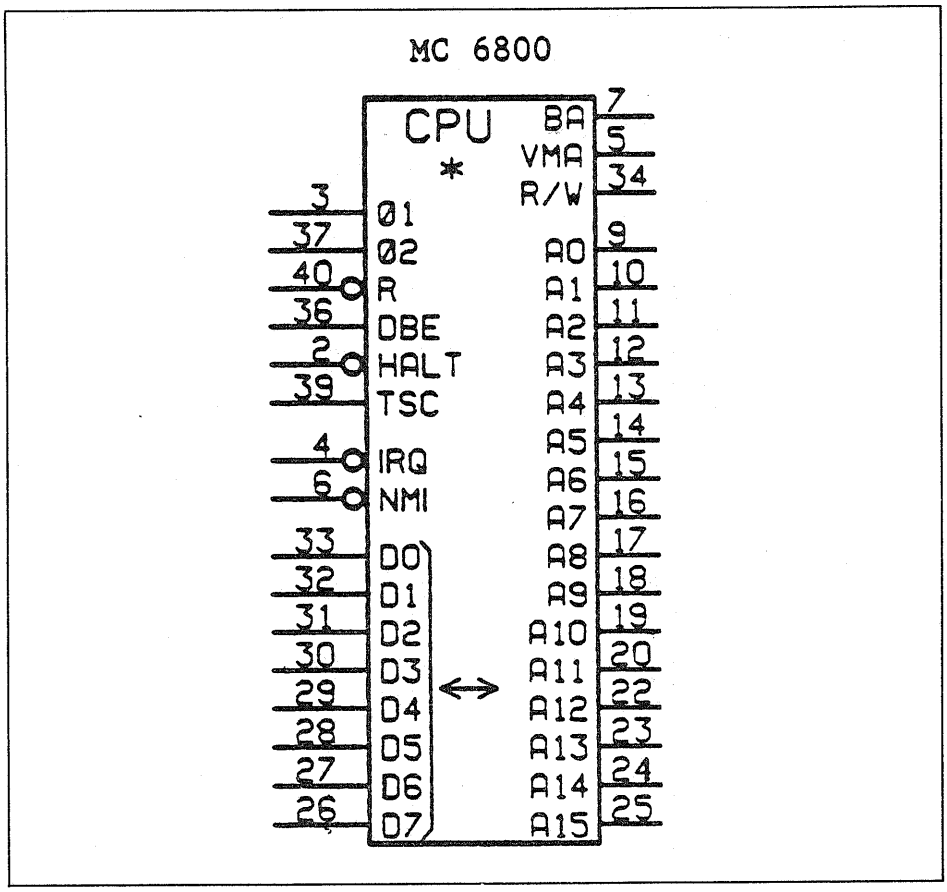


Fig. 3. CPU drawing symbol

Instruction Set and Addressing Modes

The instruction set is shown in Fig. 4. The seven addressing modes mentioned in the headings of the instruction set are described below:

- Immediate addressing or zero-level addressing. In this case one operand is found in the memory cell (two cells for index register and stack manipulations) immediately following the operation code location.
- Direct and extended addressing or one-level addressing. In these modes the operand or the operation code of the next instruction at a jump instruction is found in the cell pointed out by the byte (for direct addressing) or 2 bytes (for extended addressing) following the operation code. Thus, with extended addressing it is possible to reach cells $0000_{(16)}$ through $FFFF_{(16)}$ but with direct addressing only cells 0 through 255 in the memory map.
- Indexed addressing. In this case the effective address of the operand location is formed by adding an offset from 0 to 255 (contained in the second byte of the instruction) to the 16-bit address contained in the index register.
- Relative addressing. Here, the meaning of relative addressing (used for the branch instructions) is restricted to the case when the address to the next instruction is relative to the address of the cell following the two branch instruction bytes. Note that, as the relative address is expressed in 2's complement, it is a number between − 128 and + 127.

ACCUMULATOR AND MEMORY INSTRUCTIONS

		ADDRESSING MODES															BOOLEAN/ARITHMETIC OPERATION	COND. CODE REG.					
		IMMED			DIRECT			INDEX			EXTND			IMPLIED				5	4	3	2	1	0
OPERATIONS	MNEMONIC	OP	~	=	OP	~	=	OP	~	=	OP	~	=	OP	~	=	(All register labels refer to contents)	H	I	N	Z	V	C
Add	ADDA	8B	2	2	9B	3	2	AB	5	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	5	2	FB	4	3				B + M → B	↕	•	↕	↕	↕	↕
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add with Carry	ADCA	89	2	2	99	3	2	A9	5	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	5	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	5	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	5	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Bit Test	BITA	85	2	2	95	3	2	A5	5	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	5	2	F5	4	3				B · M	•	•	↕	↕	R	•
Clear	CLR							6F	7	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	5	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	5	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
Compare Acmltrs	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Complement, 1's	COM							63	7	2	73	6	3				$\overline{M}$ → M	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A}$ → A	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B}$ → B	•	•	↕	↕	R	S
Complement, 2's (Negate)	NEG							60	7	2	70	6	3				00 − M → M	•	•	↕	↕	①	②
	NEGA													40	2	1	00 − A → A	•	•	↕	↕	①	②
	NEGB													50	2	1	00 − B → B	•	•	↕	↕	①	②
Decimal Adjust, A	DAA													19	2	1	Converts Binary Add. of BCD Characters into BCD Format	•	•	↕	↕	↕	③
Decrement	DEC							6A	7	2	7A	6	3				M − 1 → M	•	•	↕	↕	4	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	4	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	4	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	5	2	B8	4	3				A⊕M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	5	2	F8	4	3				B⊕M → B	•	•	↕	↕	R	•
Increment	INC							6C	7	2	7C	6	3				M + 1 → M	•	•	↕	↕	⑤	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	⑤	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	⑤	•
Load Acmltr	LDAA	86	2	2	96	3	2	A6	5	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	5	2	F6	4	3				M → B	•	•	↕	↕	R	•
Or, Inclusive	ORAA	8A	2	2	9A	3	2	AA	5	2	BA	4	3				A + M → A	•	•	↕	↕	R	•
	ORAB	CA	2	2	DA	3	2	EA	5	2	FA	4	3				B + M → B	•	•	↕	↕	R	•
Push Data	PSHA													36	4	1	A → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
	PSHB													37	4	1	B → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULA													32	4	1	SP + 1 → SP, M_{SP} → A	•	•	•	•	•	•
	PULB													33	4	1	SP + 1 → SP, M_{SP} → B	•	•	•	•	•	•
Rotate Left	ROL							69	7	2	79	6	3				M	•	•	↕	↕	⑥	↕
	ROLA													49	2	1	A	•	•	↕	↕	⑥	↕
	ROLB													59	2	1	B	•	•	↕	↕	⑥	↕
Rotate Right	ROR							66	7	2	76	6	3				M	•	•	↕	↕	⑥	↕
	RORA													46	2	1	A	•	•	↕	↕	⑥	↕
	RORB													56	2	1	B	•	•	↕	↕	⑥	↕
Shift Left, Arithmetic	ASL							68	7	2	78	6	3				M	•	•	↕	↕	⑥	↕
	ASLA													48	2	1	A	•	•	↕	↕	⑥	↕
	ASLB													58	2	1	B	•	•	↕	↕	⑥	↕
Shift Right, Arithmetic	ASR							67	7	2	77	6	3				M	•	•	↕	↕	⑥	↕
	ASRA													47	2	1	A	•	•	↕	↕	⑥	↕
	ASRB													57	2	1	B	•	•	↕	↕	⑥	↕
Shift Right, Logic	LSR							64	7	2	74	6	3				M	•	•	R	↕	⑥	↕
	LSRA													44	2	1	A	•	•	R	↕	⑥	↕
	LSRB													54	2	1	B	•	•	R	↕	⑥	↕
Store Acmltr.	STAA				97	4	2	A7	6	2	B7	5	3				A → M	•	•	↕	↕	R	•
	STAB				D7	4	2	E7	6	2	F7	5	3				B → M	•	•	↕	↕	R	•
Subtract	SUBA	80	2	2	90	3	2	A0	5	2	B0	4	3				A − M → A	•	•	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	5	2	F0	4	3				B − M → B	•	•	↕	↕	↕	↕
Subtract Acmltrs.	SBA													10	2	1	A − B → A	•	•	↕	↕	↕	↕
Subtr. with Carry	SBCA	82	2	2	92	3	2	A2	5	2	B2	4	3				A − M − C → A	•	•	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	5	2	F2	4	3				B − M − C → B	•	•	↕	↕	↕	↕
Transfer Acmltrs	TAB													16	2	1	A → B	•	•	↕	↕	R	•
	TBA													17	2	1	B → A	•	•	↕	↕	R	•
Test, Zero or Minus	TST							6D	7	2	7D	6	3				M − 00	•	•	↕	↕	R	R
	TSTA													4D	2	1	A − 00	•	•	↕	↕	R	R
	TSTB													5D	2	1	B − 00	•	•	↕	↕	R	R
																		H	I	N	Z	V	C

LEGEND:

OP	Operation Code (Hexadecimal);	+	Boolean Inclusive OR;
~	Number of MPU Cycles;	⊕	Boolean Exclusive OR;
=	Number of Program Bytes;	$\overline{M}$	Complement of M;
+	Arithmetic Plus;	→	Transfer Into;
−	Arithmetic Minus;	0	Bit = Zero;
·	Boolean AND;	00	Byte = Zero;
M_{SP}	Contents of memory location pointed to be Stack Pointer;		

Note – Accumulator addressing mode instructions are included in the column for IMPLIED addressing

CONDITION CODE SYMBOLS:

H	Half-carry from bit 3;
I	Interrupt mask
N	Negative (sign bit)
Z	Zero (byte)
V	Overflow, 2's complement
C	Carry from bit 7
R	Reset Always
S	Set Always
↕	Test and set if true, cleared otherwise
•	Not Affected

Fig. 4a Instruction set

INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

POINTER OPERATIONS	MNEMONIC	IMMED OP	IMMED ~	IMMED =	DIRECT OP	DIRECT ~	DIRECT =	INDEX OP	INDEX ~	INDEX =	EXTND OP	EXTND ~	EXTND =	IMPLIED OP	IMPLIED ~	IMPLIED =	BOOLEAN/ARITHMETIC OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Reg	CPX	8C	3	3	9C	4	2	AC	6	2	BC	5	3				$X_H - M, X_L - (M + 1)$	•	•	⑦	↕	⑧	•
Decrement Index Reg	DEX													09	4	1	$X - 1 \rightarrow X$	•	•	•	↕	•	•
Decrement Stack Pntr	DES													34	4	1	$SP - 1 \rightarrow SP$	•	•	•	•	•	•
Increment Index Reg	INX													08	4	1	$X + 1 \rightarrow X$	•	•	•	↕	•	•
Increment Stack Pntr	INS													31	4	1	$SP + 1 \rightarrow SP$	•	•	•	•	•	•
Load Index Reg	LDX	CE	3	3	DE	4	2	EE	6	2	FE	5	3				$M \rightarrow X_H, (M + 1) \rightarrow X_L$	•	•	⑨	↕	R	•
Load Stack Pntr	LDS	8E	3	3	9E	4	2	AE	6	2	BE	5	3				$M \rightarrow SP_H, (M + 1) \rightarrow SP_L$	•	•	⑨	↕	R	•
Store Index Reg	STX				DF	5	2	EF	7	2	FF	6	3				$X_H \rightarrow M, X_L \rightarrow (M + 1)$	•	•	⑨	↕	R	•
Store Stack Pntr	STS				9F	5	2	AF	7	2	BF	6	3				$SP_H \rightarrow M, SP_L \rightarrow (M + 1)$	•	•	⑨	↕	R	•
Indx Reg → Stack Pntr	TXS													35	4	1	$X - 1 \rightarrow SP$	•	•	•	•	•	•
Stack Pntr → Indx Reg	TSX													30	4	1	$SP + 1 \rightarrow X$	•	•	•	•	•	•

JUMP AND BRANCH INSTRUCTIONS

OPERATIONS	MNEMONIC	RELATIVE OP	RELATIVE ~	RELATIVE #	INDEX OP	INDEX ~	INDEX #	EXTND OP	EXTND ~	EXTND #	IMPLIED OP	IMPLIED ~	IMPLIED #	BRANCH TEST	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Branch Always	BRA	20	4	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC	24	4	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS	25	4	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ	27	4	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE	2C	4	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT	2E	4	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI	22	4	2										C + Z = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE	2F	4	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS	23	4	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT	2D	4	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI	2B	4	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE	26	4	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC	28	4	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS	29	4	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL	2A	4	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR	8D	8	2											•	•	•	•	•	•
Jump	JMP				6E	4	2	7E	3	3					•	•	•	•	•	•
Jump To Subroutine	JSR				AD	8	2	BD	9	3					•	•	•	•	•	•
No Operation	NOP										01	2	1	Advances Prog. Cntr. Only	•	•	•	•	•	•
Return From Interrupt	RTI										3B	10	1		⑩					
Return From Subroutine	RTS										39	5	1		•	•	•	•	•	•
Software Interrupt	SWI										3F	12	1		•	•	•	•	•	•
Wait for Interrupt*	WAI										3E	9	1		•	⑪	•	•	•	•

*WAI puts Address Bus, R/W, and Data Bus in the three-state mode while VMA is held low.

CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

OPERATIONS	MNEMONIC	IMPLIED OP	IMPLIED ~	IMPLIED =	BOOLEAN OPERATION	COND. CODE REG. 5 H	4 I	3 N	2 Z	1 V	0 C
Clear Carry	CLC	0C	2	1	$0 \rightarrow C$	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	$0 \rightarrow I$	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	$0 \rightarrow V$	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	$1 \rightarrow C$	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	$1 \rightarrow I$	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	$1 \rightarrow V$	•	•	•	•	S	•
Acmltr A → CCR	TAP	06	2	1	$A \rightarrow CCR$	⑫					
CCR → Acmltr A	TPA	07	2	1	$CCR \rightarrow A$	•	•	•	•	•	•

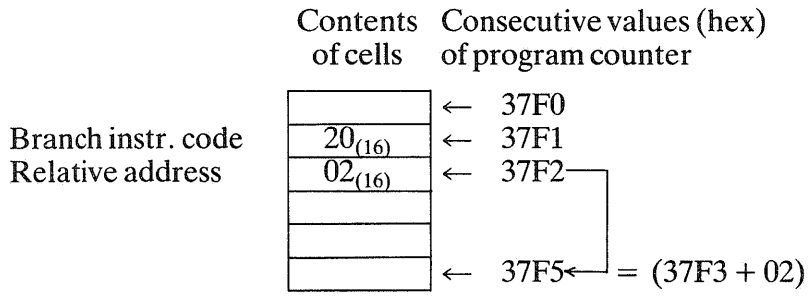
CONDITION CODE REGISTER NOTES: (Bit set if test is true and cleared otherwise)

1 (Bit V) Test: Result = 10000000?
2 (Bit C) Test: Result = 00000000?
3 (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set.)
4 (Bit V) Test: Operand = 10000000 prior to execution?
5 (Bit V) Test: Operand = 01111111 prior to execution?
6 (Bit V) Test: Set equal to result of N⊕C after shift has occurred.
7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?
8 (Bit V) Test: 2's complement overflow from subtraction of MS bytes?
9 (Bit N) Test: Result less than zero? (Bit 15 = 1)
10 (All) Load Condition Code Register from Stack. (See Special Operations)
11 (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
12 (All) Set according to the contents of Accumulator A.

MOTOROLA Semiconductor Products Inc.

Fig. 4b Instruction set

Example:



- Implied and accumulator addressing. The instructions using these addressing modes are one byte instructions, where the operand or operands are understood from or contained in the operation code. Mostly these are instructions affecting only one or two MPU registers. For the Push data (PSH), Pull data (PUL), return (RTI, RTS), Software interrupt (SWI), and Wait for interrupt (WAI) instructions, however, memory locations indicated by the stack pointer are also affected. The No operation (NOP) instruction is special in that it only advances the program counter one step and introduces a delay of two MPU cycles before active program execution is resumed.

Interrupt Handling

Please refer to Fig. 1.

General

The MPU only reacts to four interrupts (Reset, Non-maskable interrupt, Software interrupt and Interrupt request, see Fig. 5). More interrupts are wanted and an interrupt logic is therefore contained in the microcomputer. This logic permits eight interrupts to use the IRQ line to the MPU.

In addition to the possibility to mask all IRQs by setting the I-bit of the MPU condition code register (CCR), the MPU can set a value between 0 and 7 in a mask register to prevent IRQs below the handeled level from interrupting the program execution. IRQ 7 is the highest priority interrupt and IRQ 0 the lowest. Thus, if e.g. the mask register is set to 5, only interrupts on level 5, 6 or 7 will activate the IRQ line to the MPU.

The MPU reacts to an IRQ (if the I-bit is reset) by:

- Completing the execution of the present instruction and stepping the program counter to point to the next instruction
- Saving the contents of its registers in the stack (contents of program counter = return address, etc)
- Trying to fetch the address to the interrupt routine from the memory cells $FFF8_{(16)}$ and $FFF9_{(16)}$
- Setting the interrupt mask bit of the CCR to disable further IRQs.

When returning from an interrupt routine (by the RTI instruction) the registers of the MPU get their old contents back from the stack. Thus the execution of the interrupted program is resumed. Note that the I-bit value is 0, i.e. not changed when stored in the stack.

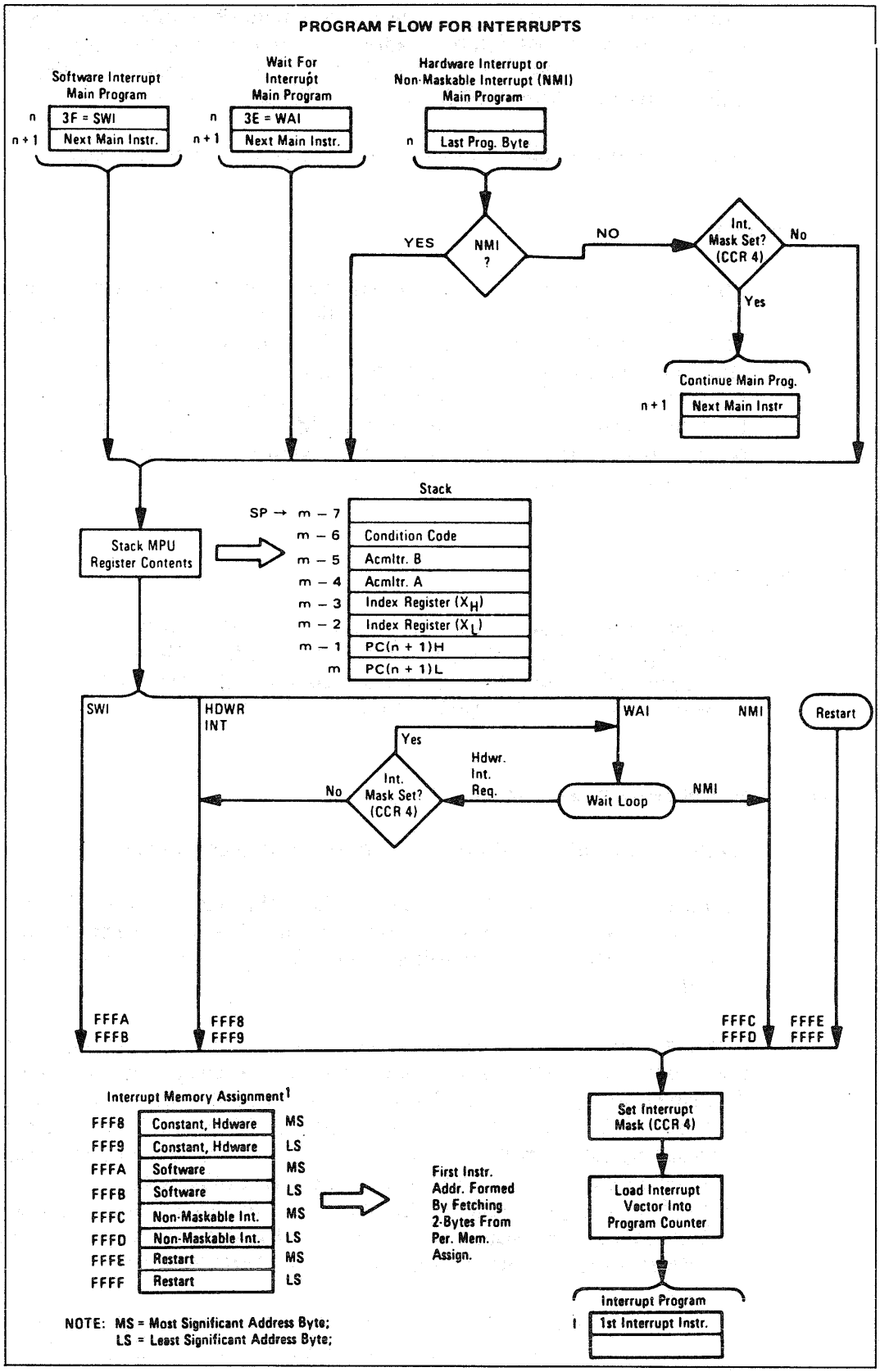


Fig. 5. Interrupt sequence

Address Modifier

When the MPU addresses cells $FFF8_{(16)}$ and $FFF9_{(16)}$, an address modifier will change the addresses to a pair of addresses in the area $FFE8_{(16)}$ – $FFF7_{(16)}$ depending on the highest priority valid interrupt in an interrupt register. That is to say, if the interrupt register still holds an interrupt with a level higher than or same as the actual mask level when $FFF8_{(16)}$ is applied, the MPU will fetch the address to that specific interrupt routine when addressing cells $FFF8_{(16)}$ and $FFF9_{(16)}$. Note that address modification only occurs for addresses $FFF8_{(16)}$ and $FFF9_{(16)}$ *from the MPU.*

See Fig. 10 for detailed information on interrupt vector addresses.

Mask Register

The mask register is set to an associated value if a specific vector address in the area $FFE8_{(16)}$ to $FFF7_{(16)}$ is applied on the address bus from the MPU. If the MPU e.g. reads cell $FFE8_{(16)}$, the mask will be set to zero and all interrupts will be permitted in case the MPU CCR I-bit is reset. No address modification will take place; the read operation will fetch the address to interrupt routine 0 to the MPU from the IPL ROM. If no change of the MPU register contents is wanted when setting the mask value, a write operation should be performed.

Interrupt Register

The interrupt register latches the state of the IRQs at the beginning of each Ø2 period except when cells FFF8, 9, A, or $B_{(16)}$ (IRQ or SWI vector) are addressed by the MPU. The reason for not latching is that the modified address lines should be stable when the MPU fetches a vector.

Generation and Direction of IRQs

As mentioned above, there are eight possible incoming IRQs.

Several IRQs originate from sources where the IRQ is stable. The reset of such an IRQ, which has to be done before clearing the MPU I-bit, is generally performed by reading a specific register of the interrupting circuit.

Some interrupts may originate from circuits that only provide a short pulse that could disappear before the program is ready to be interrupted. To prevent this, at least one IRQ is coupled to the interrupt register via a latch. This latch may be reset by resetting a bit of a MIC PIA register (microcomputer peripheral interface adapter), but is unconditionally set when an interrupt occurs. Thus, if an interrupt line with a latch is used for stable interrupts, the MIC PIA bit should always be reset. Otherwise the IRQ in question would have to be reset by both reading a register of the interrupting peripheral and resetting the MIC PIA bit.

Interrupts may also be given by the microcomputer to the microprocessor of e.g. a connected synchronous communication controller (SCC). This is done when a bit of the MIC PIA peripheral register is set.

The specific use of the various interrupts in the different microcomputers of the system is described in the different chapters on the DU, FD, CPR etc. Detailed descriptions of the locations of the different interrupt vectors and addresses for mask setting is found in the memory maps.

Interrupt Priority and Address modifier (FPLA)

The interrupt priority control and address modification is made by an FPLA (field programmable logic array). See Fig. 6. The logic array is programmed to generate certain outputs in response to different combinations on the inputs. The programming is shown below:

Activating input combinations (pin 19 = 0V)				Activated output		
Internal address 4 3 2 1	AI = FFEX FFFX	Mask reg 3 2 1	Interrupts from latch 7 6 5 4 3 2 1 0	Signals	Active level	Comments
Active level						
1 1 1 1	0	1 1 1	0 0 0 0 0 0 0 0			
X X X X	X	0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	1 1 1 1 1 1 1 0* 1 1 1 1 1 1 0 X 1 1 1 1 1 0 X X 1 1 1 1 0 X X X 1 1 1 0 X X X X 1 1 0 X X X X X 1 0 X X X X X X 0 X X X X X X X	IRQ	Low	Only valid interrupts are signalled on IRQ line
0 1 X X 1 0 X X	0 0	X X X	X X X X X X X X	Set mask	High	FFE8-F FFF0-7
1 1 0 X	0	X X X	X X X X X X X X	IRQ, SWI vector fetch**	High	FFF8-B
0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 1	X	X X X	X X X X X X X X	P4–1 = 0000 0001 0010 0011		No modification
0 1 0 0 1 1 0 0	X 0	X X X 0 0 0	X X X X X X X X 1 1 1 1 1 1 1 0	0100 0100		No modification I0 vector addr–FFE8, 9
0 1 0 1 1 1 0 0	X 0	X X X 0 0 1*	X X X X X X X X 1 1 1 1 1 1 0 X	0101 0101		No modification I1 vector addr–FFEA, B
0 1 1 0 1 1 0 0	X 0	X X X 0 1 0*	X X X X X X X X 1 1 1 1 1 0 X X	0110 0110		No modification I2 vector addr–FFEC, D
0 1 1 1 1 1 0 0	X 0	X X X 0 1 1*	X X X X X X X X 1 1 1 1 0 X X X	0111 0111		No modification I3 vector addr–FFEE, F
1 0 0 0 1 1 0 0	X 0	X X X 1 0 0*	X X X X X X X X 1 1 1 0 X X X X	1000 1000		No modification I4 vector addr–FFF0, 1
1 0 0 1 1 1 0 0	X 0	X X X 1 0 1*	X X X X X X X X 1 1 0 X X X X X	1001 1001		No modification I5 vector addr–FFF2, 3
1 0 1 0 1 1 0 0	X 0	X X X 1 1 0*	X X X X X X X X 1 0 X X X X X X	1010 1010		No modification I6 vector addr–FFF4, 5
1 0 1 1 1 1 0 0	X 0	X X X 1 1 1*	X X X X X X X X 0 X X X X X X X	1011 1011		No modification I7 vector addr–FFF6, 7
1 1 0 0 1 1 0 0	1 0	X X X No one comb marked I0-I7	X X X X X X X X	1100 1100		No modification Default vector address
1 1 0 1 1 1 1 0 1 1 1 1	X	X X X	X X X X X X X X	1101 1110 1111		No modification

1 = +TTL 0 = 0V X = Don't care * or lower mask value * or higher interrupt
** Inhibits setting of the interrupt register

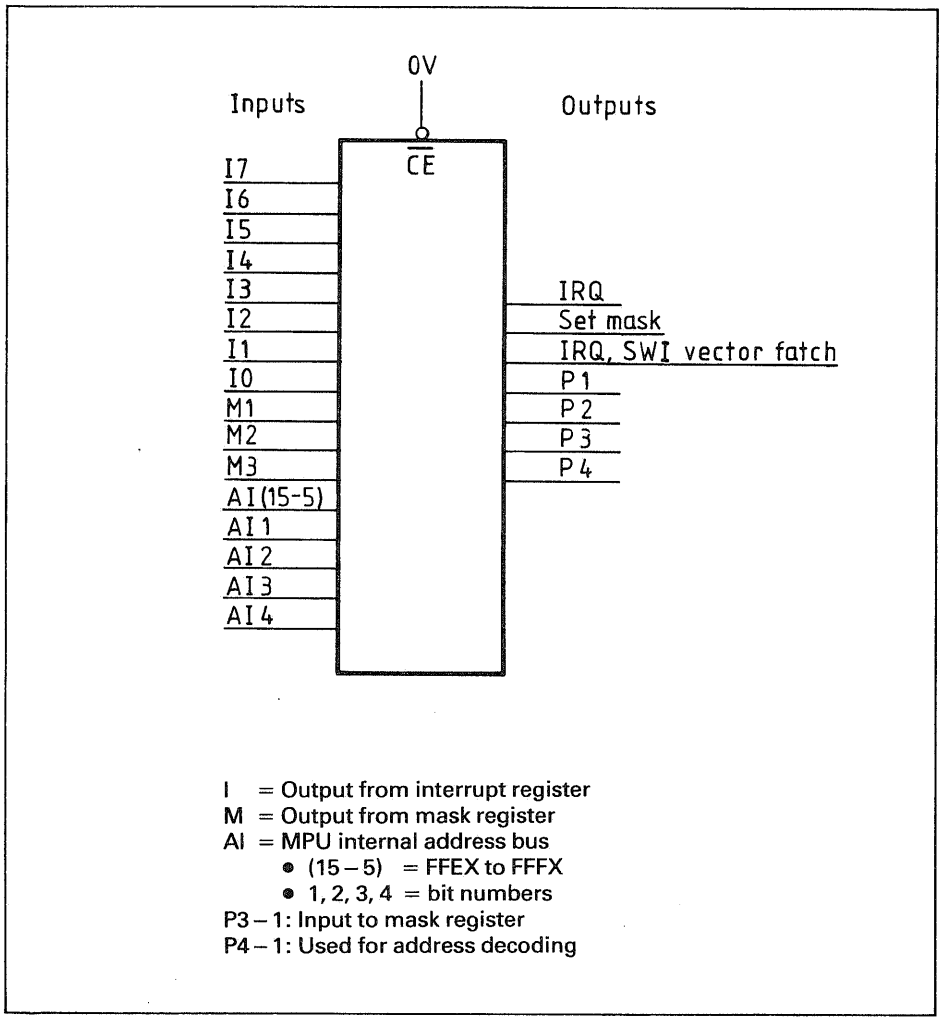


Fig. 6. FPLA inputs and outputs

Peripheral Interface Adapter, PIA

A peripheral interface adapter, PIA, is used in several places in the equipment, but the hardware description is placed here in connection with the microcomputer PIA, MIC PIA, used for interrupt control. The programming of the MIC PIA is discussed together with the microprocessors of each specific unit as it varies between them.

Signals

The PIA is typically used as an adapter between the 8-bit MPU data bus (D 7 through 0, see Fig. 7) and two peripherals with parallel 8-bit interfaces. The peripheral pins (PA7 – 0 and PB7 – 0) can be individually programmed to work as outputs or inputs. Each side (A and B) has two peripheral control lines (CA1 and CA2/CB1 and CB2 respectively) with programmable functions and an associated interrupt request lines (IRQA/IRQB). The IRQ outputs can e.g. be or-wired to the MPU IRQ input or each one tied to a priority interrupt controller input if different interrupt levels are used and/or fast identification of the interrupt source is wanted.

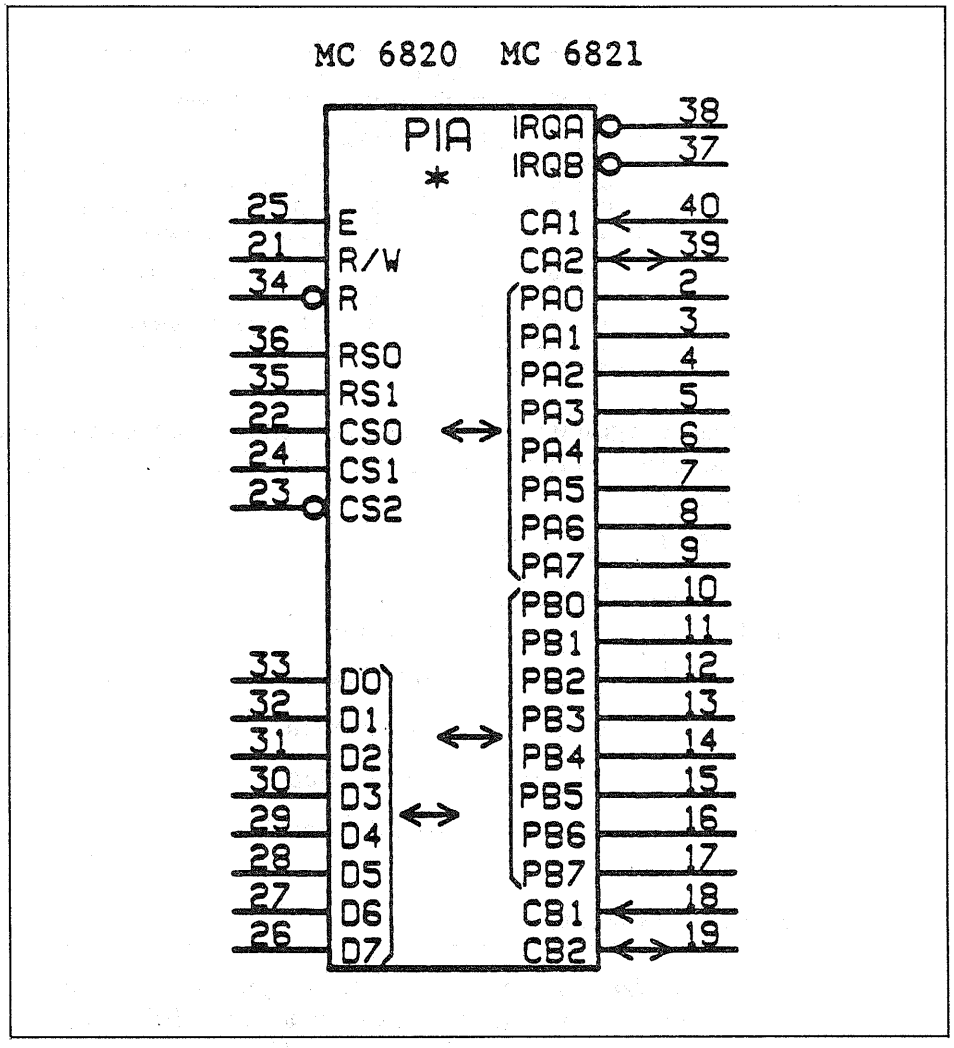


Fig. 7. PIA drawing symbol

Registers

Each side of the PIA has three 8-bit registers associated with it. These are:

- Data direction register (DDR). If a bit in the DDR is one the corresponding peripheral line will function as an output.
- Control register (CR), containing interrupt flags, interrupt mask and control, address control and control of the function of the peripheral control line CA2/CB2 (input or output).
- Peripheral register (P), containing data to/from the peripheral.

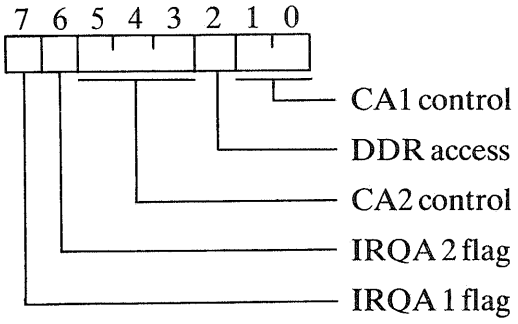
At reset all register contents are taken to zero.

The addressing of the six registers is made with the two register select lines RS1 and RS0 and one bit (bit 2) of the control register (to choose between DDR and peripheral register). At a specific time thus only four registers are available for the MPU as only two address lines are used:

RS1	RS0	Control Register Bit		Location selected
		CRA 2	CRB 2	
0	0	1	X	Peripheral Register A
0	0	0	X	Data Direction Register A
0	1	X	X	Control Register A
1	0	X	1	Peripheral Register B
1	0	X	0	Data Direction Register B
1	1	X	X	Control Register B

X = Don't care

The control registers for the A and the B side are similar with one exception and the format is therefore only described for CRA with the exception pointed out:



Bit(s) Comments

7 *IRQA 1*, Interrupt flag, goes high on active transition of CA1 and is automatically cleared when the MPU reads the peripheral register A. It may also be cleared by hardware reset.

6 *IRQA 2*, Interrupt flag. When CA2 is established as input (b5 = 0): Goes high on active transition of CA2 and is automatically cleared when the MPU reads the peripheral register A. It may also be cleared by hardware reset.

When CA2 is established as output (b5 = 1): IRQA 2 = 0, not affected by CA2 transitions.

5–3 CA2 (CB2) established as output by b5 = 1

b5 b4 b3
1 0 → CA2

b3 = 0: Read strobe with CA1 restore. CA2 goes low on the first high-to-low E transition following an MPU read of peripheral register A and is returned high by the next active CA1 transition.

b3 = 1: Read strobe with E restore. CA2 goes low on the first high-to-low E transition following an MPU read of peripheral register A and is returned high by the next high-to-low E transition.

CB2. The output functions of CB2 will not be similar to the same of CA2 when the control register B b5 – 4 = $10_{(2)}$. They are therefore inserted below:

b3 = 0: Write strobe with CB1 restore. CB2 goes low on the first low-to-high E transition following an MPU write into peripheral register B and is returned high by the next active CB1 transition.

b3 = 1: Write strobe with E restore. CB2 goes low on the first low-to-high E transition following an MPU write into peripheral register B and is returned high by the next low-to-high E transition.

b5	b4	b3	
1	1	0	Reset CA2. CA2 goes low when the MPU writes b3 = 0 into the control register.
		1	Set CA2. CA2 goes high when the MPU writes b3 = 1 into the control register.

CA2 established as input by b5 = 0

b5	b4	b3	
0	X	0	CA2 interrupt request disable. Disables IRQA MPU interrupt by CA2 active transition. IRQA will occur on the next (MPU generated) positive transition of b3 if CA2 active transition occurred while interrupt was disabled.
		1	CA2 interrupt enable. Enables IRQA MPU interrupt by CA2 active transition.

b5	b4	b3	
0	0	X	Active CA2 transition. IRQA 2 set by high-to-low transition on CA2.
	1		Active CA2 transition. IRQA 2 set by low-to-high transition on CA2.

2 Determines whether the data direction register or the peripheral register is addressed

0 = Data direction register is selected.

1 = Peripheral register is selected.

1 Determine active transition on CA1 input for setting interrupt flag IRQA 1 (CRA bit 7)

0 = IRQA is set by high-to-low transition on CA1.

1 = IRQA is set by low-to-high transition on CA1.

0 CA1 interrupt request enable/disable

0 = Disable IRQA MPU interrupt by CA1 active transition. IRQA will occur on the next (MPU generated) positive transition of b0 if CA1 active transition occurred while interrupt was disabled.

1 = Enable IRQA MPU interrupt by CA1 active transition.

Timing

Basic Timing

A basic 19.17 MHz clock signal is generated by a crystal clock. From this clock signal all timing signals, e.g. dynamic memory timing, two-wire transfer bit clock etc. are derived. Five 2.13 MHz clocks are generated (T1 – T5). Combinations of these are used to define different points of time during each half of the system clock period. The system clock, Ø, (microprocessor instruction clock) has a frequency of 1.065 MHz. It appears in several phases and with different pulse-pause ratios to compensate for propagation delays etc. The relations between the basic timing signals are shown in Fig. 8. Of the signals in Fig. 8; 9.585 MHz, Ø signals, RAS, COL, CAS and WES appear on buses connecting different logic boards.

The MPU demands two non-overlapping complementary clocks. Circuitry is added to provide these; MPU Ø1 and MPU Ø2. These two clock signals together with the data bus enable signal (DBE) for the MPU (Observe that these three signals are only fed to the MPU.) differ from the other Ø signals in that they may be frozen (in the Ø1 state) by a stretch signal. This signal is generated under certain conditions during a direct memory access (see Direct Memory Access).

Bus Timing Example

Consider the case when the MPU reads data from the basic read/write memory area.

- During Ø1 MPU puts out the new address and read level on the R/W line.
- DMA Ø2 will enable a ROM used for memory address decoding. An enable signal for one of the memory blocks will then be produced. During this time the memory address from the MPU will be applied and used as row address and column address to the RWM. The generation of row address strobe, selection of column address and generation of column address strobe is governed by the RAS, COL and CAS signals respectively.
- The internal and external data buses will be connected during Ø2. At the falling edge of Ø2 the data will be read into the MPU.

Programmable Timer Module, PTM

A programmable timer module, PTM, is sometimes tied to the microcomputer in order to avoid timing loops in the program.

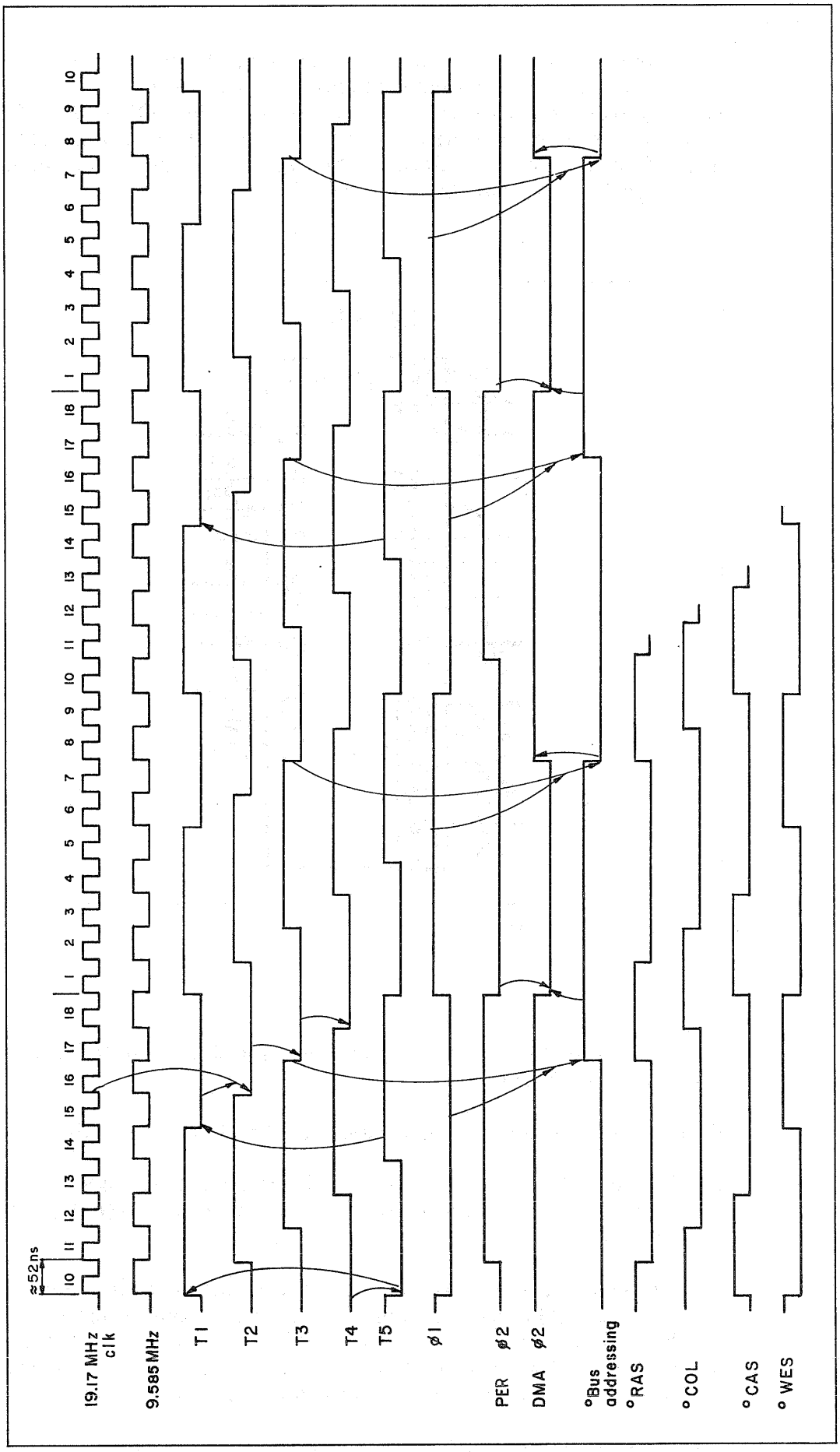


Fig. 8. Basic timing signals

Signals

The programmable timer module contains three 16-bit counters. Each one of these may be programmed to produce symmetric or asymmetric square waves (or single shots) at the respective outputs O1, O2, and O3 and also interrupts on an IRQ output. See Fig. 9. The clock signal for a specific counter may be selected to be either the ∅2 signal or a unique clock signal (C1, C2, and C3 respectively).

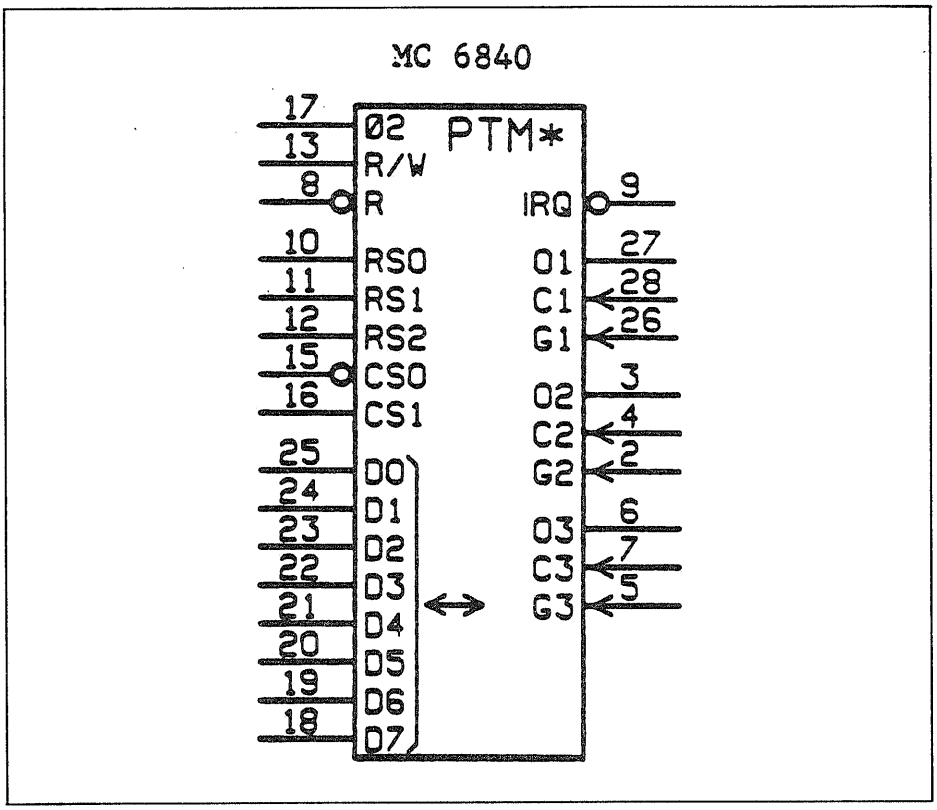


Fig. 9. PTM drawing symbol

RS2, RS1, and RS0 are used together with the R/W input for register addressing. See below.

D7 through 0 are used for connection to an 8-bit bidirectional data bus.

G1, G2, and G3 are used as enable signals for the three counters. When a G input goes low, the associated counter will be restarted (if the PTM is not reset).

A low level on the reset (R) input will stop all counters and reset the O1, O2, O3, and IRQ outputs as well as all status and control register bits but the internal reset bit. Furthermore, both the latches (see below) and the counters of all three timers will be preset to maximum values, and thus, if not reprogrammed before release, make the longest possible count cycle.

CS1 and 0 are the chip enable signals, needed to transfer data to or from the PTM.

Registers

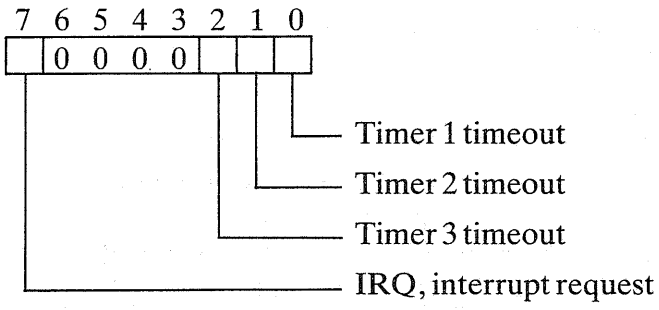
The PTM contains one control register for each timer and one status register. Each timer contains a 16-bit timer latch (write only) and an associated 16-bit counter. The register addresses are listed below:

Address inputs			Accessed register	
RS2	RS1	RS0	Write (R/W = 0)	Read (R/W = 1)
0	0	0	Control register 3 (if CR2:0 = 0) Control register 1 (if CR2:0 = 1)	No register No register
0	0	1	Control register 2 (CR2)	Status register (SR)
0	1	0	MSB buffer (timer 1)	Timer 1 counter MSB
0	1	1	Timer 1 LSB latches*	LSB buffer (timer 1)**
1	0	0	MSB buffer (timer 2)	Timer 2 counter MSB
1	0	1	Timer 2 LSB latches*	LSB buffer (timer 2)**
1	1	0	MSB buffer (timer 3)	Timer 3 counter MSB
1	1	1	Timer 3 LSB latches*	LSB buffer (timer 3)**

* When writing the LSB latches of a specific timer, the contents of the MSB buffer will be transferred to the MSB latches of that timer.

** After reading the MSB of a specific timer counter, the LSB buffer will, when read, present the contents of the LSB of that specific counter.

The interpretation of the status register (SR) bits is shown below:



Timeout of a counter always sets the associated status bit (SR:2, 1 or 0). It sets SR:7 only if bit 6 of the associated control register is 1.

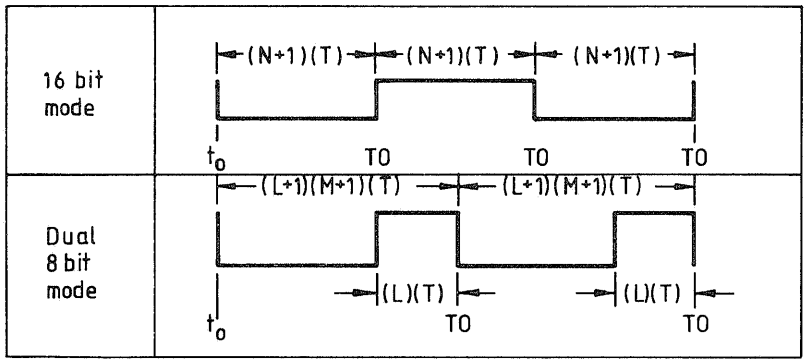
All status bits are cleared when the PTM is reset. A status bit that is present when the status register is read will be cleared by a following read timer counter command.

A specific timeout flag (status bit) is also cleared by an initialization or a write operation into the timer latches of the associated timer.

The interpretation of the control register bits is summarized below (CRX = control register associated with timer No. X)

Bit No.	Bit name	Function if = 1	Function if = 0
CRX:7	Output enable	Output (O1, O2, or O3) enabled	Output masked
CRX:6	Interrupt enable	SR:7 and IRQ enabled	No indications on SR:7 or IRQ output
CRX:5 (if CRX: 3 = 0)	Mode control	Single shot operation (Reinitialization needed for repetitive operation)	Continuous (cyclic) count operation
CRX:4 (if CRX: 3 = 0)	Initialization select	GX input going low restarts counter (See also CR1:0 = 1)	A write operation into the timer X latches (or GX ↓ or CR1:0 = 1) will initialize timer
CRX:3	Mode control	Frequency/pulse width comparison mode	Continuous/single shot mode
CRX:2	Counting mode	Dual 8-bit counting mode*	16-bit counting mode*
CRX:1	Clock select	Timer X counter uses clock on Ø2 input	Timer X counter uses clock on CX input
CR3:0	Timer 3 clock control	Timer 3 clock is divided by 8 before applied to the counter	Timer 3 clock functions as the other clocks
CR2:0	Control reg. address	CR1 may be written	CR3 may be written
CR1:0	Internal reset	All timers preset (Contents of latches transferred to counters)	All timers allowed to operate

* The output waveforms and timeout positions (T0) for the two modes are shown below:



N = 16-bit number in counter latch.
L = 8-bit number in LSB counter latch.
M = 8-bit number in MSB counter latch.
T = Clock input negative transitions to counter.
t_0 = Counter initialization cycle.
TO = Counter time out.

Memory Organization and Access

Memory Map

The standard memory organization of the System 41 microcomputers is described here. Special functions and differences will be described in the respective documentation dealing with the display units, communication processors, flexible disk units etc.

As shown in Fig. 10, the total memory map is divided into the RWM (read/write memory) area, the I/O area, and the IPL (initial program loading) ROM area.

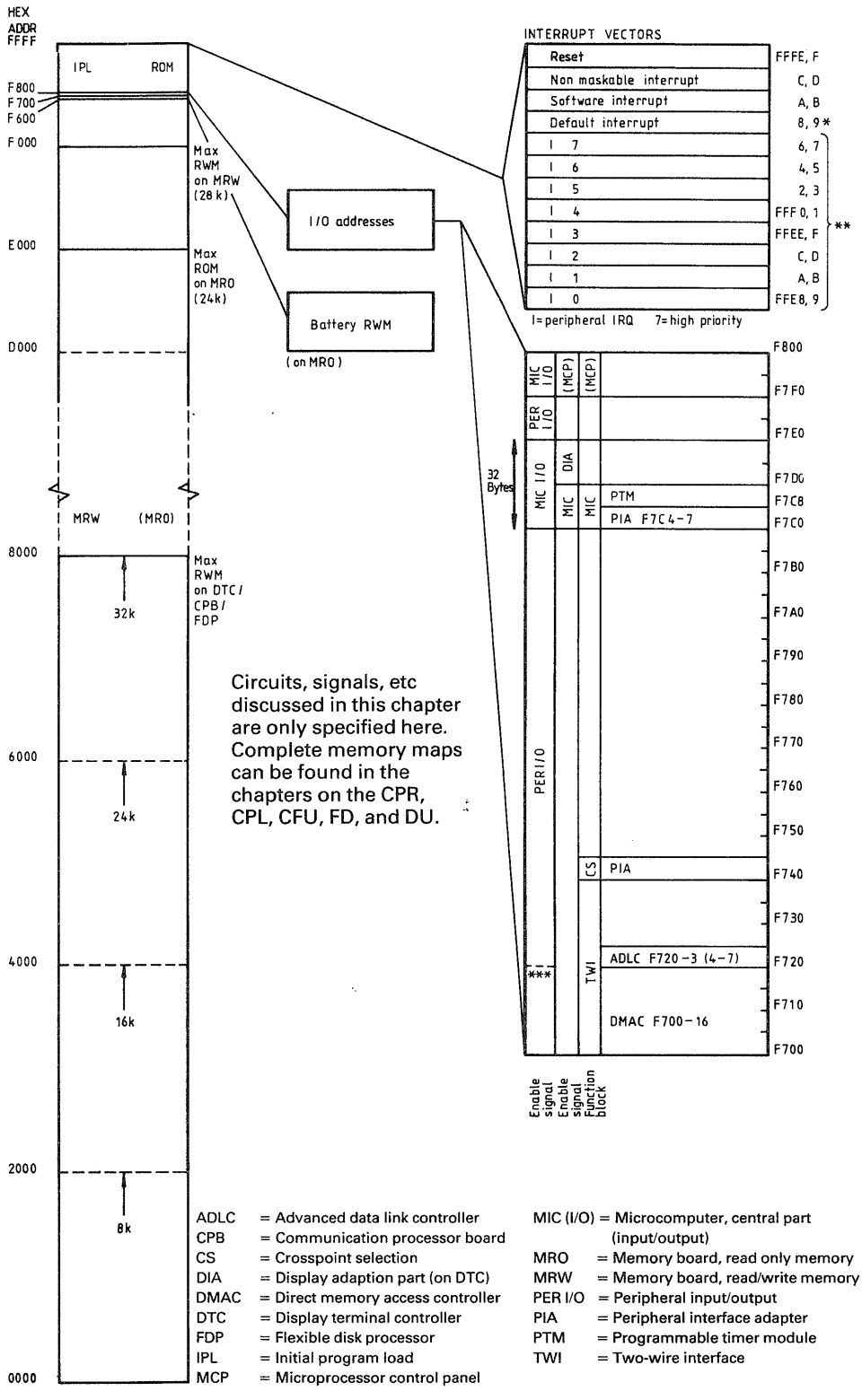
The RWM area is expandable from 32 kbytes to 60 kbytes. For more than 32 kbytes, however, memory expansion boards (MRW or MRO) has to be used. The accessible memory area can be decided by strapping. Strapping information is found in the Installation and Maintenance Manual. Parity generation and checking (odd parity) is carried out for the RWM area. Thus a 16 kbytes memory block consists of 9 × 16 kbits memory chips.

Note that the memory map also has got 256 bytes reserved for RWM with battery backup. (Please refer to the Memory Board RO chapter.)

The I/O area consists of 256 reserved addresses, i.e. addresses to registers of LSI circuits on the CPB/DTC and connected boards. For communication processors and display units some of these are associated with circuits on the internal buses (MIC I/O) and some with circuits on the external buses (PER I/O). See Figs 1 and 10.

Note that the basic system units (display units, flexible disk units, communication processors) are accessed via two-wire connections. The send and receive data registers of the two-wire interface are located at some of the I/O addresses.

The IPL ROM area consists of 2 kbytes (in display units and communication processors) or 4 kbytes (in flexible disk units). It contains programs for checking of the RWM and ROM areas, and communication and program loading routines needed at power on. It also contains twelve interrupt vectors, i.e. addresses to the twelve basic interrupt routines.



* MPU reading FFF8 – FFF9 will transfer the contents of two cells within FFE8 – FFF9, depending on highest priority valid interrupt, to the MPU.

** A (read or preferably) write operation into one of the cells FFE8 – FFF7 will result in a setting of a corresponding mask value. As an example, FFEE or FFEF **from the MPU** will disable interrupts of priority < 3.

*** Not present during the three DMA cycles DMA T1 – T3.

Fig. 10. General microcomputer memory organization

Address Decoding and Direct Memory Access

An address decoder (see Fig. 1) provides chip enable signals for the various system circuits as the IPL ROM, PIA etc. It also enables the bus buffers at the right time. For example, the internal buses will be isolated from the rest of the system during transfers of data between the memory and the two-wire interface (DMA, direct memory access, transfers). This means that during a cycle when the MPU is doing internal work (e.g. a register to register transfer), i.e. when no relevant or valid memory address is present, or when the MPU is just needing the internal buses to reach a circuit, the DMA logic does not need to steal any clock cycle from the MPU, which would slow down the program execution. Thus the DMA logic may access the memory via the external buses in the same time as the MPU communicates with, say, the peripheral interface adapter, MIC PIA, via the internal buses.

The DMA method is described in the paragraph Direct Memory Access.

Address Decoder (FPLA)

The address decoder consists of a field programmable logic array, FPLA. See table below. According to the state of the address bus *from the MPU* and other signals shown in the block diagram (Fig. 1) the address decoder provides circuit enable signals for different groups of circuits, on the internal buses and the external buses, and a stretch signal for the timing logic. See DMA Timing paragraph.

Activating input combinations				**Activated output**		**Comments**
A15–4 Hex	°VMA	°DMA T2	DMA T1–3	Signal	Active level	
000–F6F	Low	Low	X	Stretch	Low	←DMAC addressed
F70–F71	Low	X	High		Low	
F72–F7B	Low	Low	X		Low	
F7E	Low	Low	X		Low	
F60–F6F	Low	High	X	BAT MEM	High	Battery RWM enable
F70–F71	Low	X	Low	PER I/O	Low	DMAC only enabled if during no DMA cycle
F72–F7B	Low	High	X		Low	
F7E	Low	High	X		Low	
F7C–F7D	Low	X	X	MIC I/O	High	External data bus read disable ←(IPL ROM addressed)
F7F	Low	X	X		High	
F80–FFF	Low	X	X		High	
F80–FFF	Low	X	X	IPL	High	IPL ROM enable
F7D	Low	X	X	DIA	High	
F7C	Low	X	X	MIC	High	
F7F	Low	X	X	Panel	Low	MCP addressed

See also memory map, Fig. 10.

Memory Access Multiplexing

The read/write memory of the microcomputer is connected to the external buses. It may be accessed from several sources.

The memory is accessible for the MPU. The DMA logic is also capable of stealing clock cycles from the MPU and read or write data in the memory, without intermediate storing in an MPU register.

The address selector of Fig. 1 is among other things governed by the system clock signals. Access to the memory is multiplexed in the way that during one half of the system clock period the MPU or the direct memory access controller may have access the memory. During the other half a refresh or a controller address is coupled to the memory. When dynamic R/W memory is used, periodic refresh addressing is necessary. Controller addresses may be provided by, for a part of the RWM on the DTC, the cathode ray tube controller or, for a certain area of the total CP memory, by a peripheral (e.g. synchronous communication) controller. A controller will thus be able to read and write in the memory without disabling the MPU to work on the buses.

Direct Memory Access

Direct memory access (DMA) is a possibility in the Alfaskop System 41 that minimizes the load on the MPUs caused by communication between display units, flexible disk units and communication processors and also by flexible disk read/write operations carried out by the microcomputer of the flexible disk unit. See Fig. 11. The DMA enables data transfers directly between a communication interface circuit and the memory, without the MPU having to:

- Serve an interrupt request for each data byte
- Intermediately taking care of the data. The large scale integrated circuit DMAC (direct memory access controller) that is used may control up to four data channels. The manner in which the DMA affects the bus enabling is described in the paragraph Address Decoding and Direct Memory Access above. The communication interface is described in the chapter Communication.

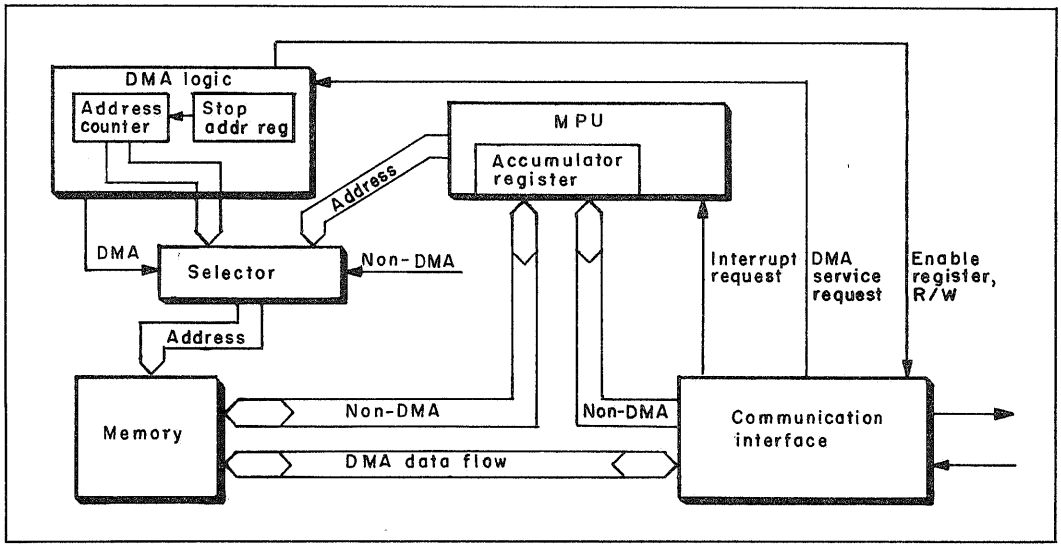


Fig. 11. DMA principle

Direct Memory Access Controller, DMAC

The direct memory access controller, DMAC, contains one 16-bit address register, one 16-bit byte count register and one 8-bit control register for each of the four channels that may be serviced. The address register holds the address of the location to or from which the next byte is to be transferred. The byte count register holds the number of bytes still to be transferred. Furthermore the DMAC contains three general control/status registers. These registers may be accessed by the MPU by use of the five address lines, A4 – A0, the CS line and the R/W line. For addressing and programming of the DMAC registers, see Fig. 12 and Table 1.

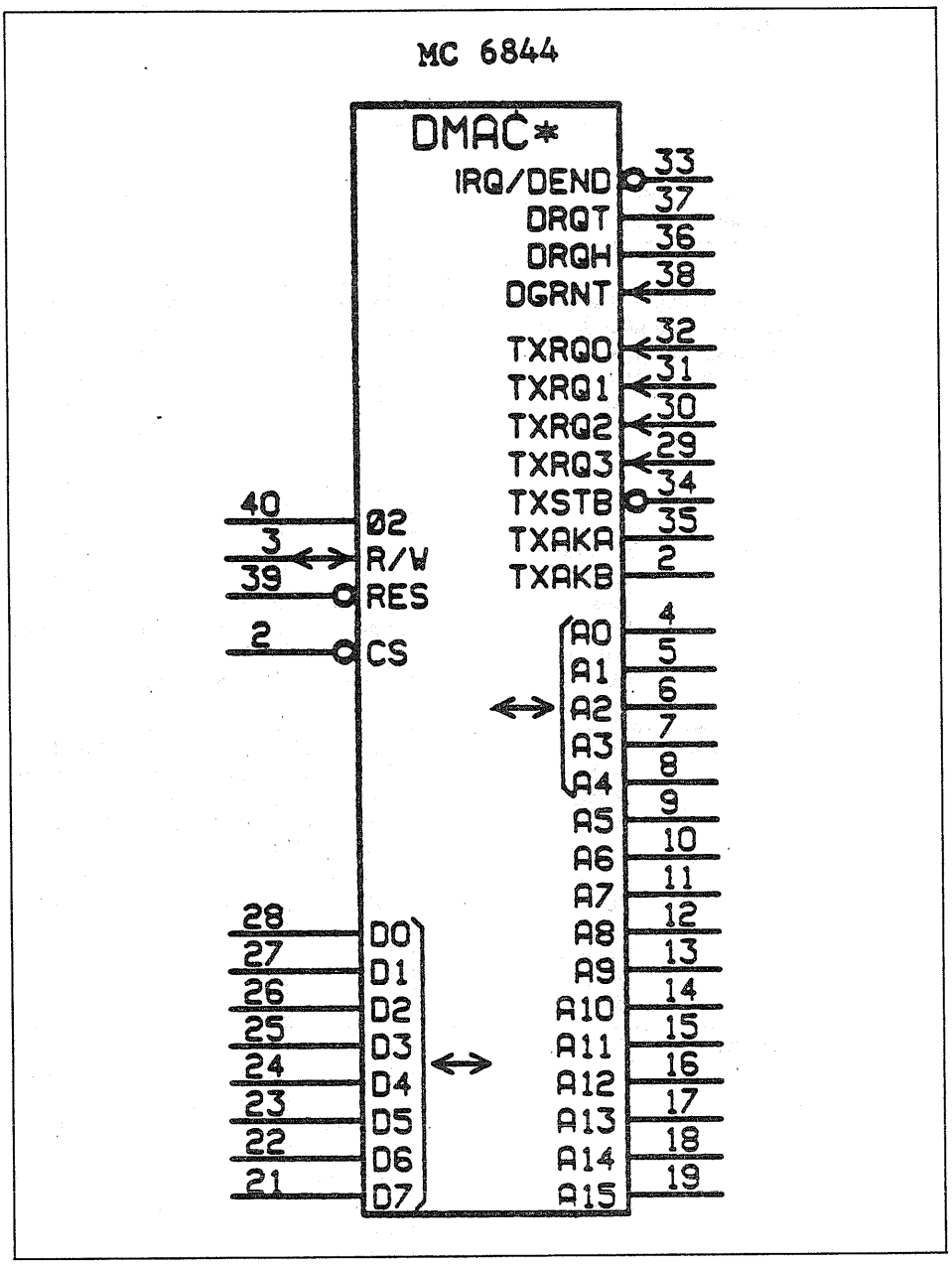


Fig. 12. DMAC drawing symbol

During DMA the DMAC provides a full 16-bit memory address, a read/write signal and a signal corresponding to the MPU VMA signal, namely Transfer strobe, TXSTB. See Fig. 12. A completed group of memory

Table 1. Registers of DMAC

Register								Register address A4 3 2 1 0
Address channel 0	Hi							0 0 0 0 0
	Lo							0 0 0 0 1
Byte count channel 0	Hi							0 0 0 1 0
	Lo							0 0 0 1 1
Address channel 1	Hi							0 0 1 0 0
	Lo							0 0 1 0 1
Byte count channel 1	Hi							0 0 1 1 0
	Lo							0 0 1 1 1
Address channel 2	Hi							0 1 0 0 0
	Lo							0 1 0 0 1
Byte count channel 2	Hi							0 1 0 1 0
	Lo							0 1 0 1 1
Address channel 3	Hi							0 1 1 0 0
	Lo							0 1 1 0 1
Byte count channel 3	Hi							0 1 1 1 0
	Lo							0 1 1 1 1
		b 7	b 6	*	b 3	b 2	b 1	b 0	
Channel 0 control Channel 1 control Channel 2 control Channel 3 control		Status bit DMA end. Reset by MPU read of CCR.	Status bit Busy. Set during transfer. Reset after DMA end.		Address up/down. 1 = Decrement, 0 = Increment, address counter	TSC/Halt. 1 = TSC mode. 0 = Halt modes.	Burst/ Steal. 1 = Burst mode. 0 = Steal modes.	R/W. 1 = Read from memory (to ADLC). 0 = Write in memory (Read ADLC).	1 0 0 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 1 1
Priority control		Rotate pri. 1 = Served channel has lowest pri. 0 = Fixed pri ch 0 1 2 3.	N. u.		Enable TXRQ3 from channel 3. 0 = Disable.	Enable TXRQ2 from channel 2. 0 = Disable.	Enable TXRQ1 from channel 1. 0 = Disable.	Enable TXRQ0 from channel 0. 0 = Disable.	1 0 1 0 0
Interrupt control		Status bit IRQ/DEND = CCR 0, 1, 2, 3 bit 7 if interrupt enabled for channel. Reset by MPU read CCR	N. u.		IRQ/DEND enable 3. IRQ at ch 3 DMA end. 0 = Disable IRQ	IRQ/DEND enable 2. IRQ at ch 2 DMA end. 0 = Disable IRQ	IRQ/DEND enable 1. IRQ at ch 1 DMA end. 0 = Disable IRQ	IRQ/DEND enable 0. IRQ at ch 0 DMA end. 0 = Disable IRQ	1 0 1 0 1
Data chain		N. u.	N. u.		4/2 ch select. 1 = 4-ch-mode; Pin 2 used as CS/TXAKB. 0 = 2-ch-mode; Pin 2 only CS	DCB. 00 ch 0 01 ch 1 10 ch 2 11 (Illegal)	DCA. } uses values in ch 3 addr. and byte count regs to update	DCE. 1 = Data function enabled. 0 = No data ch.	1 0 1 1 0

Hi (high) = bits 15 – 8, Lo (low) = bits 7 – 0, N. u. = Not used, CCR = Channel control register

* Bit 5 and 4 are not used.

accesses may be signalled (e.g. to the MPU) on an IRQ/DEND line (DMA end). Transfer request, TXRQ3-0, are the inputs for DMA requests from the four channels that may be serviced by the DMAC. A TXRQ input is thus activated when e.g. an ADLC is ready to read or write a byte in the memory. See Communication chapter, ADLC description, TDSR and RDSR signals. DMA request three-state control steal, DRQT, is the signal with which the DMAC reacts to a transfer request in the TSC (Three-state control, see below) steal mode. Thereby the R/W and address lines from the MPU could be forced into the high impedance state, e.g. by disabling the bus buffers (see Fig. 1).

If needed, the clock circuitry of the system should stretch the Ø1 period of the MPU clock. As the stretch signal is tied to the DBE (Data bus enable) input, the data bus lines of the MPU will also be in the high impedance state. Note that the system clock is not stretched, but e.g. provided to the Ø2 input of the DMAC.

When the above actions are taken, on MPU Ø2 going low (to rest low for a while), a signal DGRNT (DMA grant) should be returned to the DMAC.

If more than one receive/transmit channel is used Transfer acknowledge A, TXAKA (2 channels), and perhaps Chip select/Transfer acknowledge B, CS/TXAKB (3 or 4 channels), may be used as encoded address lines to direct a transmit acknowledge signal and a DMA end signal to the appropriate peripheral controller.

2-channel mode: TXAKA 1 = Channel 1, 0 = Channel 0

4-channel mode:

CS/TXAKB	TXAKA	Channel
0	0	0
0	1	1
1	0	2
1	1	3

DMAC Programming Example

As an example, consider the case of a single display unit that wants to send a block of information to a flexible disk unit on the two-wire connection via an ADLC using DMA channel 0 with TSC steal mode. Suppose that the block of information has a length of Len bytes and is situated at memory addresses Adr, Adr+1, ..., Adr+(Len−1). The MPU then prepares the DMAC by writing Adr in the 16-bit address register and Len in the 16-bit byte count register of channel 0. The channel control register 0 must also be programmed for:

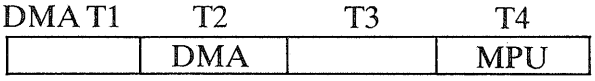
- Address counting upwards
- TSC mode
- Steal mode
- Read mode, as data is to be read from the memory.

The enable bit for TXRQ 0 should be set, to permit the Transmit data service request (TDSR) signal from the ADLC to start the sequence described above in the section Direct memory access controller DMAC.

When one memory byte has been stored in the ADLC, the MPU regains control of the whole bus system and will be busy with other tasks during about 27 μs, after which time one byte has been transferred to the flexible disk unit. Then the transmit data register of the ADLC is again available. This will be signalled on the TDSR output tied to the TXRQ 0 input of the DMAC etc., until DMA end, that may be signalled on the IRQ/DEND line.

DMA Timing

Though a direct memory access only takes one MPU clock cycle time, a full DMA period consists of four clock cycles, DMA T1 – T4.



During T2 the actual DMA takes place and the MPU clock is stretched unless the MPU is working on the internal buses only or does not provide VMA (Valid memory address). During DMA T2 a system VMA is provided.

During T1 and T3 the MPU clock is stretched only if the MPU addresses the DMAC say to read Busy or DMA end status for a certain DMA channel. This is to prevent the MPU signals from interfering with the DMAC signals that are also present during T1 and T3.

During T4 the MPU is never disturbed. This is to ensure that data in the MPU is not lost because of the absence of clock. (In CPs, when more than one channel is working, the MPU could otherwise be stopped for a longer time than three clock cycles.)

Communication

Contents

General

This chapter is compiled in order to give a survey of Alfaskop System 41 external and internal communications and to describe line procedures and communication circuitry common for two or more units.

Remote Host Computer Communication

A single display unit or a communication processor, is capable of remote communication with a host computer. The communication is usually carried out via modems.

The communication protocol is described in the Alfaskop System 41 Reference Manual for applicable emulation. The communication circuitry and hardware line interface can be studied in the Synchronous Communication Adapter or the Synchronous Communication Controller chapters in this file.

Local Host Computer Communication

A communication processor, local is capable to communicate with an IBM host computer via an I/O channel interface.

The communication protocol is described in the Alfaskop System 41 Reference Manual IBM 3270 Emulation and the Communication Processor, Local chapter in this file. The channel communication circuitry and hardware channel interface are also described in the Communication Processor, Local chapter.

Internal Communication via Two-wire

The main internal communication is carried out via two-wire connections between the following units:

- Communication processor – display unit
- Communication processor – flexible disk unit
- Flexible disk unit – display unit

The communication protocol and the communication circuitry are described below in a general but yet rather detailed way. Information, specific for a certain unit, is given in the chapter on that unit, i.e. in the chapters:

- Communication Processor, Remote
- CPR and FD Unit
- Flexible Disk Unit
- Display Unit

Software

The software that handles the internal communication is a part of the operating system, thus it is not dependent on emulation or application, only on the type of unit in which it is used, i.e. on hardware and configuration.

The main program module handling the internal communication is the Communication handler. It handles the hardware (communication interface circuits), both for reception and transmission. Poll and Answer to poll routines also form a part of the Communication handler. The Communication handler contains routines activated by hardware interrupt signals e.g. on Frame complete (whole frame received), Not-clear-to-send (whole frame transmitted) or framing error, as well as lower priority routines e.g. for changeover between transmission and reception.

The services of the Communication handler are through a Supervisor selector called for by different supervisors e.g. Input/Output manager, FD I/O supervisor, PU I/O supervisor, IPL supervisor, Console mode supervisor and Utility supervisor or by emulation software (User) through a User interface.

Fig. 1 is inserted in order to give an orientation on the software layout.

Sessions

In a configuration one unit always controls the physical communication. This unit is called configuration master. In a cluster configuration the CP is the configuration master whereas the DU is configuration master in a single configuration.

Besides, there is a logical concept of communication. This is called a session. A session may consist of one or several sequences of physical

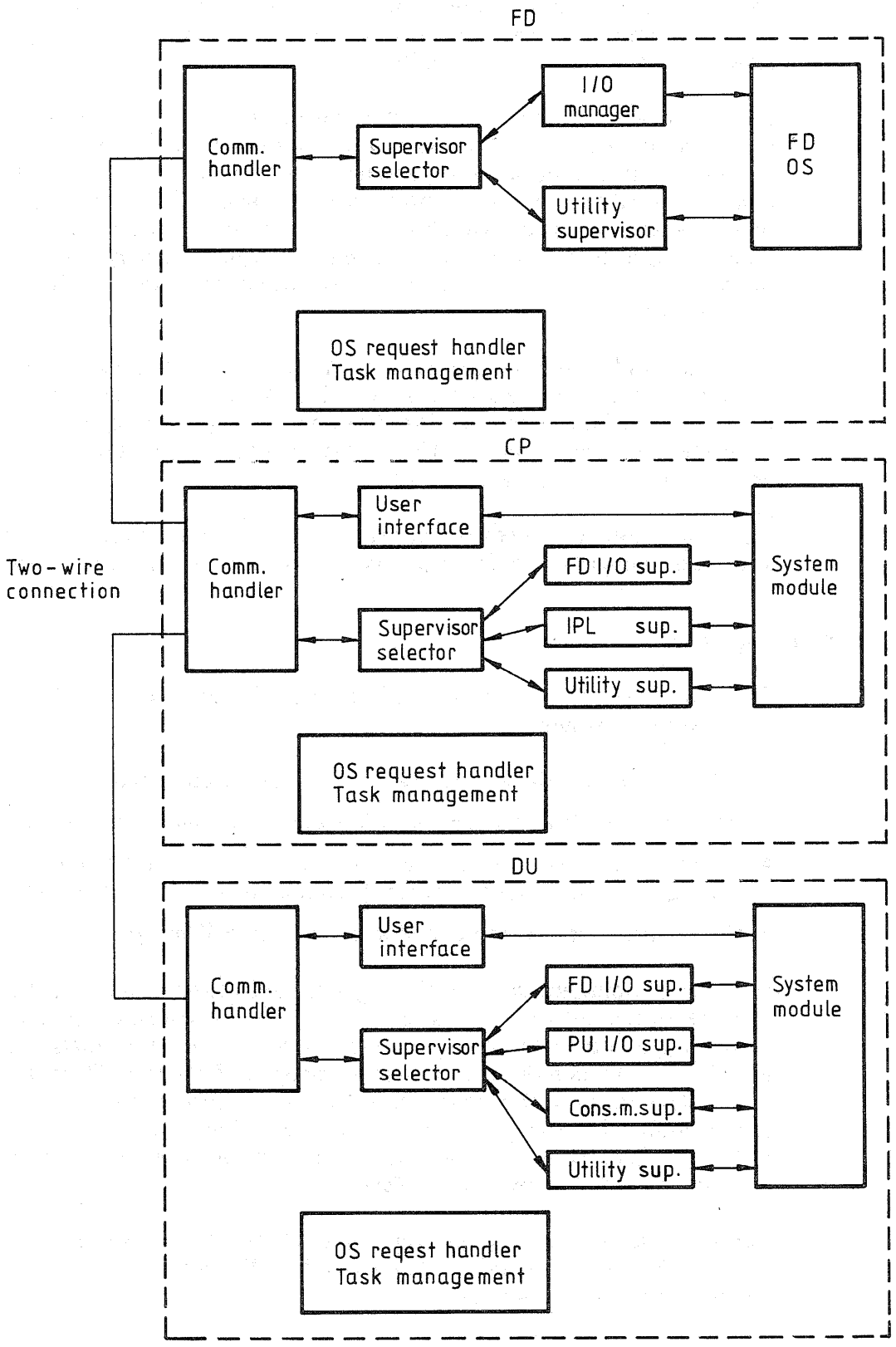


Fig. 1 Internal communication software

communication. The channel is disconnected after each sequence of physical communication and may thus be used by other units, while processing is done by the first units.

In this way a very effective use of the channel is obtained. When a unit has given a positive Answer to poll a session is opened and is then regarded as going on until a special command has concluded it. One of the units, being involved in a session, is always session master. The other unit is then session slave. The session mastership may be changed at different times during a session. Any unit in a terminal configuration may be session master. A session is executed under the control of the User interface or a supervisor.

A session is identified by a session control block, SCB, in each of the communicating units including the configuration master. The SCB is registered until the session is closed or aborted due to an error or disconnection of an affected unit. The SCB is identified by a session number and sometimes a supervisor number.

Fig. 2, showing a load session consisting of two data communication sequences, is inserted in order to give an idea about the function of a session.

Main Message Types

There are different types of messages (frames) that may be sent between the units in the system:

- Poll from the configuration master. Message length 4 bytes.
- Answer to poll from a polled unit. Maximum message length 64 bytes.
- Communication control messages like acknowledgements of received messages or queued messages etc. Maximum message length 64 bytes.
- Data. Maximum recommended message length 4096 bytes.

In a cluster configuration the CP regularly polls the terminals. It lists terminal status and investigates and queues requests from the terminals. According to a connect request queue, it interconnects terminals, e.g. a DU to a FD when the DU wants new program from the FD. It then leaves the control of the communication DU – FD by sending a Conn (connection acknowledge) frame to the FD and another Conn frame to the DU and awaits a frame from the DU when the communication is terminated. See below: Msgtyp/Status layout.

Actual connection status is registered in the CP so that no polling of the units on the involved two-wire line from the CP will take place during the time the connection lasts.

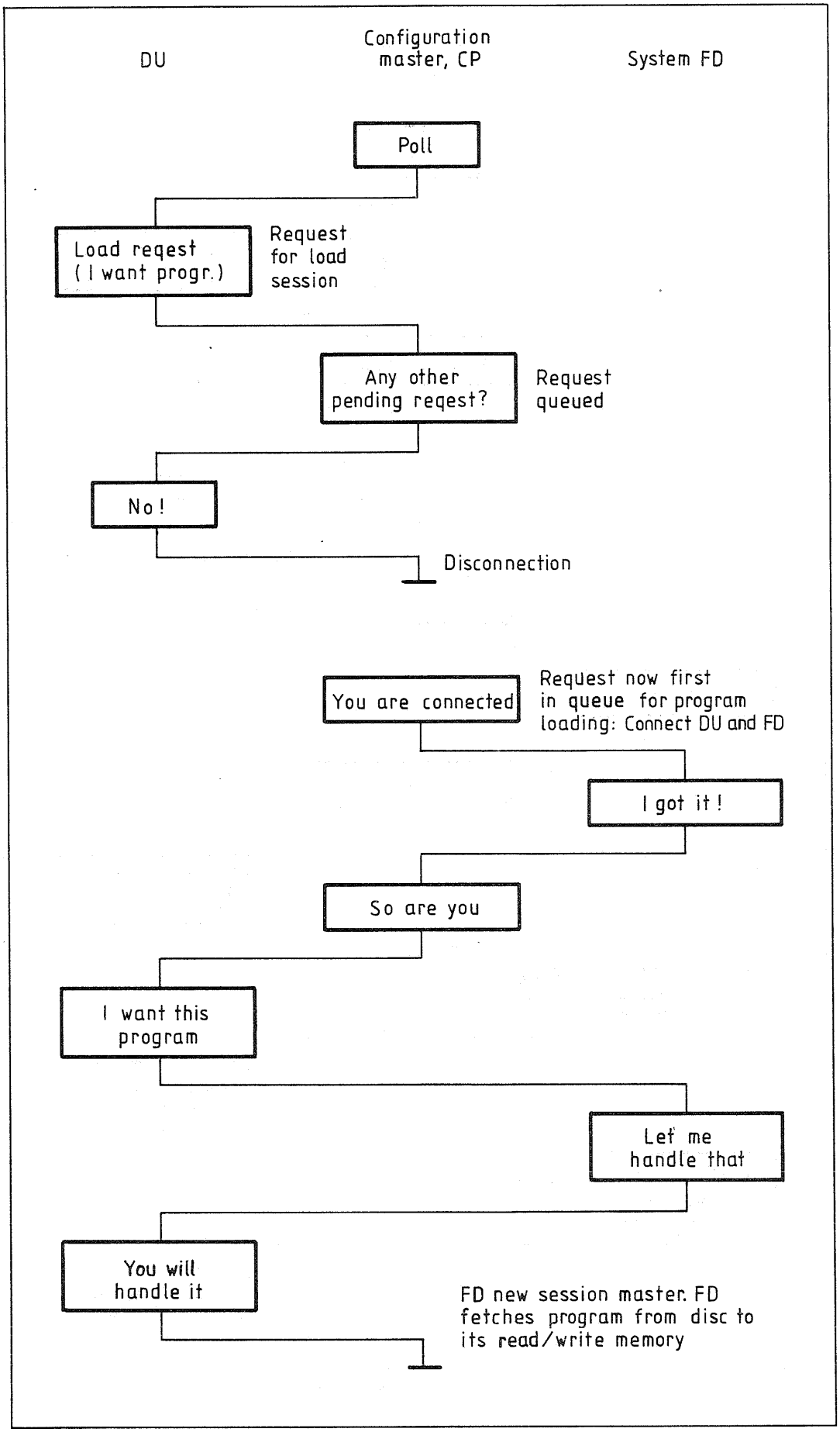


Fig. 2a Session example

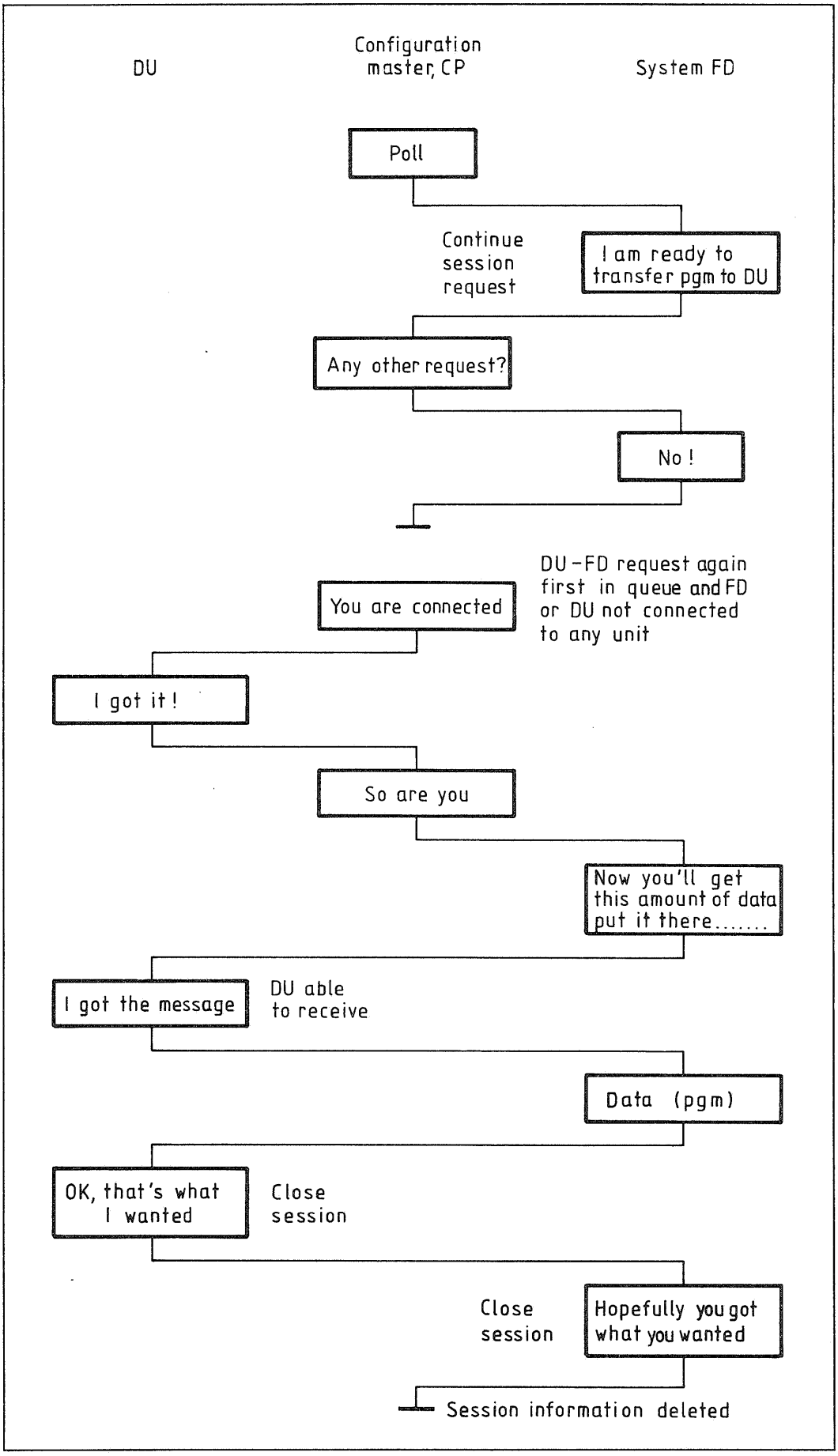


Fig. 2b Session example

Message Format

General Message Format

The message format used in communication between Alfaskop System 41 terminal units is:

- Word length 8 bits.
- Message length 4 – 4096 words (bytes), depending on type of message, flags and CRCC excluded.

The general message format is shown in Fig. 3.

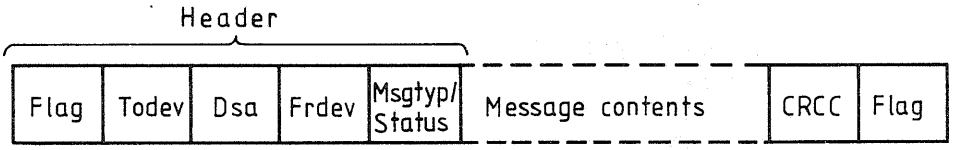


Fig. 3 General message format

The various fields in Fig. 3 have the following meaning:

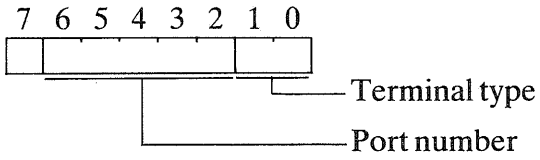
- Flag: Control byte ($7E_{(16)}$) for start of message and byte synchronization. Added/stripped by the hardware logic.
- Todev: Physical address of the unit to which the message is sent.
- Dsa: Destination session address. This byte is used in certain message types to define a destination session control block in the receiving unit.
- Frdev: Physical address of the unit sending the message.
- Msgtyp/Status: Byte, defining status of configuration master (Poll message) or slave status (Answer to poll message) or Type of message (for all other messages).
- Message contents: Field normally containing between 0 and 4092 bytes depending on the message type. It may contain more than 4092 bytes, but this is not recommended. It may contain additional control bytes or data.
- CRCC: Cyclic redundancy check character (16 bits), used to check whether data was transferred correctly or not. Calculation/appending as well as checking/stripping of CRCC is handled by the hardware logic.
- Flag ($7E_{(16)}$): Defines the end of the message. Added/stripped by the hardware logic.

Except in the flag character, which contains six consecutive ones, more than five consecutive ones are not sent in a frame (between flags). This is prevented as the transmitter logic inserts a zero after five ones and the receiver logic always strips a zero following five received ones. If $01111110_{(2)}$ anyhow is received that is interpreted as a flag.

If seven or more consecutive ones are received, inside a message, that is interpreted as an abort indication telling the receiver logic that the frame is invalid.

Todev and Frdev Layout

Todev and Frdev have the following format:



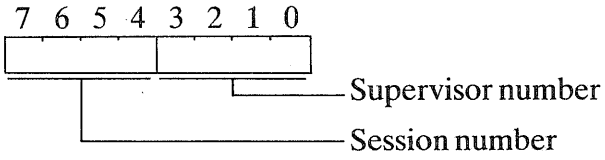
The fields means:

- Port number: Number (0 – 31) of two-wire connection (port) on the TUA boards of the CP. For single display unit configuration the port number is zero.
- Terminal type:
  00 = DU
  01 = PU-V.24/V.28
  10 = FD
  11 = Spare

Todev and Frdev contain actually the physical device addresses as shown in Appendix 1.

Dsa Layout

The layout of the Dsa field is:



The fields means:

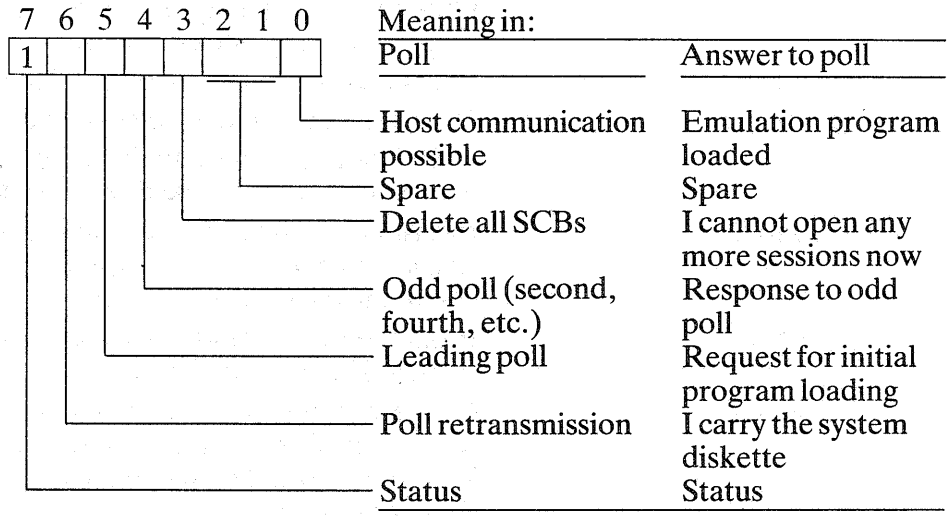
- Session number: Number of the registered session.
  - A DU may register up to five sessions plus ten more for an attached V.24/V.28 PU.
  - A PU registers up to 10 sessions.
  - An FD registers up to eight sessions.
  - If session number = $F_{(16)}$, session number is asked for (undefined).
- Supervisor number: The used numbers are listed below followed by the name of the corresponding software module and the units in which the software is used.

0	Input/output manager			FD
1	Utility supervisor	DU	CP	FD
2	FD I/O supervisor	DU	CP	
3	Printer I/O supervisor	DU		
4	Console mode supervisor	DU		
5	IPL supervisor		CP	
E	Host 1 supervisor (User interface)	DU	CP	
F	Host 2 supervisor (User interface)	DU	CP	

Dsa is $FF_{(16)}$ or defines the drop address on a two-wire in poll messages. See Appendix 2. Dsa is $FF_{(16)}$ in Answer to poll messages.

Msgtyp/Status Layout

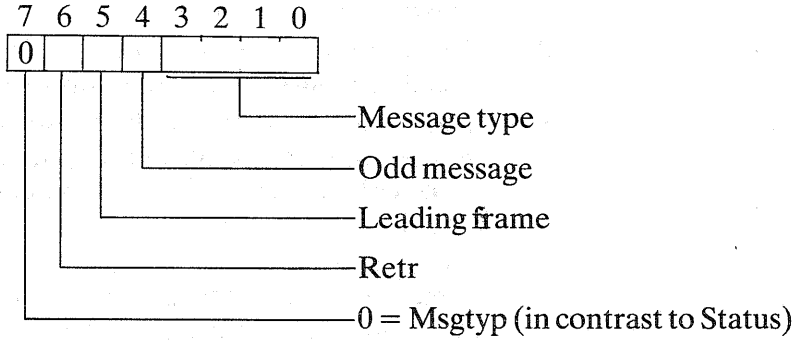
In the case of Poll or Answer to poll messages, byte 4 of the message header is interpreted as status. It then has the layout shown below. A set bit has the stated meaning:



A poll retransmission is performed when the configuration master has not got an answer on the previous poll before a timeout.

The first poll in a series of polls to a terminal unit is a leading poll and defined as an even poll (b4 = 0). A unit is polled as long as it responds with a request. When it has got nothing to transmit it sends a negative poll answer and the next poll to the unit will then be a leading poll. If a unit has a request or not is indicated in an Answer type byte following the Status byte (see Appendix 2).

For all messages, except Poll and Answer to poll, byte 4 of the message header is interpreted as a type of message definition, Msgtyp. The layout is shown below:



Retr means retransmission and is only used by session or configuration master, when it has not got an answer to a message before a timeout.

Leading frame is used by the configuration master only in the first frame of a communication sequence.

Odd message is used by a slave when received together with Retr to check whether it has already received the message or not. The first frame is defined as even.

The message type field (bit 3 – 0) has the meaning listed below:

Mnemonic	Bits 3 – 0	Meaning
Ack	0	Acknowledgement of Info, Info Dh, Data. Other messages may also work as acknowledgements, e.g. Break or Eos.
Info	1	Information frame, maximum 64 bytes.
Data	2	Data frame, recommended maximum 4 kbytes.
Dh	4	Data header, first 4 data bytes are coming in this frame. The rest of the data in the next frame.
Info Dh	5	Information data header, information and first 4 data bytes are coming in this frame. The rest of the data in the next frame.
Rtrd	7	Ready to receive data. Used as acknowledgement to Dh and Info Dh messages.
Sconn	8	Connection message to a session slave from the configuration master or connection acknowledgement from the session slave to the configuration master.
Denq	9	Data enquiry, sent by the session master when no answer is obtained within a certain time to a Data message.
Abort	A	Abort message, containing information on what session to abort, sent from the configuration master to the session slave. It can also be sent from the session master to the configuration master in response to a Mconn message.
Break	B	Break message, initiating change of session mastership, sent from the session slave to the session master. Also sent from the session master to the configuration master to order a disconnection.
Mconn	C	Connection message from the configuration master to the session master.
Reject	D	Reject message, sent from a unit not able to handle a request now. Queue up a new request.
Eot	E	End of transmission, sent by the session master to the configuration master to order disconnection without mastership change and without ending of the session.
Eos	F	End of session, sent by the session slave to the session master and from the session master to the configuration master to order disconnection *and* ending of the session.

A summary of the formats of specific messages is shown in Appendix 2.

Polling

The polling of terminals in the system is handled by the configuration master (CP, single DU or the like). All configured terminals are polled.

The fourth byte of a Poll message defines e.g. whether the poll is a retry of an earlier poll which resulted in no answer, if the poll is even or odd, and whether the host computer communication is maintained or not.

Retransmission of a poll occurs once after 10 ms when an answer is lacking. If no answer is obtained to this first retransmitted poll, the terminal is regarded as "off" and polled more seldom. (All answering terminals will be polled and then one out of the "off" terminals – all answering terminals polled again and then the next one of the "off" ones etc.)

The answer to a Poll message may either be negative (no request) or contain a request for opening or continuation of a session or for communication with the host computer (emulation request). Only one request is sent in each Answer to poll. The same terminal is polled until a negative answer is obtained. The fifth byte of an Answer to poll, i.e. the Answer type byte, indicate no request or defines the type of request. See Appendix 2 for details.

Data Communication

The requests for opening or continuation of sessions are registered in first-in-first-out queues in the configuration master. There is one queue for each one of eight priority levels (see Appendix 2).

According to these queues, the configuration master connects the terminals in the system. After that a physical connection is established a Conn (connect) message is sent to each one of the involved terminals (and in certain cases also to the affected supervisor program in the configuration master). Then data or more detailed information concerning the request may be transferred between the affected units.

The session example of Fig. 2 shows how a load session is divided into two data communication sequences to free the communication channel for other sessions and units when it is not needed by the load session. (The present session master asks for continuation of the session in an Answer to poll.) However, during IPL (initial program loading) the connection between the program store (e.g. FD) and the terminal requesting IPL will not be broken e.g. while the FD is reading from a track into its memory (applicable if more than 3.25 kbytes, one track, of program is to be transferred to the terminal). During a transfer of emulation data (initiated by the configuration master) no disconnection is made.

Transmission of More than 64 Bytes

At a transmission in a data frame (Msgtyp = $X2_{(16)}$, Data), when the data is not to be transferred to the standard 64 byte receive buffer, the data frame is preceded by a data header (Dh) or information data header (Info Dh) frame directed to the standard receive buffer. In the Info Dh case, e.g. information on where to store the data of the data frame may be included. See Fig. 4.

However, apart from the normal four byte header, both the Dh and Info Dh frames contain:

- The first four bytes of the data to be transferred.
- Two bytes defining the length of the data block (e.g. $0D00_{(16)}$ for 3.25 kbytes of data from the FD)

When the data header frame is acknowledged by the receiving unit the data frame is transferred into a non-standard, user-defined buffer in the receiving unit. The data frame contains:

- The normal four byte header.
- Byte 5 and following of the data to be transferred.

Locally, within the receiving unit, the first four bytes of the data are then transferred to the user-defined buffer so that the whole block of data is available for the addressee.

Communication Examples

The following examples show a number of typical internal communication sequences. Only cluster configurations are regarded. For each example certain assumptions are valid. These are defined in each subsection. Error recovery paths are left out.

If e.g. no answer to a message is received within a certain time, the message is retransmitted once. After another timeout without an answer, the unit is regarded as "off" and the session is aborted together with all sessions involving the failing unit. (Abort frame sent to the involved terminal in connection with polling.) If a Data message gets no answer a Denq message is sent to request the answer, e.g. Ack or Eos.

If e.g. a data frame is too long to be received, the receiving unit may send a Reject frame to the transmitting unit (as a response to a Dh or Info Dh frame). As the transmitting unit sends Eot to the configuration master, a continue session request is queued up in the configuration master and a connection is made when all more prioritized requests have been handled.

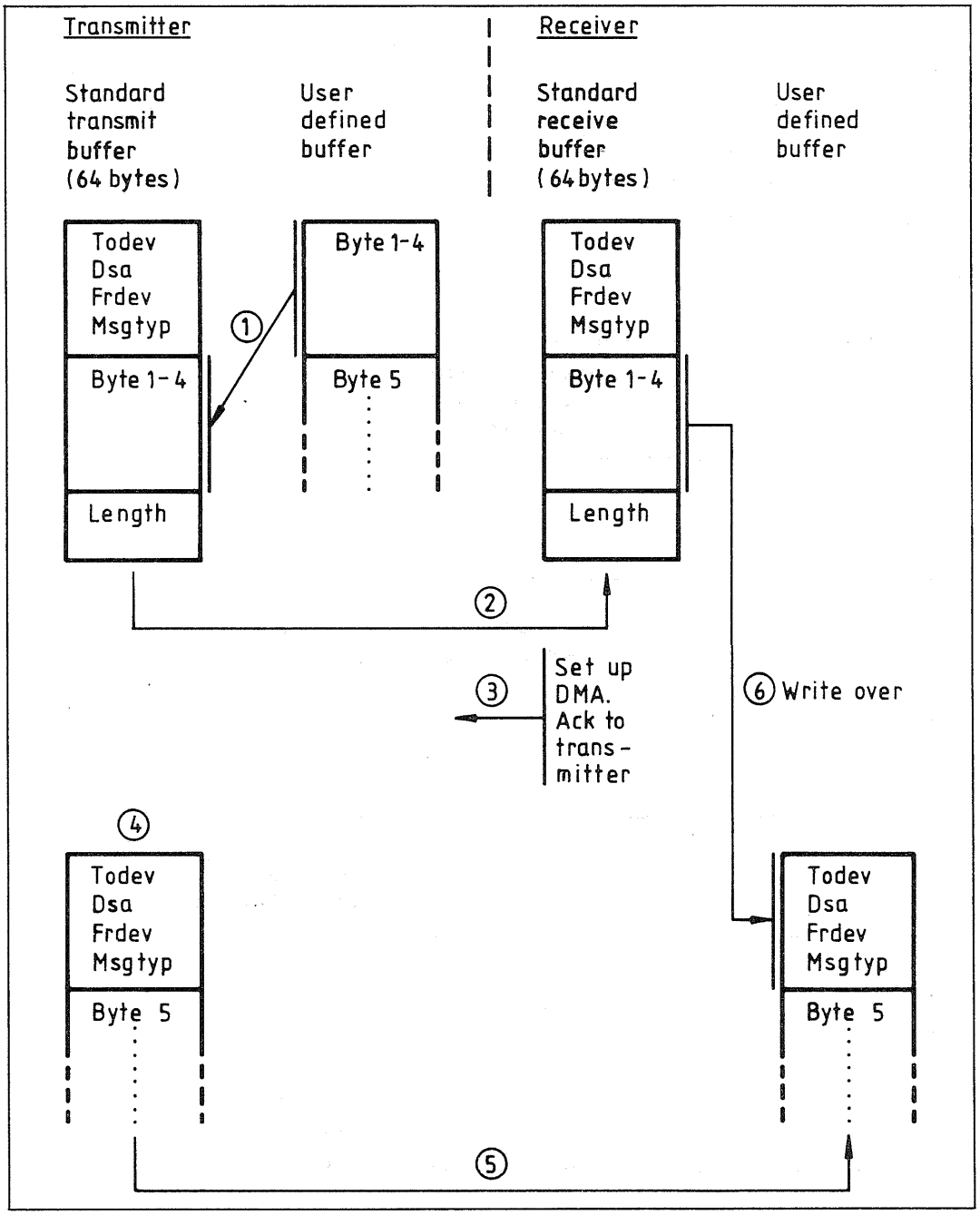
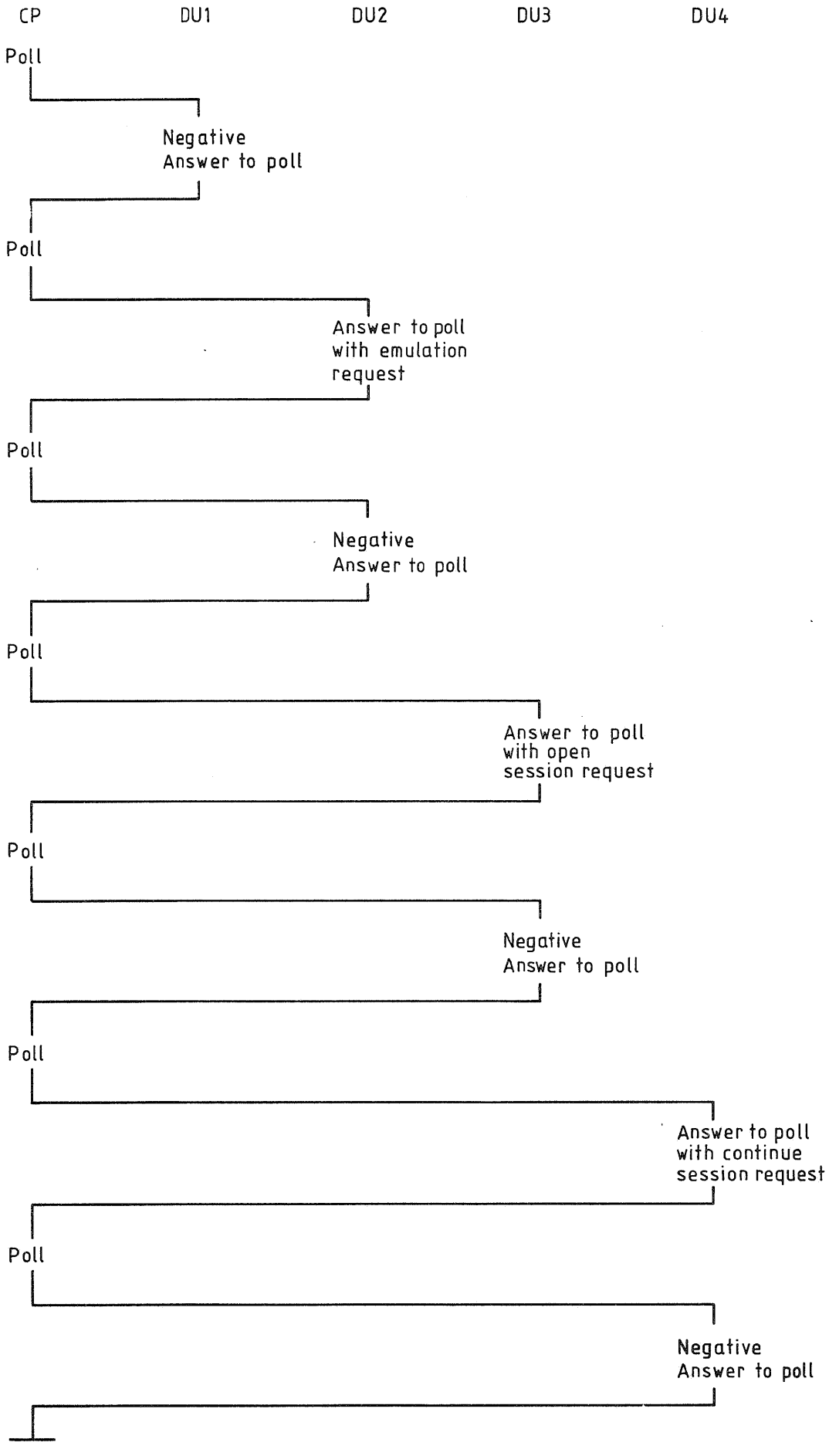


Fig. 4 Data block transfer (more than 60 bytes of data)

Poll and Answer to Poll

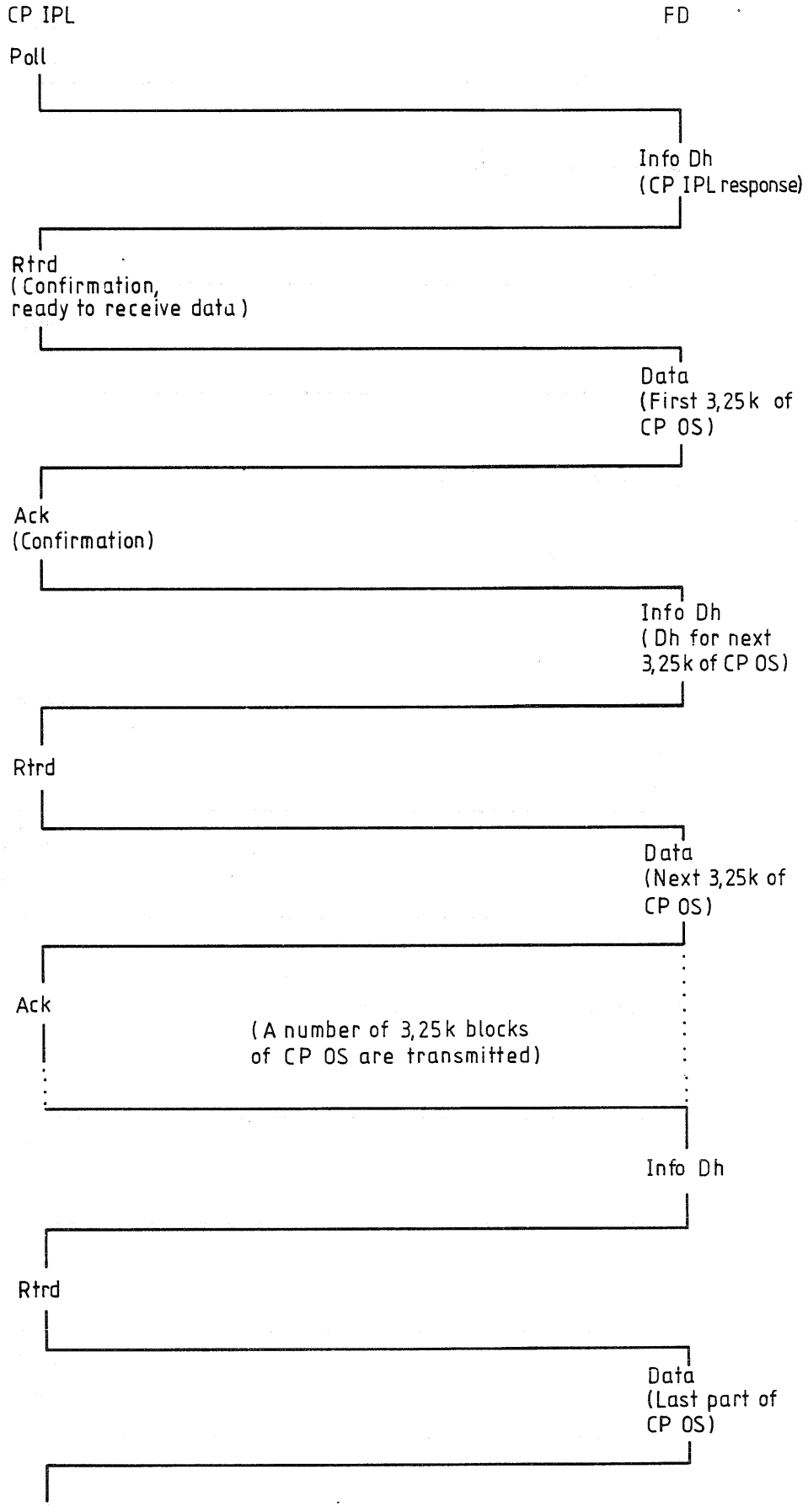
In the example below 4 display units in a cluster configuration are polled. For the different units the following assumptions are made:

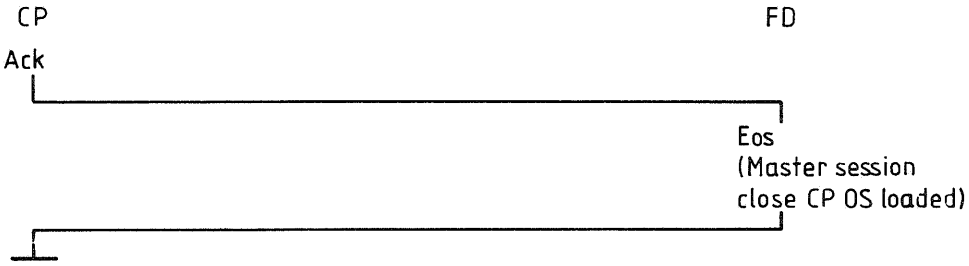
CP	Host communication is possible.
DU1	has got nothing to transmit and gives a negativ poll answer. The emulation is logged on.
DU2	wants to communicate with the host.
DU3	wants to communicate with the system FD and therefore initiates an open session request.
DU4	wants to continue a print session.


CP
DU1
DU2
DU3
DU4
Poll
Negative
Answer to poll
Poll
Answer to poll
with emulation
request
Poll
Negative
Answer to poll
Poll
Answer to poll
with open
session request
Poll
Negative
Answer to poll
Poll
Answer to poll
with continue
session request
Poll
Negative
Answer to poll

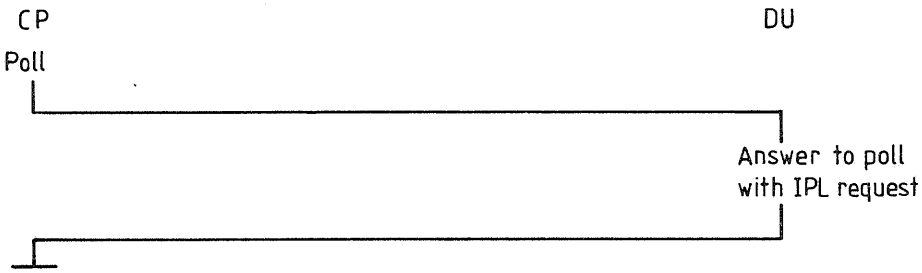
IPL Sessions

The example below shows how a CP and a DU in a cluster configuration do IPL request and how the OS (operating system) is loaded.

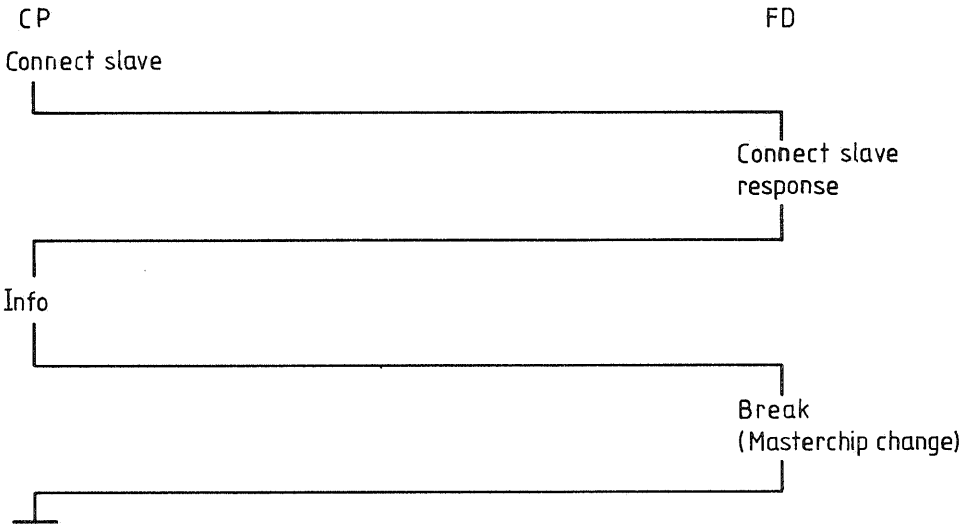




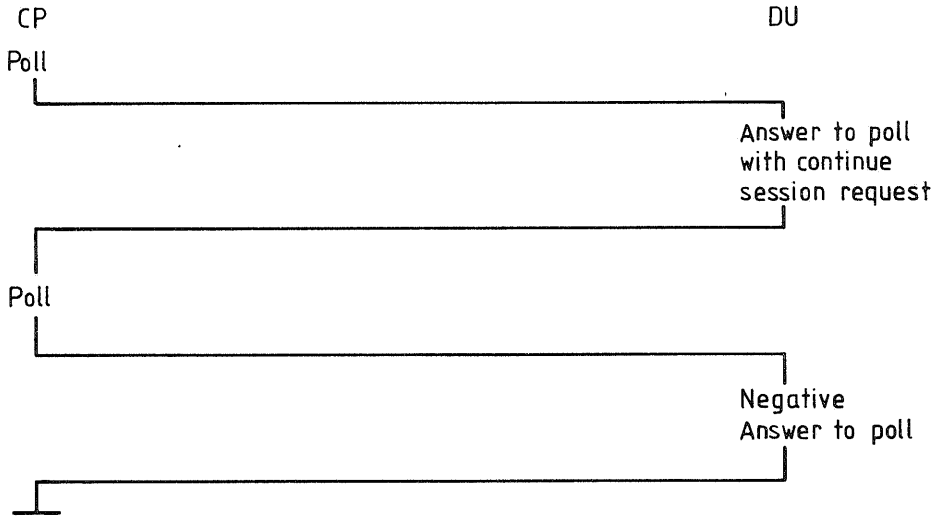
Poll to DU with IPL request

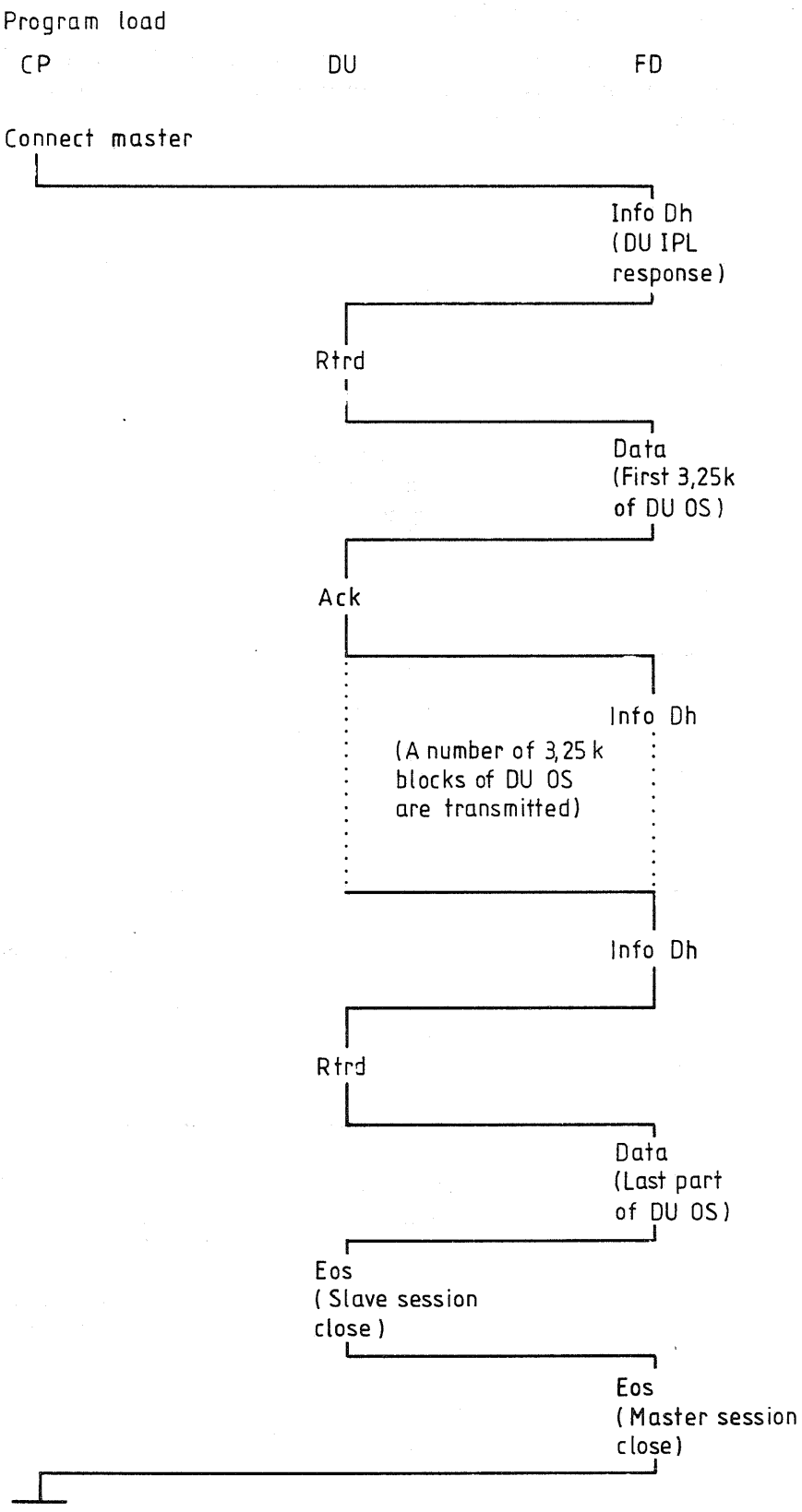


DU IPL session initiation



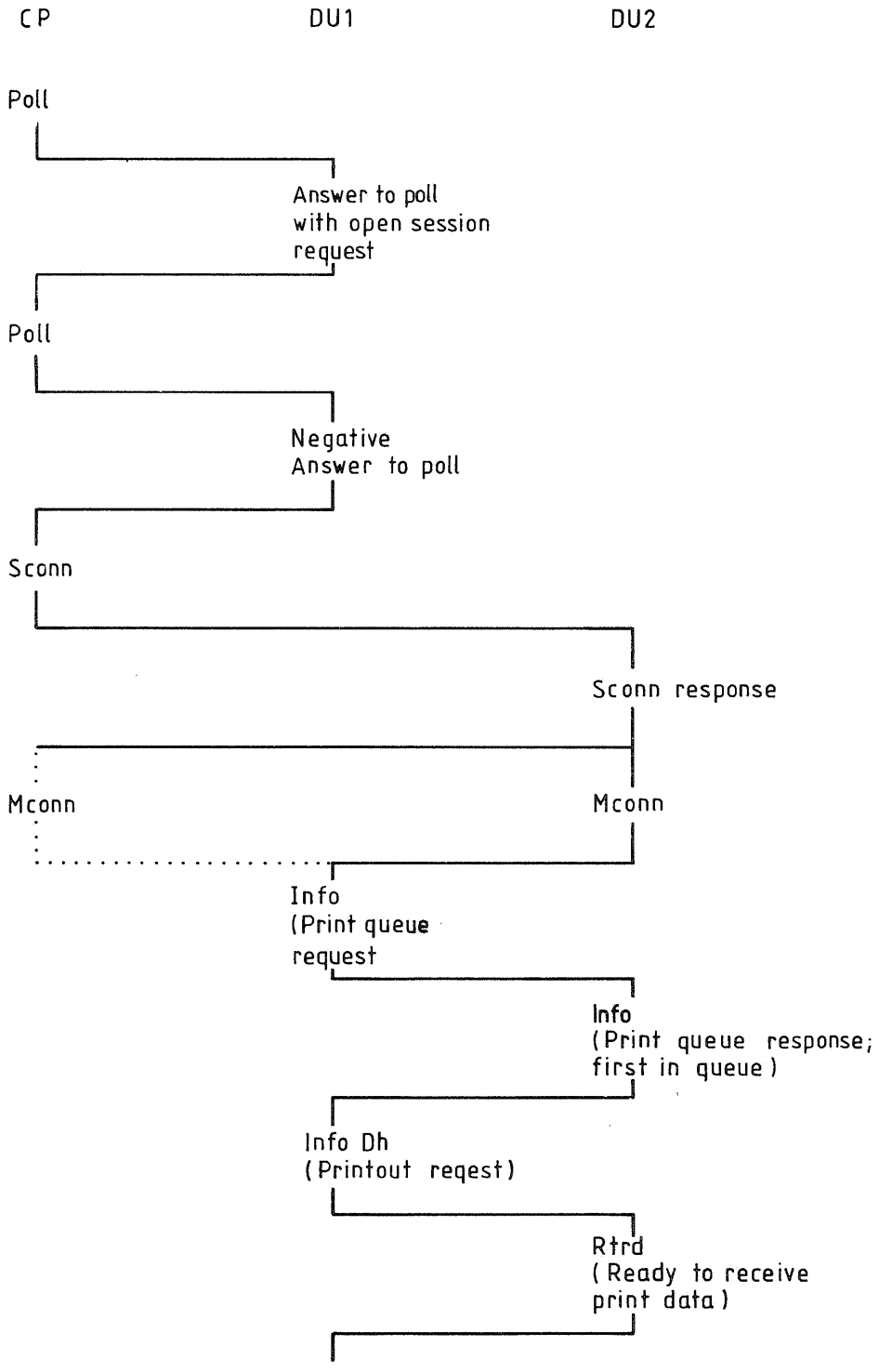
DU IPL session continue request

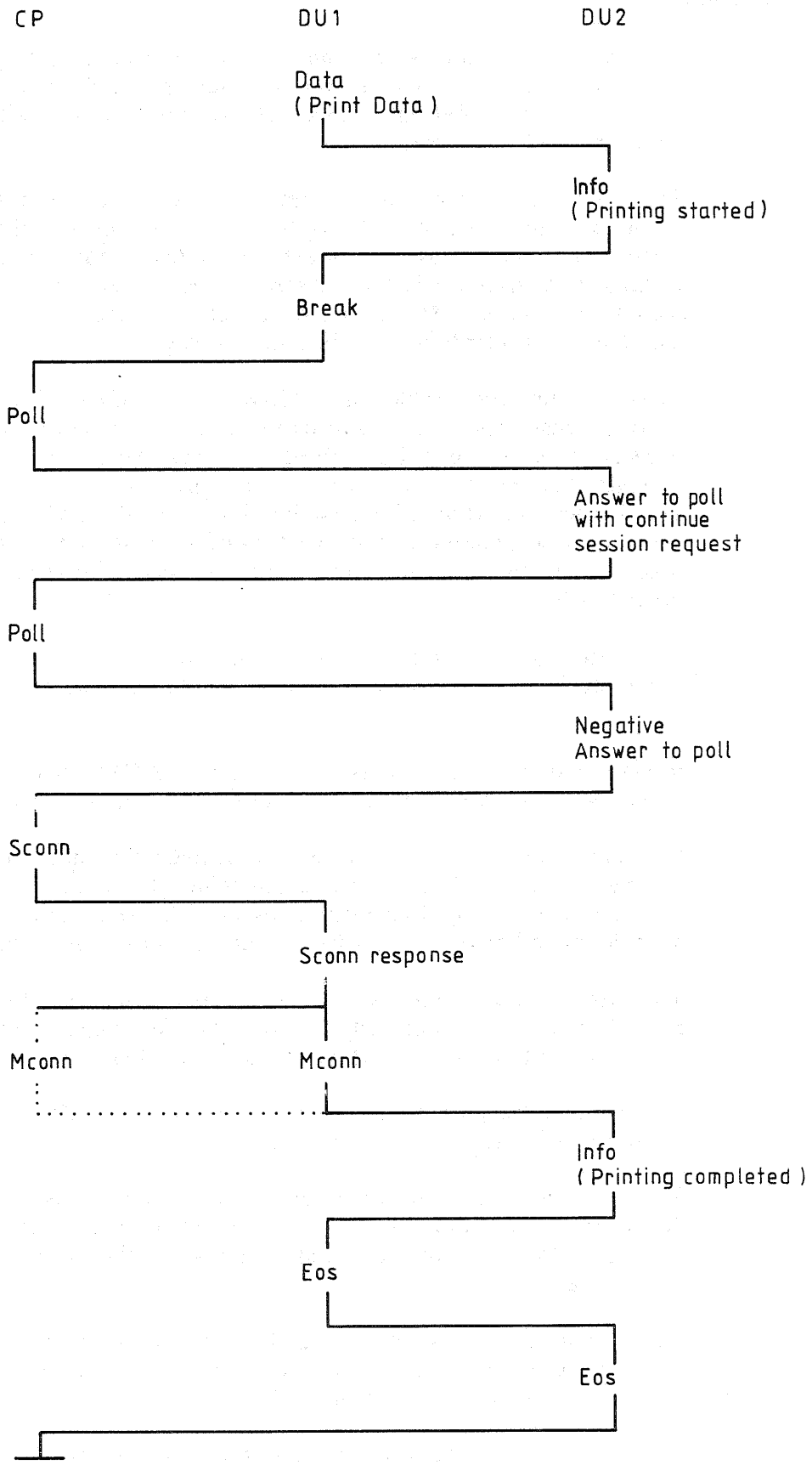



Program load
CP
DU
FD
Connect master
Info Dh
(DU IPL
response)
Rtrd
Data
(First 3,25k
of DU OS)
Ack
Info Dh
(A number of 3,25 k
blocks of DU OS
are transmitted)
Info Dh
Rtrd
Data
(Last part
of DU OS)
Eos
(Slave session
close)
Eos
(Master session
close)

Transfers of Printout Information

The following example shows the communication between a DU1 without a printer and a DU2 with printer in the case of a combined print queue and printout request.




CP
DU1
DU2
Data
(Print Data)
Info
(Printing started)
Break
Poll
Answer to poll
with continue
session request
Poll
Negative
Answer to poll
Sconn
Sconn response
Mconn
Mconn
Info
(Printing completed)
Eos
Eos

Hardware

The internal balanced two-wire communication takes place via programmable ADLC (advanced data link controller) circuits, that perform the conversion between parallel 8-bit data and serially transmitted data with a frame structure of HDLC type.

The data is transferred directly between the memory and the ADLC circuit (without passing the MPU) by help of a programmable DMAC circuit (direct memory access controller). The DMAC controls up to four communication channels. (Byte by byte transfers between the MPU and the ADLC on interrupt basis are also possible. However, existing software uses DMA for transfers for all internal messages.)

The display unit and flexible disk unit have one ADLC each. The communication processors are provided with four ADLCs, each one controlling the transfers of one channel. Up to three units may be connected to the same two-wire (as seen from a communication processor). Thus a selection of a two-wire interface advanced data link controller, ADLC, does not necessarily correspond to the selection of a specific unit. Actually, to reach a specific unit in a cluster configuration, the communication processor has to:

- Set the right address in a crosspoint selection PIA (parallel interface adapter), in order to connect one ADLC to a specific two-wire

  and

- Write a specific address word (into the right ADLC) to be sent on the two-wire and tested by all units connected to that two-wire.

For each channel there are two 64 byte standard memory buffers, one for the last received message and one where the next message to be transmitted is built up. Furthermore users of the communication system may define other data areas for reception and transmission of data.

The binary data on the two-wire is frequency shifted. The transfer rate is 300 kbits/s, i.e. nominally 37.5 kbytes/s. The main functional blocks common to all two-wire interfaces are shown in Fig. 5.

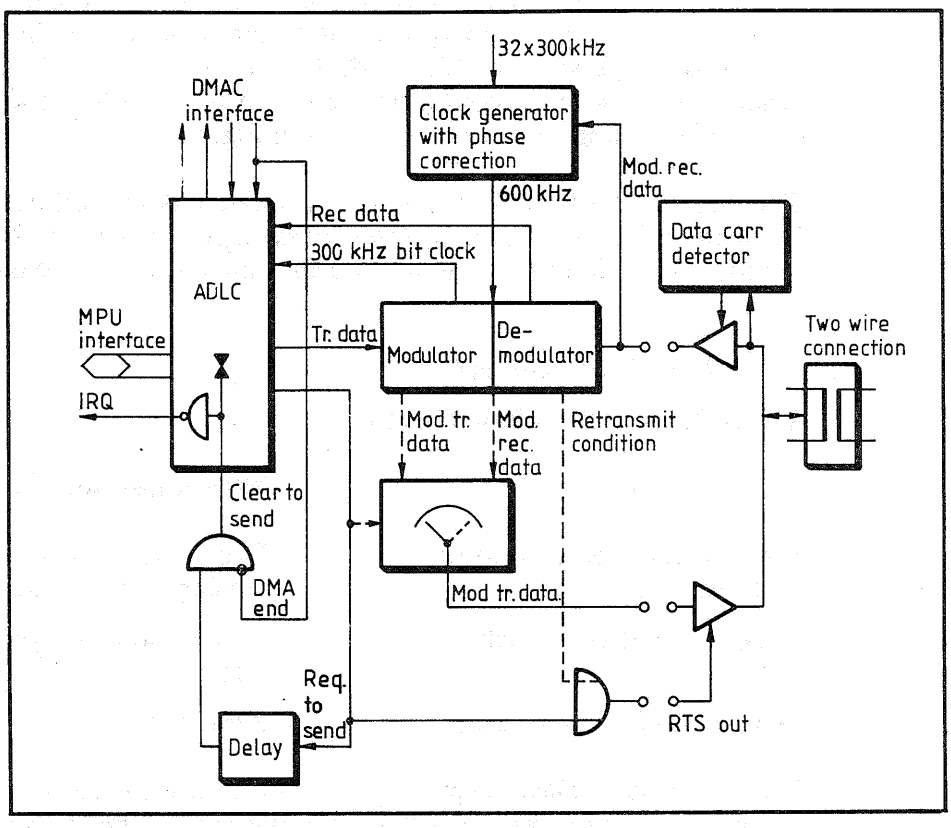
The ADLC is the parallel/serial converter between the MPU bus and the serial line using synchronous transmission.

The clock generator with phase correction is needed as the clocks of Alfaskop System 41 are local. The phase correction circuitry compares the edges of received data to the state (phase) of the clock and corrects the clock accordingly.

Delay of Request to send is needed to allow phase correction at the receiving terminal. (Before Clear to send is active only idles ones are shifted out from the transmitting ADLC.)

The data carrier detector discriminates against low level interference.

The modulator/demodulator produces frequency shifted data for the two-wire and transforms two-wire data to NRZ (no return to zero) data for the ADLC. The demodulator part also transforms continuous erroneous data to an abort pattern for the ADLC.



Note: Dashed lines and area between amplifiers and modulator/demodulator indicate communication processor functions.

Fig. 5 Principles of two-wire interface

Modulation Method

The ADLC is programmed to provide and accept binary data of NRZ format. (One level during the whole bit time represents a data bit.) Data on the two-wire cables are frequency shifted and sent as a balanced square wave. A logical 1 is represented as one pulse during one whole bit time and a logical 0 is represented as one full square wave period during one bit time. Thus: 1 = half period of 150 kHz square wave, 0 = full period of 300 kHz.

Clock Generation and Phase Correction

The clock generator contains a divide by 16 counter. See Figs 5 and 6. In case no incoming data is detected, a 600 kHz ($\frac{9.585}{16}$ MHz) signal is

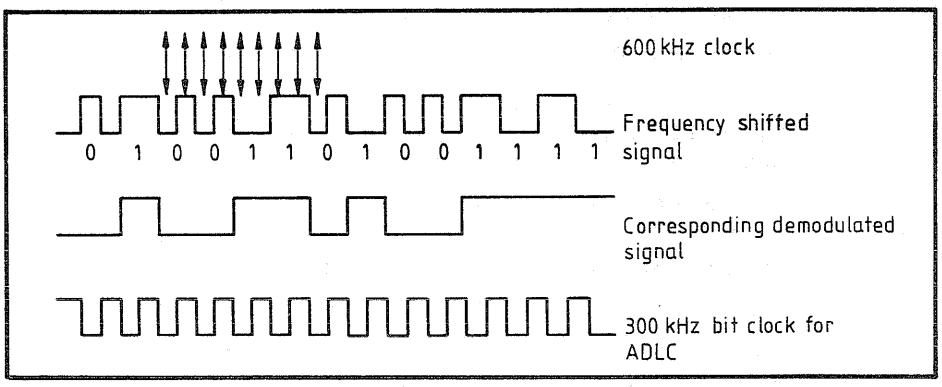


Fig. 6 Demodulation

provided to the modulator/demodulator. In case Modulated received data is present, (e.g. all ones preceding a frame) the counter intermittently functions as a divide by 17, divide by 16, or divide by 15 counter to synchronize the 600 kHz (carry) clock to the middle of the shortest received (300 kHz) pulses (see arrows in Fig. 6).

The phase of the 300 kHz bit clock may still be 180° wrong but is corrected in the modulator/demodulator to provide a positive-going shift-in clock for the ADLC in the middle of the demodulated Received data period.

Advanced Data Link Controller, ADLC

The ADLC includes functions for:

Receiver section	**Transmitter section**
Flag detection, synchronization and stripping	Flag appending (opening and closing flags of a message)
Cyclic redundancy check, CRC	Calculation and appending of CRCC
Deletion of (inserted) zeros	Zero insertion

Further provided: DMA request signals, modem interface, loop configuration possibilities, variable word length of information field, full duplex operation, NRZ or NRZI format of received/transmitted data bits.

It contains four (8-bit) control registers, two status registers, transmit and receive registers, accessible for the MPU.

ADLC External Signals

The ADLC interfaces the MPU 8-bit data bus, a chip enable line (CS), two address lines (RS1, RS0) and a R/W line, used for addressing internal registers together with an address control bit, an enable line (E, typically Ø2, system clock) and a reset line (R). See Fig. 7. An interrupt request output (IRQ) is also provided. The interrupt sources may be internally masked and given priorities. The register addressing is shown below:

Selected function	**RS1**	**RS0**	**R/W**	**Address Control Bit (CR_1b_0)**
Write control register 1	0	0	0	X
Write control register 2	0	1	0	0
Write control register 3	0	1	0	1
Write transmitter FIFO (Frame continue)	1	0	0	X
Write transmitter FIFO (Frame terminate)	1	1	0	0
Write control register 4	1	1	0	1
Read status register 1	0	0	1	X
Read status register 2	0	1	1	X
Read receiver FIFO	1	X	1	X

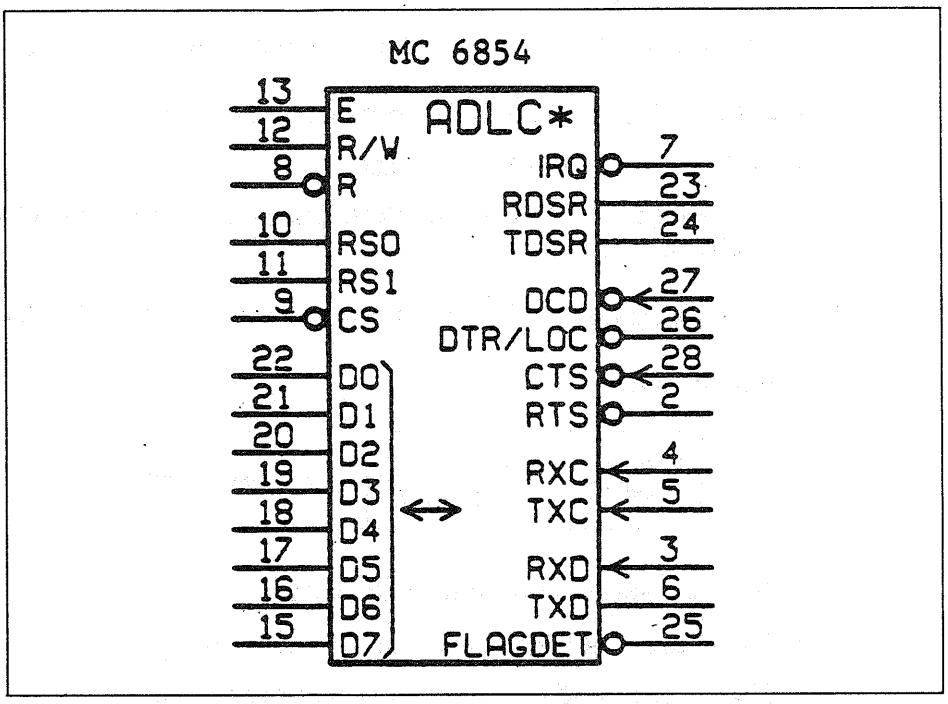


Fig. 7 ADLC drawing symbol

The receiver section lines are:

- RXC and RXD, clock and data inputs.
- Flag detect output, always low for one bit time when flags $(7E)_{16}$ are received.
- DCD, Data carrier detected input. The receiving section is inhibited when this input is high.
- RDSR, Receiver data service request output, may be used as a DMA request from the receiver section. RDSR reflects the value of the status register bit RDA (Receiver data available). Priority status should be used.

The signals used when transmitting are:

- TXC and TXD, clock input and data output.
- RTS and CTS, Request to send to e.g. a modem and Clear to send, answer that e.g. a modem is ready for sending.
- DTR/LOC, Data terminal ready/Loop on line control. For the point-to-point configuration mode, DTR follows what is written in an associated control register bit (DTR/LOC bit). For the loop mode, DTR/LOC set will make the terminal go on the loop and activate the DTR/LOC output after receiving 7 ones. When the control bit is reset and 8 successive ones have been received from up the loop, DTR/LOC output is deactivated (goes high).
- TDSR, Transmit data service request output, may be used as a DMA request from the transmitter section. TDSR reflects the value of status register bit Transmitter data register available/Frame complete if programmed to work as TDRA. Priority status should be used.

ADLC Control Registers

The following abbreviations are used under this and the next headings:

- CR1, 2, 3, and 4 stands for control register 1, 2, 3, and 4.
- SR1 and 2 stands for status register 1 and 2.
- AC, address control, is an internal address bit found in CR1:0 (bit 0 of CR1).

The following format definitions are used for the ADLC circuit:

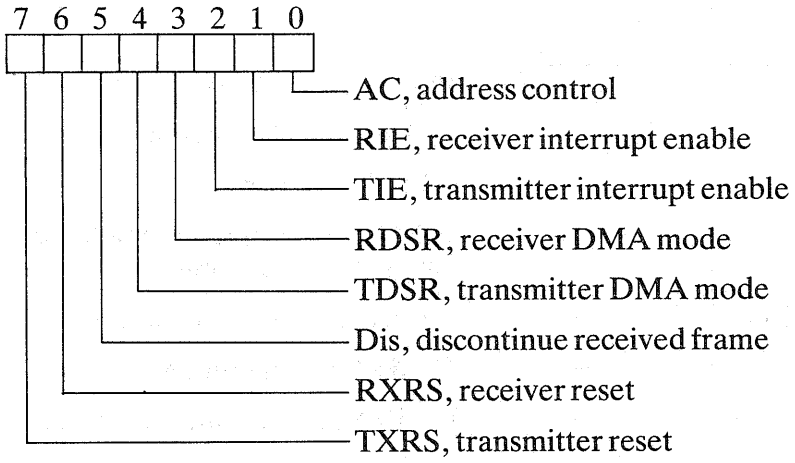
Frame format:

Flag	Address field	Control field	Logical control field	Information field	CRCC	Flag

- $7E_{(16)}$ is generated or identified as flag.
- The address field can consist of one or more bytes.
- The control field can consist of one or two bytes.
- The logical control field can consist of none, one or more bytes.
- The information field can consist of an arbitrary number of 5, 6, 7, or 8 bits words.
- The CRCC (Cyclic redundancy check character) consists of 16 bits.

An arbitrary number of flags ($7E_{(16)}$) can be transferred between frames of the above format. The end flag can be the start flag of the next frame. Between end and start flags can instead of flags, as time fill, idling ones (a number of $FF_{(16)}$, called marks) be transferred. More than 6 ones within a frame is interpreted as an abort pattern.

Control register 1. Address $000X_{(2)}$ (= RS1, RS0, R/W, AC)



A set bit will give the following function:

b7 TXRS. All ones are transmitted. The transmitter FIFO register cannot be loaded. The transmitter status bits are reset, although clear to send (SR1:4) follows the CTS input. Set by Reset input low or write operation. Reset by write operation after that Reset input has gone high.

b6 RXRS. The receiver FIFO register and the receiver status bits are reset, though data carrier detect (SR2:5) follows the DCD input. Set by Reset input low or write operation. Reset by write operation after that the Reset input has gone high.

b5 Dis. Further received bytes will not interrupt the MPU or set receiver data available (SR1:0 or SR2:7). Set by write operation. Reset after the last discarded byte.

b4 TDSR. Transmitter data register available (SR1:6), will not interrupt the MPU but set the TDSR output.

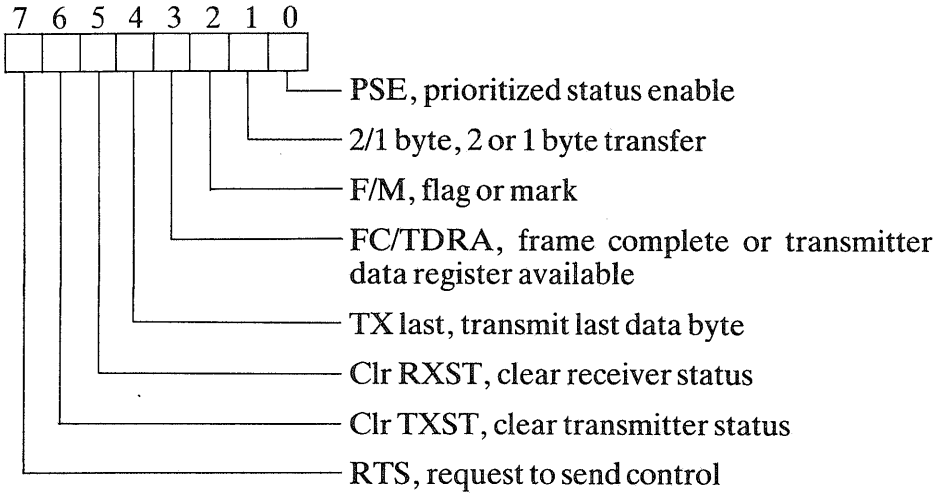
b3 RDSR. Receiver data available (SR1:0), will not interrupt the MPU but set the RDSR output.

b2 TIE. The IRQ status (SR1:7) and the IRQ output will be activated by transmitter status.

b1 RIE. The IRQ status (SR1:7) and the IRQ output will be activated by receiver status.

b0 AC. Control register 3 or 4 is selected.

Control register 2. Address $0100_{(2)}$ (= RS1, RS0, R/W, AC)



The bits are further explained below:

b7 RTS. 1 = the RTS output is latched low (active). 0 = the RTS output returns high when a frame is completed.

b6 Clr TXST. 1 = reset transmitter status bits. This control bit is then automatically reset.

b5 Clr RXST. 1 = reset receiver status bits set before last read except receiver data available (SR1:0 and SR2:7) and address present (SR2:0). This control bit is then automatically reset.

b4 TX last. 1 = the data byte, loaded in the FIFO register immediately before this bit is set, is the last one in the frame.

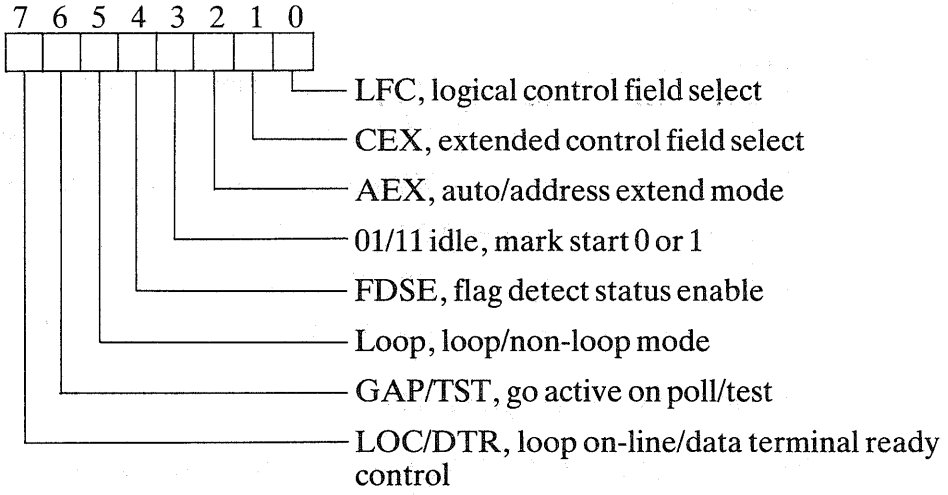
b3 FC/TDRA. 1 = SR1:6 indicates when the last bit of a frame is transmitted. No IRQ or TDSR on transmitter data register available (SR1:6) condition. 0 = SR1:6 indicates when the transmitter FIFO register is available.

b2 F/M. 1 = flags ($7E_{(16)}$) are transmitted between frames. 0 = marks ($FF_{(16)}$) are transmitted between frames.

b1 2/1 byte. 1 = transmitter data register available (SR1:6) or receiver data available (SR1:0 and SR2:7) and associated signals tell that two bytes of data can be read or written in the FIFO registers. 0 = status tells that one byte can be read or written.

b0 PSE. 1 = status priority: $\overline{CTS}$ > TXU > TDRA/FC and FD > S2RQ > RDA and (for SR2) others > RX idle > AP > RDA. 0 = no status suppresses another, but $\overline{CTS}$ which always suppresses TDRA bit.

Control register 3. Address $0101_{(2)}$ (= RS1, RS0, R/W, AC)



The bits are further explained below:

b7 LOC/DTR. In point to point mode (CR3:5 = 0): 1 = DTR output active. 0 = DTR output inactive.

In loop mode (CR3:5 = 1): 1 = the ADLC goes to the on-line state after 7 consecutive ones at the RXD input. 0 = the ADLC goes to the off-line state after 8 consecutive ones at the RXD input.

b6 GAP/TST. In point to point mode (CR3:5 = 0): 1 = the TXD output is internally connected to the RXD input. 0 = normal operation.

In loop mode (CR3:5 = 0): 1 = the receiver searches for "Go ahead" (or End of poll, EOP). The "Go ahead" is then converted to a start flag and transmitting is started. 0 = a transmitting in progress is completed, the "go-active-on-poll" operation is completed and the received data to transmitted data is reestablished. The ADLC then returns to the "loop-on-line" state.

b5 Loop. 1 = loop mode. 0 = point to point mode.

b4 FDSE. 1 = flag detect (SR1:3) and IRQ (if enabled by receiver interrupt enable (CR1:1)) tells if a flag is received. 0 = the above function is disabled but the FLAGDET output still works.

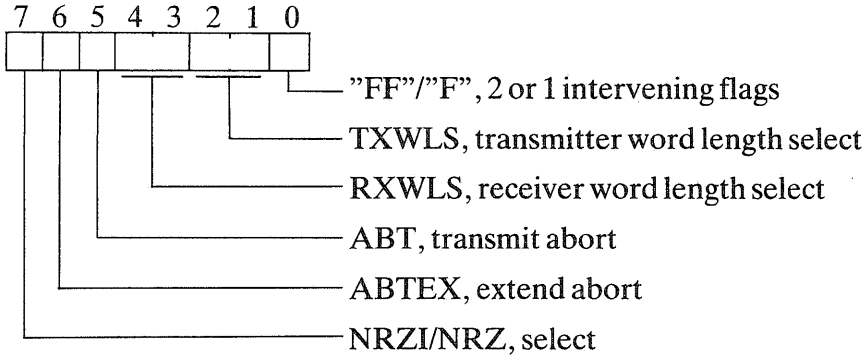
b3 01/11 idle. (If flag or mark (CR2:2) = 0.) 1 = inactive idling (marks) starts with 0, i.e. $7FFF..._{(16)}$. 0 = all ones are transmitted when idling, i.e. $FFFF..._{(16)}$.

b2 AEX. 1 = if bit 0 of an address byte is 0 the next byte is also an address byte unless it is the first address byte and = $00_{(16)}$. 0 = only one address byte exists.

b1 CEX. 1 = 2 bytes in the control field, 0 = 1 byte in the control field.

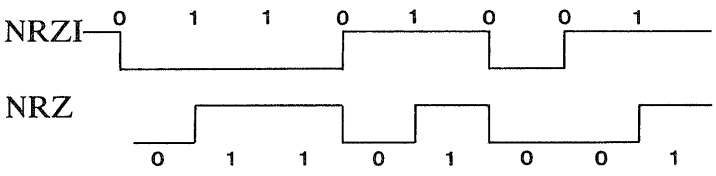
b0 LFC. 1 = logical control field follows control field. The field is extendable as long as bit 7 in each byte equals 1. 0 = no logical control field is used.

Control register 4. Address $1101_{(2)}$ (= RS1, RS0, R/W, AC)



The bits are further explained below:

b7 NRZI/NRZ. 1 = NRZI mode both for transmitted and received data. (One bit-time delay to transmitted data.) 0 = NRZ mode both for transmitted and received data.



b6 ABTEX. 1 = abort pattern is transmitted (due to transmitter underflow (SR1:5) or transmit abort (CR4:5) until this bit is reset. When reset the transmitter works according to flag or mark (CR2:2) and mark start 0 or 1 (CR3:3).

b5 ABT. When set 8 ones are transmitted. The action thereafter is dependent on extend abort (CR4:6). This bit is automatically reset when abort transmitting has started.

b4, b3 RXWLS. The word length in the received information field is defined by these two bits:

$00_{(2)}$ = 5 bits $01_{(2)}$ = 6 bits $10_{(2)}$ = 7 bits $11_{(2)}$ = 8 bits

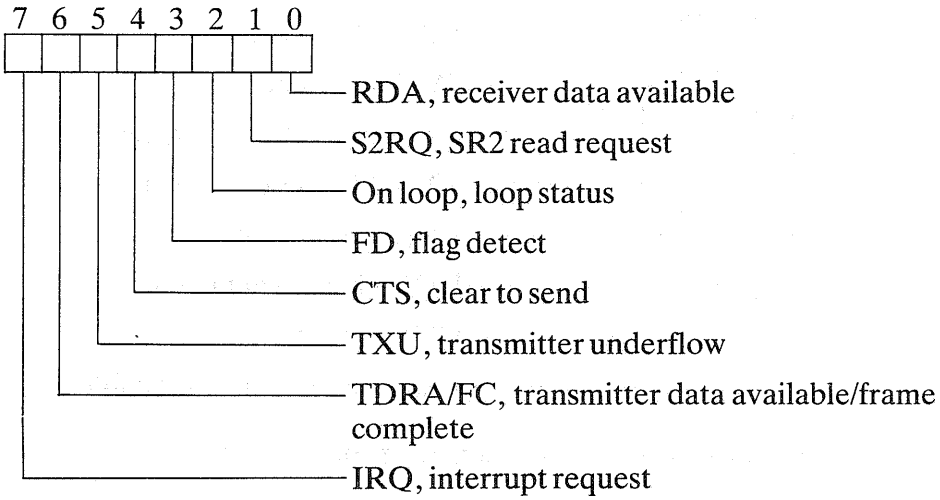
b2, b1 TXWLS. The word length in the transmitted information field is defined by these two bits:

$00_{(2)}$ = 5 bits $01_{(2)}$ = 6 bits $10_{(2)}$ = 7 bits $11_{(2)}$ = 8 bits

b0 "FF"/"F". 1 = the end flag is followed by a start flag if the next frame is transmitted immediately after the first. 0 = the end flag of a frame works as the start flag of an immediately following frame.

ADLC Status Registers

Status register 1. Address $001X_{(2)}$ (= RS1, RS0, R/W, AC)



The bits are further explained below:

b7 IRQ. This bit is set and the IRQ output is activated if any status bit except loop (SR1:2) is set depending on the following conditions:

- Reset input, transmitter reset (CR1:7), and receiver reset (CR1:6).
- Transmitter interrupt enable (CR1:2) and receiver interrupt enable (CR1:1).
- Discontinue received frame (CR1:5), transmitter DMA mode (CR1:4), receiver DMA mode (CR1:3), prioritized status enable (CR2:0), and flag detect status enable (CR3:4).

b6 TDRA/FC. Depending on frame complete or transmitter data register available (CR2:3) and 2 or 1 byte transfer (CR2:1) this bit is set if 1 or 2 bytes can be written in the transmitter FIFO register or if flags or abort are transmitted. This bit is reset by transmitter reset (CR1:7).

b5 TXU indicates an aborted frame. This bit is reset by transmitter reset (CR1:7) or clear transmitter status (CR2:6).

b4 CTS is set if the CTS input becomes inactive. After this bit is reset by transmitter reset (CR1:7) or clear transmitter status (CR2:5) it reflects the state of the CTS input.

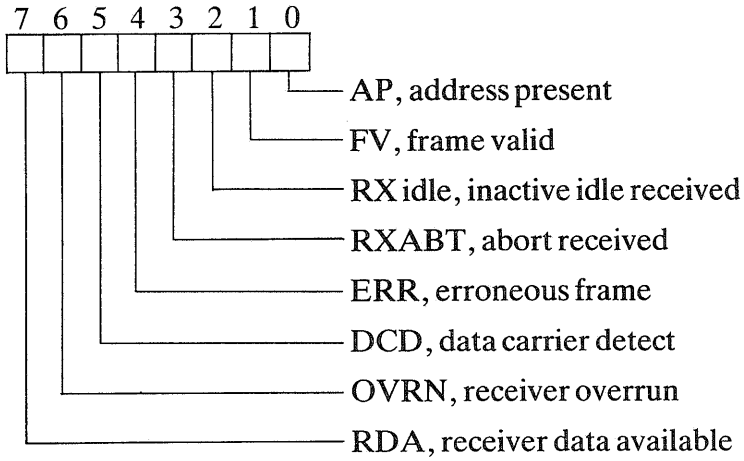
b3 FD indicates that a flag has been received if flag detect status enable (CR3:4) is set. This bit is reset by receiver reset (CR1:6) or clear receiver status (CR2:5).

b2 On loop is set when on loop in loop mode (CR3:5 = 1) but does not activate IRQ. Otherwise this bit is always 0.

b1 S2RQ is set when a bit, except receiver data available (SR2:7), has been set in SR2. This bit is reset by receiver reset (CR1:6) or when applicable SR2 bit(s) is (are) reset.

b0 RDA. When this bit is set 1 or 2 bytes, depending on 2 or 1 byte transfer (CR2:1) may be fetched from the receiver FIFO register (see also SR2:7).

Status register 2. Address $011X_{(2)}$ (= RS1, RS0, R/W, AC)



The bits are further explained below:

b7 RDA. Same as SR1:0.

b6 OVRN indicates that one or more bytes of a received frame is overwritten and lost. The receiver FIFO register 2 and 3 are not affected even by continued overrunning. This bit is reset by receiver reset (CR1:6) or clear receiver status (CR2:5).

b5 DCD is set if the DCD input becomes inactive. After this bit is reset by receiver reset (CR1:6) or clear receiver status (CR2:5) it reflects the state of the DCD input.

b4 ERR is set at CRC error or if the address or control fields are incomplete. It is set at the same time but instead of frame valid (SR2:1).

b3 RXABT is set when 7 or more consecutive ones are received within a frame. This bit is reset when 15 or more consecutive ones are received or by receiver reset (CR1:6) or clear receiver status (CR2:5).

b2 RX idle is set when 15 or more consecutive ones are received. This bit is reset by receiver reset (CR1:6) or clear receiver status (CR2:5) if idling has stopped.

b1 FV indicates that a correct message has been received and that the last data byte may be read by the MPU. This bit is reset by receiver reset (CR1:6) or clear receiver status (CR2:5).

b0 AP indicates that an address byte may be fetched from the receiver FIFO register. This bit is reset by a reading of the receiver FIFO register or by receiver reset (CR1:6).

Functional Examples

See Fig. 5.

Transmission

Suppose that the direct memory access controller (DMAC) is initialized. The Request to send signal is activated as the RTS control bit of the ADLC is set by the program. Then all ones (modulated) are sent on the two-wire to enable phase correction at the receiving end. After a time, Clear to send will be signalled to the ADLC. The ADLC in its turn will then be able to tell the DMAC that its transmitter register is available.

A byte will then be transferred from the memory to the ADLC under the control of the DMAC. Following the flag byte, that byte is modulated and transferred on the two-wire and the MPU may be busy with other tasks.

When the last information byte is loaded into the ADLC, the DMA end signal is activated. This disables Clear to send which may be signalled by an IRQ to the MPU. The MPU could then reset the request to send control bit. After the ADLC has appended CRCC and flag characters to the message, the Request to send signal is automatically deactivated.

Reception

Suppose that the appropriate channel of the DMAC has been initialized for reception and receiver DMA mode has been set in the ADLC. After phase correction on received idle ones and when a flag and one more character have been demodulated and shifted into the ADLC, the first DMA takes place. When a whole frame has been received, the MPU is interrupted and may read the status of the ADLC to investigate whether the frame was correct or erroneous.

Note that the regeneration of Modulated received data on the Modulated transmitted data line and the generation of an RTS out signal when Received data is correct (Retransmit condition) only occur in the communication processors.

Internal Communication with Printers

A printer unit can be connected to a display unit. The communication takes place via a CCITT V.24/V.28 interface.

The communication protocol is described in the Service Manual for the applicable printer. The communication circuitry and hardware interface in the display unit is described in the Asynchronous Communication Adapter chapter. The corresponding information about the printer side is found in the Service Manual.

The LSI circuit (ACIA) that handles the communication in the communication adapter is also used in the keyboard interface. It is therefore described below in order to avoid double information.

Asynchronous Communication Adapter, ACIA

The asynchronous communication interface adapter, ACIA, is used to adapt serial communication lines (where bytes are transmitted asynchronously, using start and stop bits) to the microcomputer bus.

Signals

The ACIA interfaces the MPU by (see Fig. 8) the data bus (D7 – 0), three chip select lines (CS0 – 2), an enable line (E), generally tied to Ø2, which enables the output or input data buffers if the chip is selected, a R/W and one register address line (RS). The IRQ line is used to signal to the MPU that received data may be read or transmit data may be written in the ACIA. Interrupt is also given when overrun or loss of data carrier (DCD input high) occurs during reception.

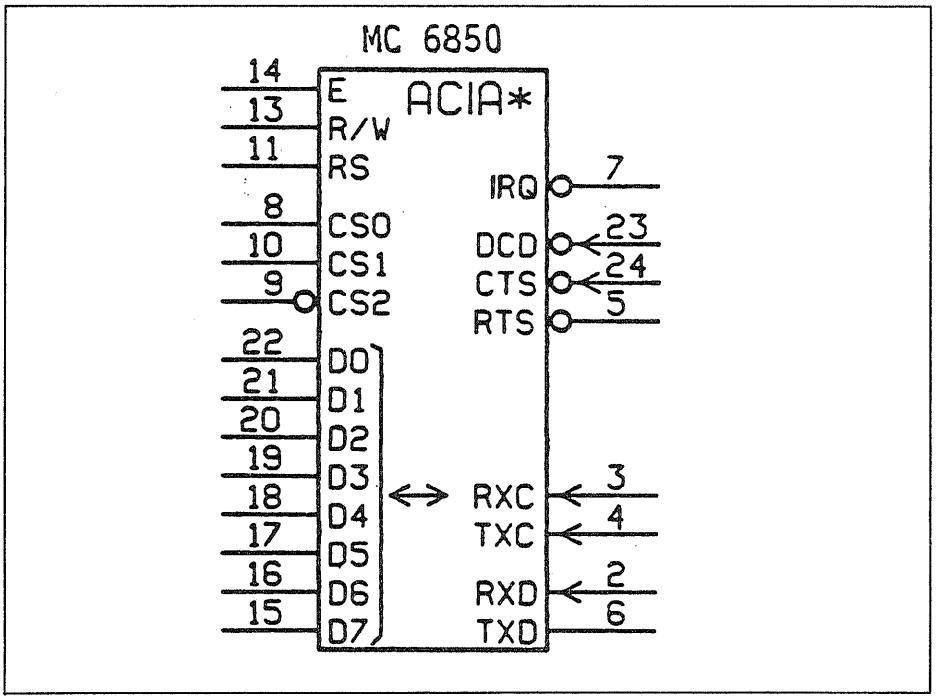


Fig. 8 ACIA drawing symbol

The IRQ for the receiver and transmitter sections may be masked by setting bits in an internal control register or by resetting the ACIA via the control register.

The peripheral interface lines are, for the receiver section:

- RXD, Received data input.
- RXC, Receiver clock input. Received data bits are shifted into the ACIA on the positive edge of RXC when clock rate = bit rate. In this case clock and data have to be synchronized externally. The clock rate may be selected (by control register bits) to be 16 or 64 times the bit rate. In this case the ACIA synchronizes internally.
- DCD, Data carrier detected input. When this input is inactive (high), the receiver section of the ACIA is inhibited and initialized. DCD causes IRQ if receiver interrupt is enabled.

The transmitter section peripheral lines are:

- TXD, Transmitted data output.
- TXC, Transmitter clock input. Transmitted data bits are shifted out from the ACIA on the negative edge of TXC if clock rate = bit rate. The clock rate may be selected together with RXC clock rate to be 16 or 64 times the bit rate.
- RTS, Request to send output. This output may e.g. be used to initiate a modem for sending. The signal is controlled by two control register bits. -
- CTC, Clear to send input. This input may e.g. be used to tell the ACIA and the MPU that a modem is ready for sending. When inactive (high) the status bit TDRE (transmitted data register empty) and the possible resulting IRQ are inhibited.

Asynchronous Communication

As bytes may be transmitted asynchronously, a start bit has to be sent first to tell the receiving equipment that a byte is coming. When no character is transmitted, there is a high level on the line.

A low level is then transmitted during one bit time, i.e. the start bit. See Fig. 9. Then the data byte is transmitted with the least significant bit first. A parity bit may follow. The sequence is closed by transmitting one or two stop bits (high level) followed by idling (high level) or next character. Note that the signal levels mentioned above are valid for positive logic only (as on ACIA pins RXD and TXD).

Word length (7 or 8 bits), use of parity bits and number of stop bits are selected by control register bits.

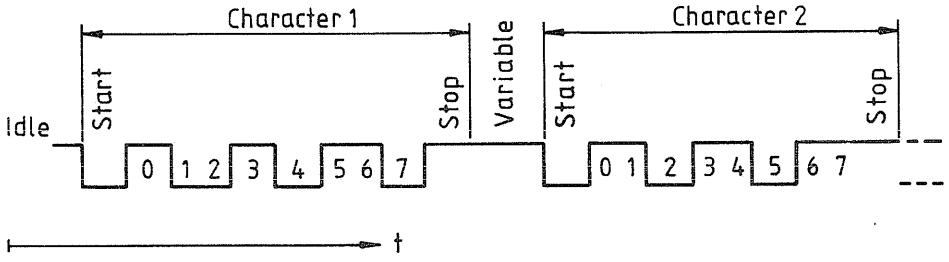


Fig. 9 Asynchronous transmission, character structure

Registers

The ACIA contains four registers that are available from the data bus. A register is selected by the address line (RS) and the R/W line:

	R/W = 0 (write only)	**R/W = 1 (read only)**
RS = 0	Control register	Status register
RS = 1	Transmitted data register	Received data register

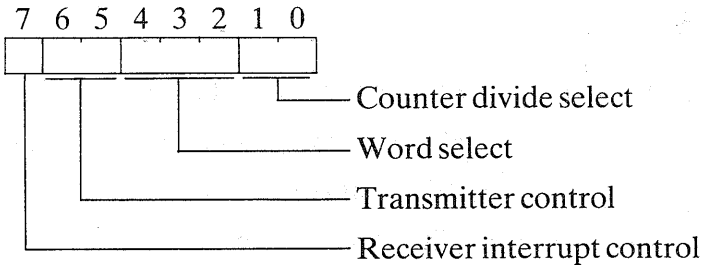
Transmitted data register, stores 8 bits of data to be transmitted from the TXD output. Received data register stores 8 bits of data, received on the RXD input.

While one byte is being shifted out from the ACIA, another may be loaded into the transmitted data register.

In the same way, when a byte is shifted in, it is automatically transferred to the received data register from where it should be read within one byte time, before the next byte is fully received.

Control Register

The control register has the following format:



The bits are further explained below:

b7 Receiver interrupt control. 1 = IRQ at received data register full (RDRF), overrun or low to high transition on the DCD input. 0 = receiver interrupt disabled.

b6
b5 Transmitter control:

b6	b5	Function
0	0	RTS output is low. Transmitter interrupt is disabled.
0	1	RTS output is low. Transmitter interrupt is enabled.
1	0	RTS output is high. Transmitter interrupt is disabled.
1	1	RTS output is low. Transmitter interrupt is disabled. A low level is transmitted on the TXD output.

b4
b3
b2 Word select:

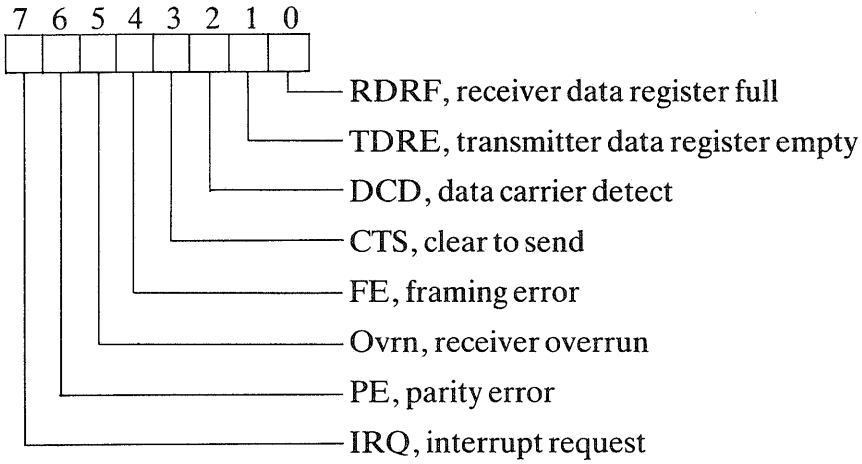
b4	b3	b2	Data bits	Parity bit	Stop bits
0	0	0	7	Even	2
0	0	1	7	Odd	2
0	1	0	7	Even	1
0	1	1	7	Odd	1
1	0	0	8	None	2
1	0	1	8	None	1
1	1	0	8	Even	1
1	1	1	8	Odd	1

b1
b0 Counter divide select. These bits determine the clock divide ratios utilized both in the transmitter and receiver. The receiver clock and data must be externally synchronized in the divide by one case. These bits provides also a master reset which must reset the ACIA before initialization:

b6	b5	Function
0	0	Divide clock by 1.
0	1	Divide clock by 16.
1	0	Divide clock by 64.
1	1	Master reset.

Status Register

The status register has the following format:



The bits are further explained below:

b7 IRQ. 1 = IRQ output low, i.e. at receiver data register full, transmitter data register empty, overrun or DCD input inactive. The bit is reset by master reset, reading receiver or writing transmitter data registers.

b6 PE is set as long as a data character with parity error is resident in the receiver data register and master reset is inactive.

b5 Ovrn. 1 = one or more data characters not read in time were lost. The Ovrn is not set until the valid character prior to overrun has been read. Ovrn is reset by readings of the receiver data register or by the master reset.

b4 FE. 1 = the first stop bit is absent in a received character. FE is reset by readings of the receiver data register or by the master reset.

b3 CTS, follows the CTS input. Transmitter data register empty is inhibited when the CTS input is inactive.

b2 DCD, goes high when the DCD input goes, or is, inactive after DCD reset. DCD is reset by first reading the status register and then the receiver data register or by the master reset after that the DCD input has gone active.

b1 TDRE. 1 = new data may be entered into the transmitter data register. TDRE is reset when new data is entered in the register or by the master reset.

b0 RDRF. 1 = received data may be read from the receiver data register. RDRF is reset by a reading of the register, by the master reset or if the DCD input is inactive.

Internal Communication with Keyboard

A keyboard can be connected to a display unit. A keyboard expansion unit and a magnetic identification device are, when used, connected to a display unit via a keyboard unit.

The communication procedure, hardware circuitry and line interface are described in the Keyboard chapter. Some information is also found in the Display Unit and Magnetic Identification Device chapters.

Internal Communication with Selector Pen Device

A selector pen can be connected to a display unit. The signals between them are described in the Selector Pen chapter.

Appendix 1

Alfaskop System 41 Physical Device Addresses

Bits	7	0	0	0	0	0	0	0	0
	6	0	0	0	0	1	1	1	1
	5	0	0	1	1	0	0	1	1
	4	0	1	0	1	0	1	0	1
3 2 1 0	Hex	0	1	2	3	4	5	6	7
0 0 0 0	0	SINGLE DU or DU0	DU4	8	12	16	20	24	28
0 0 0 1	1	SINGLE PU or PU0	PU4						
0 0 1 0	2	SINGLE FD or FD0	FD4						
0 0 1 1	3	–	–						
0 1 0 0	4	DU1	DU5	9	13	17	21	25	29
0 1 0 1	5	PU1	PU5						
0 1 1 0	6	FD1	FD5						
0 1 1 1	7	–	–						
1 0 0 0	8	DU2	6	10	14	18	22	26	30
1 0 0 1	9	PU2							
1 0 1 0	A	FD2							
1 0 1 1	B								
1 1 0 0	C	DU3	7	11	15	19	23	27	31
1 1 0 1	D	PU3							
1 1 1 0	E	FD3							
1 1 1 1	F								

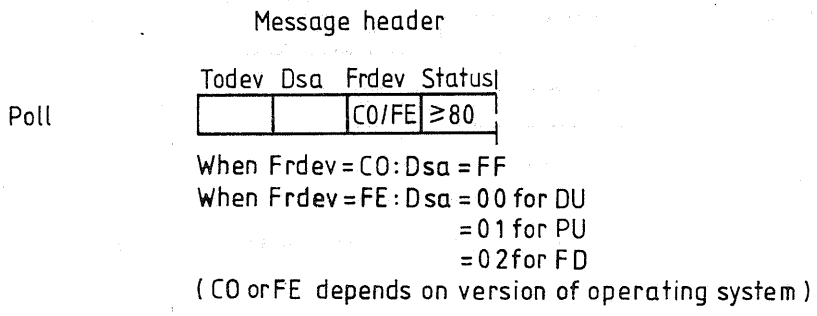
$C0_{(16)}$ = Communication processor address (old versions)
$FE_{(16)}$ = Communication processor address (late versions)
($C0_{(16)}$ or $FE_{(16)}$ is also used by the configuration master functions of a single DU)
$FD_{(16)}$ = System FD address

Appendix 2

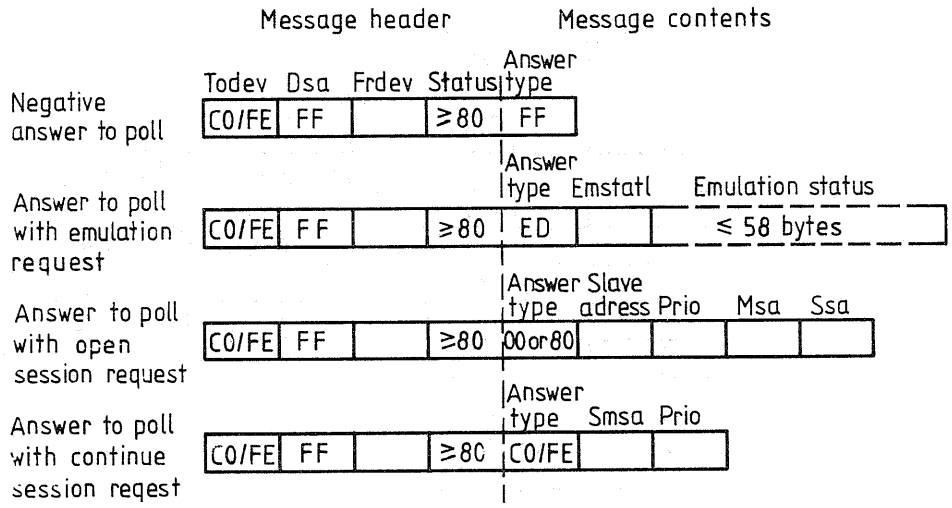
Message Format Summary

All numbers are in hexadecimal. For the interpretation of X (in Msgtyp below) and the message header see Msgtyp/Status layout. See also Explanations below.

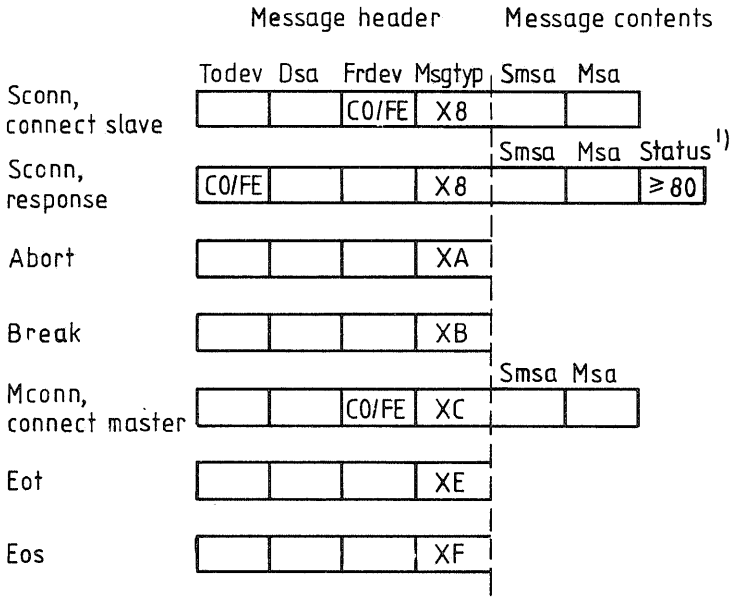
Poll Message



Answer to Poll Messages

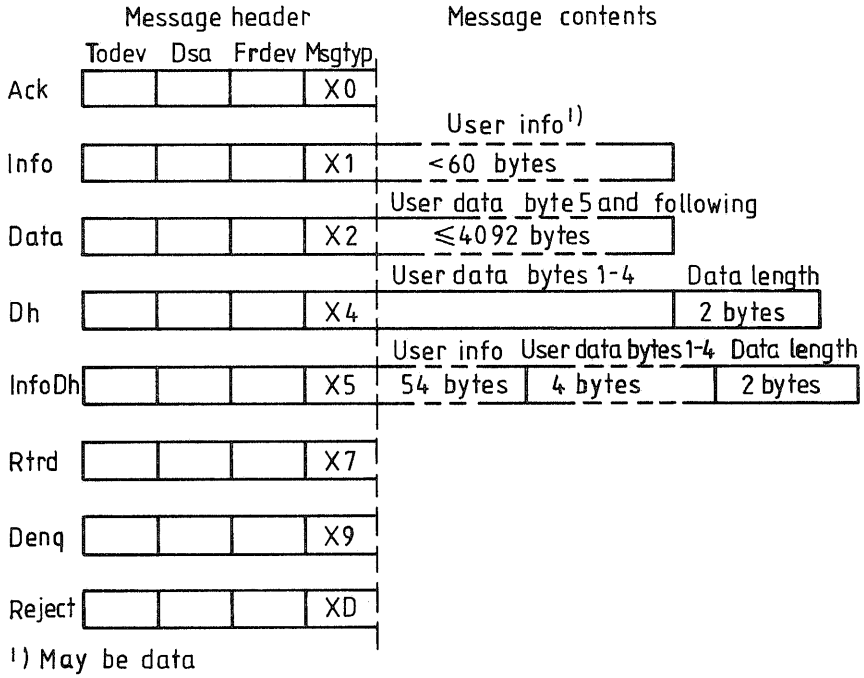


Communication Control Messages



1) Same interpretation as Status in Answer to poll messages

Data Messages



1) May be data

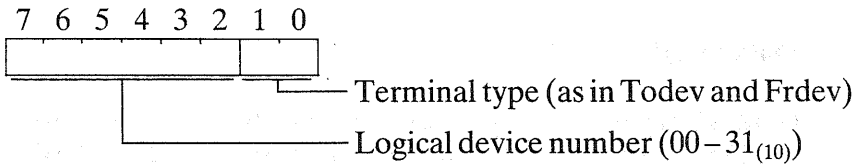
Explanations

Some of the definitions used in the message field above are explained below.

Answer Type and Slave Address

The Answer type has the following meanings.

For a negative answer to poll the Answer type has the value $FF_{(16)}$ and is the only byte in the message contents field. For an answer to poll with emulation request it also indicates the host system number, at present the only value allowed is $ED_{(16)}$. In an answer to poll with open session request the Answer type value may be $00_{(16)}$ or $80_{(16)}$, thereby marking the address type of the following slave address byte. $00_{(16)}$ means physical device address. $80_{(16)}$ means logical device address, where the logical address has a layout as presented below.



In an answer to poll with continue session request the Answer type value is $C0_{(16)}$ or $FE_{(16)}$ (depending on early or late version of operating system (OS).

Emstatl and Emulation Status

The Emstatl states the number of bytes (maximum 20 bytes) in the following emulation status field.

Msa and Ssa

Msa and Ssa stands for master and slave session addresses. They have the same layout as Dsa.

Smsa

Smsa, System master session address, is the session control block number in the configuration master of the control block containing information on each opened session. Smsa is a number between $00_{(16)}$ and $20_{(16)}$. There is one Smsa for each opened session. This number is told to the terminals involved when the CP opens the session and may thus be used by the session master wanting to continue a suspended session.

Prio

A session is always assigned a certain priority. The Prio byte in an answer to a poll message with open or continue session request contains this priority. It decides in which queue in the configuration master the request will be entered. The priorities used are (in falling priority order):

Priority	Meaning	Used in		
0E	Host request or line monitor request or abort handler request	CP		
0C	Stressed device request			FD
0A	Load request	CP	DU	
08	Locate volume request	CP	DU	
06	FD I/O supervisor request	CP	DU	
04	Print request		DU	
02	Utility request		DU	
00	IPL request	CP	DU	FD

Data and Data length

Data is transmitted in two messages. The first four bytes are transmitted in a Dh or an Info Dh message and the rest of the data in a data message. Data length is a two byte field containing the length of the data sent in the following data message.

Communication Processor, Remote

Contents

General

Communication Processor Remote, CPR 4101, forms an interface between a host computer and a cluster of Alfaskop System 41 terminals.

The Microcomputer chapter ought to be well understood before this chapter is studied. A good orientation on the Communication chapter is also recommended; especially the Hardware part under Internal Communication via Two-wire. This chapter will in principle only treat circumstances not discussed in or solved in other ways than said in the two chapters mentioned above.

This chapter is written for CPRs with CPB E34060 0010. Differences in CPB E34060 0000 are, however, pointed out within brackets.

Functions

The CPR is used when a remote connection is wanted (e.g. via modem and telephone network) of a cluster of up to 32 display units (and printer units) or/and flexible units to a host computer. Two CPRs are needed for dual host systems.

The CPR:

- Regularly polls the connected terminals for status and transmission requests.
- Governs transmission to and from the host computer via modem adapters (SCA/SCC), performs editing and code transformation etc.
- Interconnects terminals in the cluster.
- Keeps trace of connected (turned on) units and which volumes (diskettes) that are inserted into the flexible disk units of the system.

Subunits

Basically the CPR can handle eight terminals. It then contains (see Fig. 1):

- CPM, CPR mechanics assembly, i.e. cabinet including front panel.
- FPS or CPS, power supply including universal power board, UPB. See chapter Power Supplies.
- CPB, communication processor board, containing the basic microcomputer with MPU, timing logic, interrupt control logic, address decoding logic, memory access multiplexing logic (including direct memory access logic), 32 + 2 kbytes of memory, crosspoint selection logic, and (two-wire) data link controllers.
- TUA, terminal unit adapter, providing eight two-wire connections. The TUA contains a crosspoint system for connection of the terminals in the cluster, line drivers/receivers, and line transformers for the two-wires. The transfer speed is 300 kbits/s.

- TAB, TUA interconnection board, connecting the CPB and up to five TUA boards. The TAB contains modulators/demodulators, clock generators with phase correction, delay logics and off switches.
- CTF, CP and terminal fan.

As mentioned above five TUA boards can be connected although four are at most used today. There is further place for six optional boards, which can be:

- MRW or MRW-A, memory board, read/write. An MRW or MRW-A can be mounted if additional 16 or 32 kbytes of memory are wanted. See chapter Memory Board R/W.
- MRO, memory board, read only. An MRO can be mounted if additional 24 kbytes of read only memory are wanted. See chapter Memory Board RO.
- SCA, synchronous communication adapter which is used for communication with a host computer via modem. See chapter Synchronous Communication Adapter.
- SCC, synchronous communication controller which is used for communication with a host computer via modem. See chapter Synchronous Communication Controller.

Mechanics

The mechanical build-up of CPR 4101 is shown in Fig. 1. The plastic cabinet is the same as for the CFU and (late versions of) the FD. The main parts of the CPR, from the mechanical aspect, are:

- Box with CPB.
- Rack for TAB, five TUA, and six optional boards.
- Power supply.
- Fan unit.

The TUA boards are connected to the TAB board. The optional boards are stacked on each other and connected to the CPB with a cable.

The following connectors may exist in front of the TUA and the optional boards:

- Eight two-wire or coax connectors for each inserted TUA.
- Connector for V.24/V.28 (25 pins).
- Connector for X21 (15 pins).

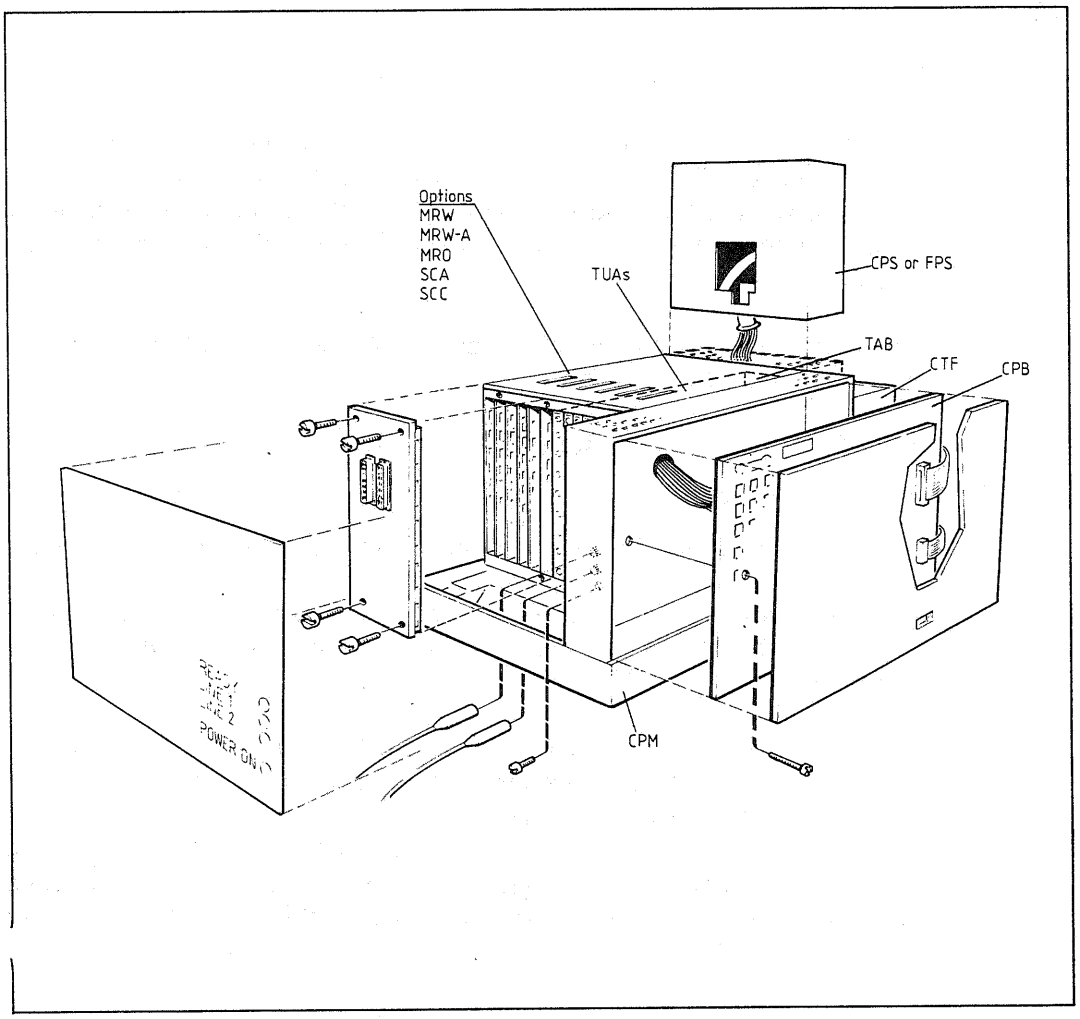


Fig. 1. CPR assembly.

Brief Outline

Monitoring

The CPR front panel shows four lamps. See Fig. 1. The lamps are mounted on the CPB:

- Ready lamp is governed by the CP program. It blinks when the CP has started an internal poll in order to find the system diskette and glows steadily when the CP has loaded the operating system in its read/write memory.
- Line 1 lamp is governed by a line activity signal from the SCA (or SCC). It glows steadily if data is regularly sent to, and blinks if data is only received from the host computer. (For details see chapters on Synchronous Communication Adapter or Synchronous Communication Controller.)
- Line 2 lamp is not used today.
- Power on lamp is directly governed by the +5 V power.

Memory Map

Fig. 2 presents the memory map of the CPR.

The memory areas are described in the Microcomputer chapter, circumstances pertaining to more than 32 kbytes of memory in the Memory Board R/W chapter, and the interrupts under the following heading. Strapping information is found in the Installation and Maintenance Manual.

As can be seen from the memory map the following circuits are addressed (I/O addresses) on the CPB:

- PTM for producing timer interrupts (IRQ 1).
- MIC PIA for interrupt control etc.
- One CS PIA, four ADLCs, and one DMAC for the two-wire interface.

Circuits on optional SCA or SCC boards can also be addressed. Communication with a host computer may proceed via circuits on an SCA at addresses $F768_{(16)} - F77F_{(16)}$.

If an SCC is used, the microcomputer of the CPB reads data from and writes data to the host computer in a dual access area on the SCC. The addresses $6000_{(16)} - 7FFF_{(16)}$ are then reserved for this purpose and the memory on the CPB must in this case be limited to 24 kbytes by strapping. The SCC handles certain message editing and the actual transfers between the dual access area and an interface circuit (SSDA or ADLC) on the SCC. For details see chapter Synchronous Communication Controller.

Interrupts

The principles of the interrupt system are described in the Microcomputer chapter. The locations of the interrupt vectors (i.e. the addresses to the interrupt routines) are found in the memory map.

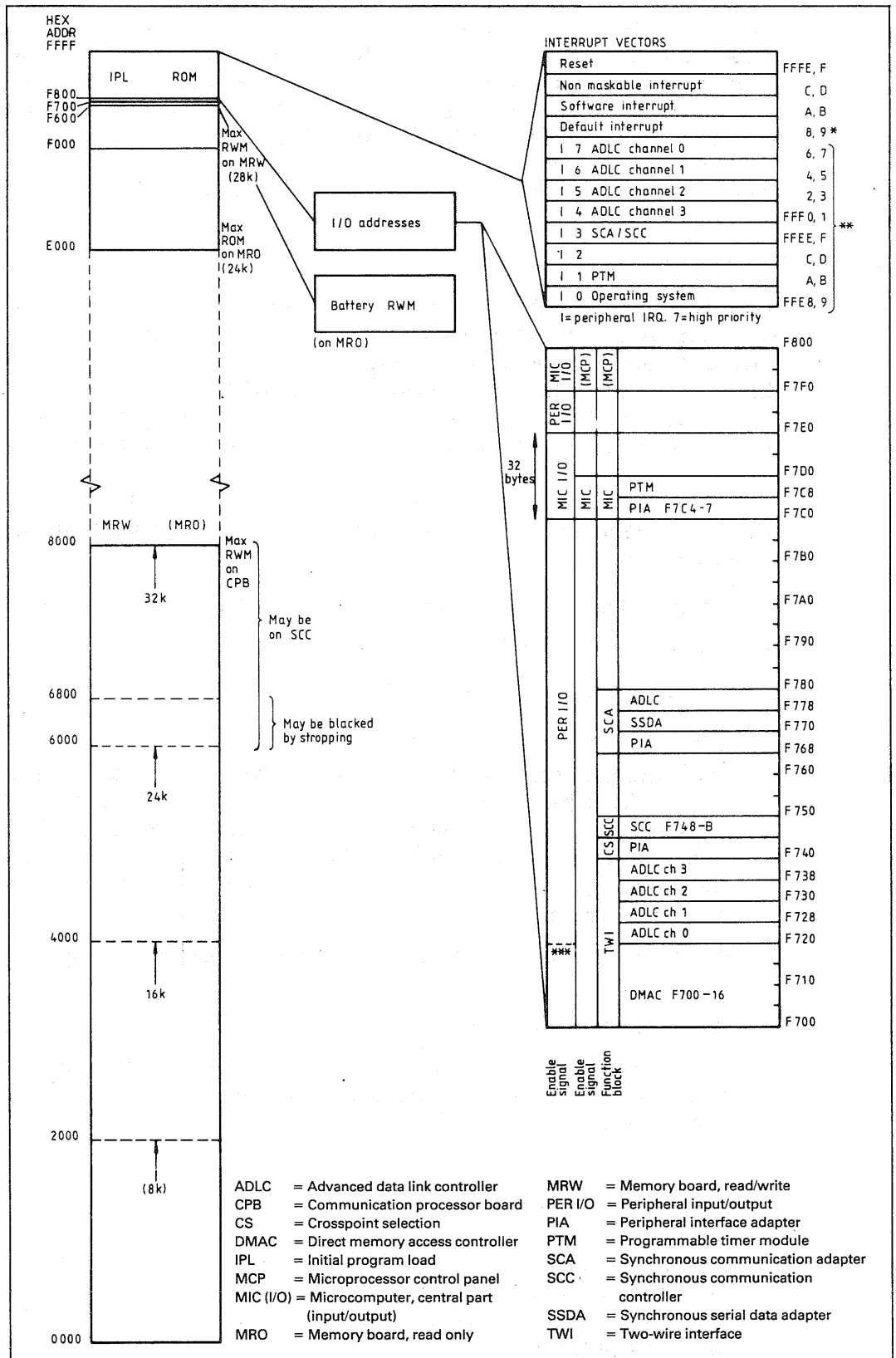
Here follows a description of the origin, properties and use of the different CPR interrupt signals.

Interrupts out

Two interrupt signals can be generated and fed to subunits. Interrupt slave I is generated when b5 is set in the peripheral register A of the MIC PIA. Interrupt slave II is generated when b3 is set in the same register.

Reset, NMI, and SWI

The MPU is reset at power on or at depression of the Reset button. The MPU will after a reset fetch the contents of the cells $FFFE_{(16)}$ and $FFFF_{(16)}$ to the program counter and execute the reset program (see also Reset functions).(On CPB E34060 0000 a depression of the Reset button will cause an NMI instead of a reset.)



* MPU reading FFF8 – FFF9 will transfer the contents of two cells within FFE8 – FFF9, depending on highest priority valid interrupt, to the MPU.

** A (read or preferably) write operation into one of the cells FFE8 – FFF7 will result in a setting of a corresponding mask value. As an example, FFEE or FFEF **from the MPU** will disable interrupts of priority < 3.

*** Not present during the three DMA cycles DMA T1 – T3.

Fig. 2. CPR memory map.

NMI, non-maskable interrupt may be generated from the test unit MCP (Microprocessor control panel) or when a parity error is detected at readouts from the read/write memory. The program can test the source of the interrupt by reading the peripheral register A of the MIC PIA and test bits 0 and 2 (and 7 for CPB E34060 0000) (see MIC PIA functions).

SWI, software interrupt is generated by an MPU instruction.

Interrupt 7 – 0 and Default Interrupt

Interrupts 7 – 0 (I7 – 0) are set in the interrupt register. The interrupts are arranged in a priority order where I7 has the highest priority. A set interrupt must be of the same as or higher priority level than denoted by the present value of the mask register in order to be valid. The interrupt vector is fetched from the cells $FFF8_{(16)}$ and $FFF9_{(16)}$ (default interrupt) if a triggering interrupt is not valid at the time of the vector fetch.

The maskable interrupts, I7 – 0, are further only serviced if the I-bit of the MPU condition code register is reset. I7 – 0 originate from the following sources:

I7 originates from the ADLC (advanced data link controller) of channel 0.

I6 originates from the ADLC of channel 1.

I5 originates from the ADLC of channel 2.

I4 originates from the ADLC of channel 3.

I3 may originate from a non-steady source (short pulse interrupt). It will be latched by the interrupt logic if bit 4 of the MIC PIA peripheral register A is set. Any latched interrupt is reset and the state of the I3 line to the interrupt register will be equal to the state of the incoming interrupt line if bit 4 is reset. The I3 line is used for instance by an SCA or an SCC.

I2 functions as I3 but is controlled by bit 6 of the MIC PIA peripheral register A.

I1 originates from the PTM.

I0 originates from the CA2 output of the MIC PIA. The program can thus make an interrupt by writing $110_{(2)}$ in bits 5 – 3 of the MIC PIA control register A.

Programmable timer module, PTM

A programmable timer module, PTM, is used to provide two or three time bases for generation of timeouts and a real time clock (calendar clock) in the CPR. A detailed description of the PTM circuit is found in the Microcomputer chapter.

The PTM contains three timers with 16-bit counters. Timer 1 is typically programmed to divide the basic clock (Ø2) by 21 300 and thus make an interrupt (I1) every 20 ms (50 Hz). The output from timer 1 is fed to the clock input of timer 2 and there typically divided by 50. Timer 2 will thus make an interrupt (I1) every second. Timer 3 is a spare timer, which for instance may be used for short time measurements.

Terminal Communication

General

The CPR communicates with the terminals in the cluster via four ADLC (advanced data link controller) circuits. For a description of the used communication procedure and a survey of the hardware functions please refer to the Communication chapter.

Each one of the ADLCs forms a part of one of four channels. Each channel can by a crosspoint system on each used TUA board be connected to eight two-wires. See Fig. 3. Each two-wire may be chained to a display unit (and a printer unit) or/and a flexible disk unit. Note that the demands on maximum response times and number of terminal addresses in different applications introduce restrictions in the number of connected terminals.

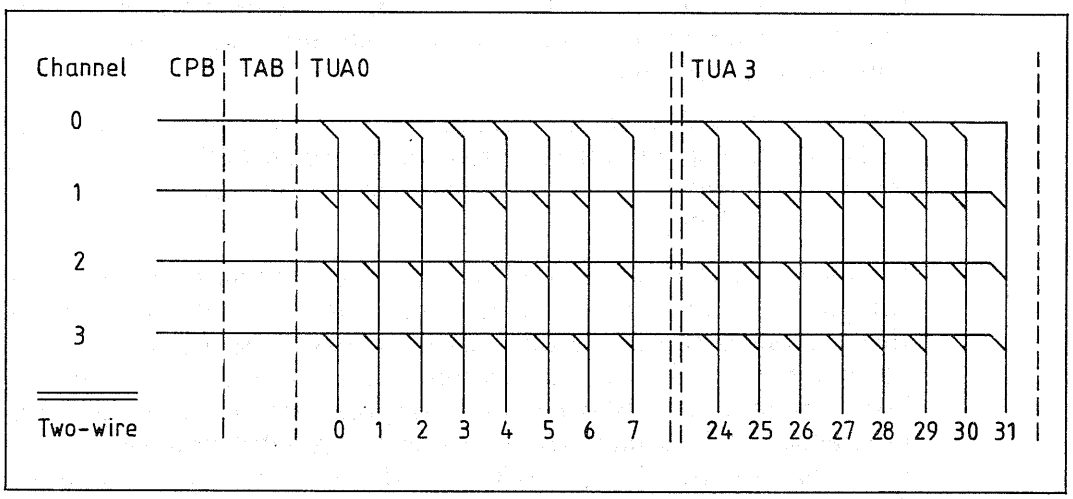


Fig. 3. Crosspoint system principles.

A certain channel may be connected to any one and any number of two-wires in the crosspoint system. Transfers may also take place on more than one channel simultaneously. Note, however, that a certain two-wire should not be connected to more than one channel at the same time.

The ADLCs are controlled by the MPU or by the direct memory access logic, including a DMAC (direct memory access controller) circuit. The crosspoint system is controlled by a crosspoint selection PIA (peripheral interface adapter) and a crosspoint selection logic. See block diagram.

Direct Memory Access Functions

The DMAC (see Microcomputer chapter) control of the four channels, means that the DMAC registers associated with a certain channel must be used both for read memory (transmit data) and write memory (receive data) access sequences. As suggested above, more than one channel may be enabled simultaneously. The direct memory access timing (including DMAC), ensures that only one channel transfers a byte at a certain time.

An ADLC may thus be accessed in two ways (see block diagram):

- Addressed by the MPU; CE ADLC provided by the two-wire interface address decoder, R/W and address lines for register selection provided by the MPU. All ADLC registers can be accessed.
- Addressed by the DMAC during DMA T2 (DMA mode); CE ADLC provided as DMAC encoded channel address (TXAKA and TXAKB) directed DMA mode signal to the proper ADLC, R/W and one address line provided by the DMAC. Receiver and transmitter FIFOs can only be accessed as the other address line (A1) is tied high.

Channel Usage

The four channels are used in the following way:

- Channel 0 is used for internal polling, i.e. to investigate if the connected terminals want to communicate with the host computer or transfer data to or from another terminal in the cluster. (Note that FDs are considered as terminals.)
- Channel 1 is used for connections between the CPR and the terminals in the cluster, e.g. for initial program loading of the CPR or connecting sequences. It can also be used for terminal to terminal communication (see Channel 3 below).
- Channel 2 is used for traffic between the terminals and the host computer. As an example, say that the CPR while polling has found that a number of display units want to send messages to the host computer. When the CPR then is polled (or a terminal selected) by the host computer the CPR connects the display units, one after the other on channel 2. The actual message is then via the CPR stored in the line buffer and by the SCA (or SCC) transferred to the host computer. The SCA handler program takes care of the data in the line buffer, makes necessary code transformations, and sends the data.
- Channel 3 is used for internal traffic (terminal to terminal communication). A display unit on one two-wire may e.g. answer to a poll (on channel 0) that it wants to make a local printout on a printer unit connected to a display unit on another two-wire. In case this printer unit is not busy and no other units are queuing to use the printer unit, the CPR connects the requesting display unit with the printer unit on channel 3. Note that the ADLC 3 (in the CPR) will still be listening to the traffic. Thus, the CPR program may know when the transfer is completed.

Turning Crosspoints On and Off

Each crosspoint consists of two SCRs (silicon controlled rectifiers or thyristors). To connect one two-wire to one channel a selection address and an On order are written in the crosspoint selection PIA. Thereby one TUA enable signal, one two-wire enable signal (Tw en) and one Channel select (Ch sel) signal are activated, making the selected pair of SCRs conducting.

The selection signals disappear when the On order is cancelled via the PIA. The SCRs require hold currents in order to stay on. These hold currents are provided as long as the Channel off (Ch off) signal is not

activated. The Channel off signal will, when activated, turn off all crosspoints connected to one (selected) channel.

Thus:

- When a selection word and an On order are written into the crosspoint selection PIA, only one pair of SCRs (one crosspoint) is activated. To connect more two-wires to the channel, one selection must be made for each crosspoint.
- A certain crosspoint may not be disconnected separately – all crosspoints of one channel will be disconnected at the same time.

Crosspoint Signals

There are two physical connections (for each channel) through the crosspoint switch, see block diagram. One is used for:

- Modulated received data (transferred as current) and Request to send out (RTS out, transferred as voltage).

The other is used for:

- Modulated transmitted data (transferred as voltage).

Modulated transmitted data is provided by the modulator/demodulator of the channel in question and may originate from:

- The ADLC (i.e. from the CPR memory).
- A terminal connected to the channel, that sends data to the modulator/demodulator on the Modulated received data line (transferred on the same line as Request to send out through the crosspoint switch).

Request to send out is generated in the modem when:

- The ADLC signals Request to send, set by the CPR program before the CPR starts transmitting to the terminal(s) connected to the channel, or:
- The modulator/demodulator receives valid Modulated received data.

Note that the modulator/demodulator generates Request to send out and Modulated transmitted data not only when the CPR is transmitting but also when some terminal is sending data into the crosspoint system. The two-wire line driver (on the TUA) connected to the terminal sending into the CPR will be disabled, i.e. not affected by the Request to send out signal occuring on the channel – the line drivers of all other two-wires connected to the channel will, however, be in the transmitting state.

This enables (in addition to ordinary transfers):

- Transfers of messages between terminals in the cluster.
- Connections of display units listening to the traffic on the channel.
- The CPR to keep track of terminal to terminal communications.

Example of Data Transfers

There are thus three communication modes as shown in Fig. 4.

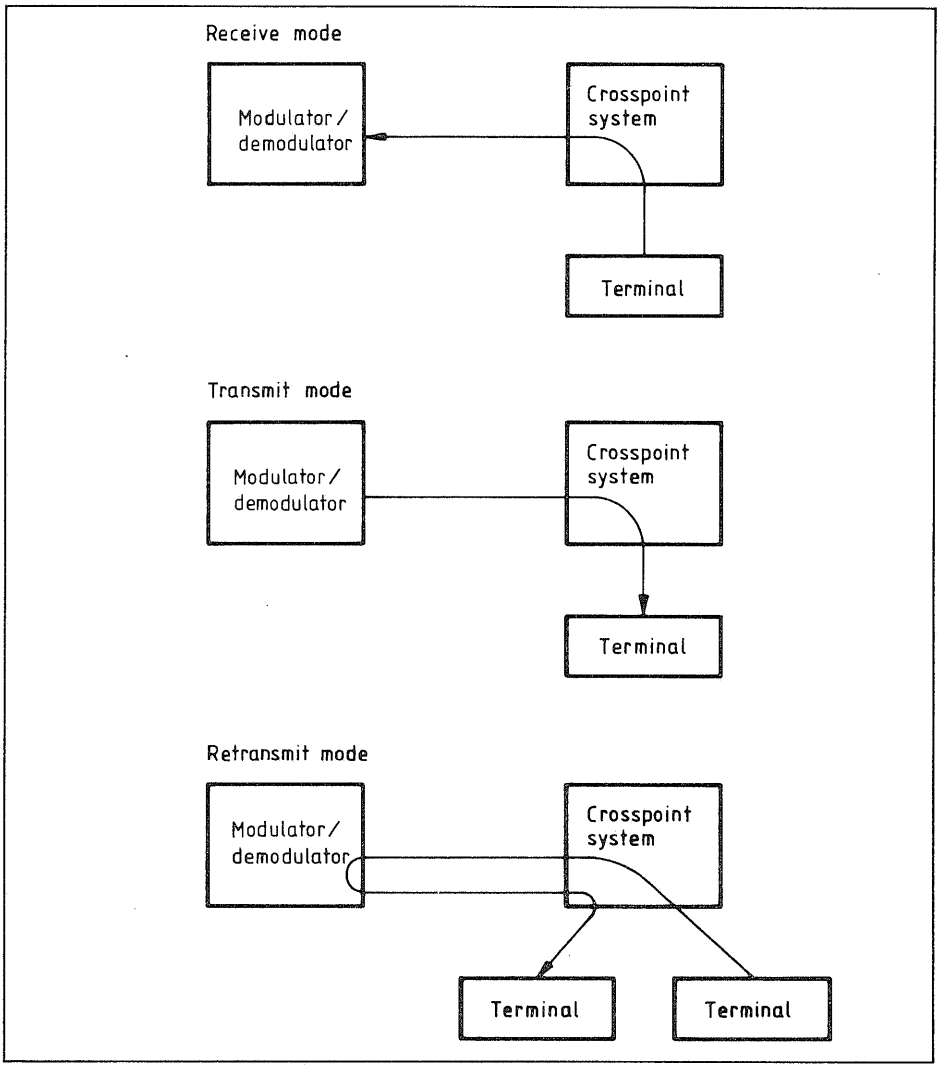


Fig. 4. Communication modes.

Fig. 5, an excerpt from the block diagram, exemplifies a data transfer situation in the crosspoint system:

- One byte is transferred from the memory of the CPR to the ADLC of channel 0 under the control of the DMAC circuit.
- A previously transferred byte of a poll message is shifted out from the ADLC of channel 0, modulated, and transmitted to a terminal connected to two-wire number 7.
- Data is transferred from a terminal connected to two-wire 0 to another terminal on two-wire 3. Note that:
  - Channel 1 is used.
  - Data proceeds via the modulator/demodulator.
  - Data is accessible for the microcomputer of the CPR (in ADLC 1).

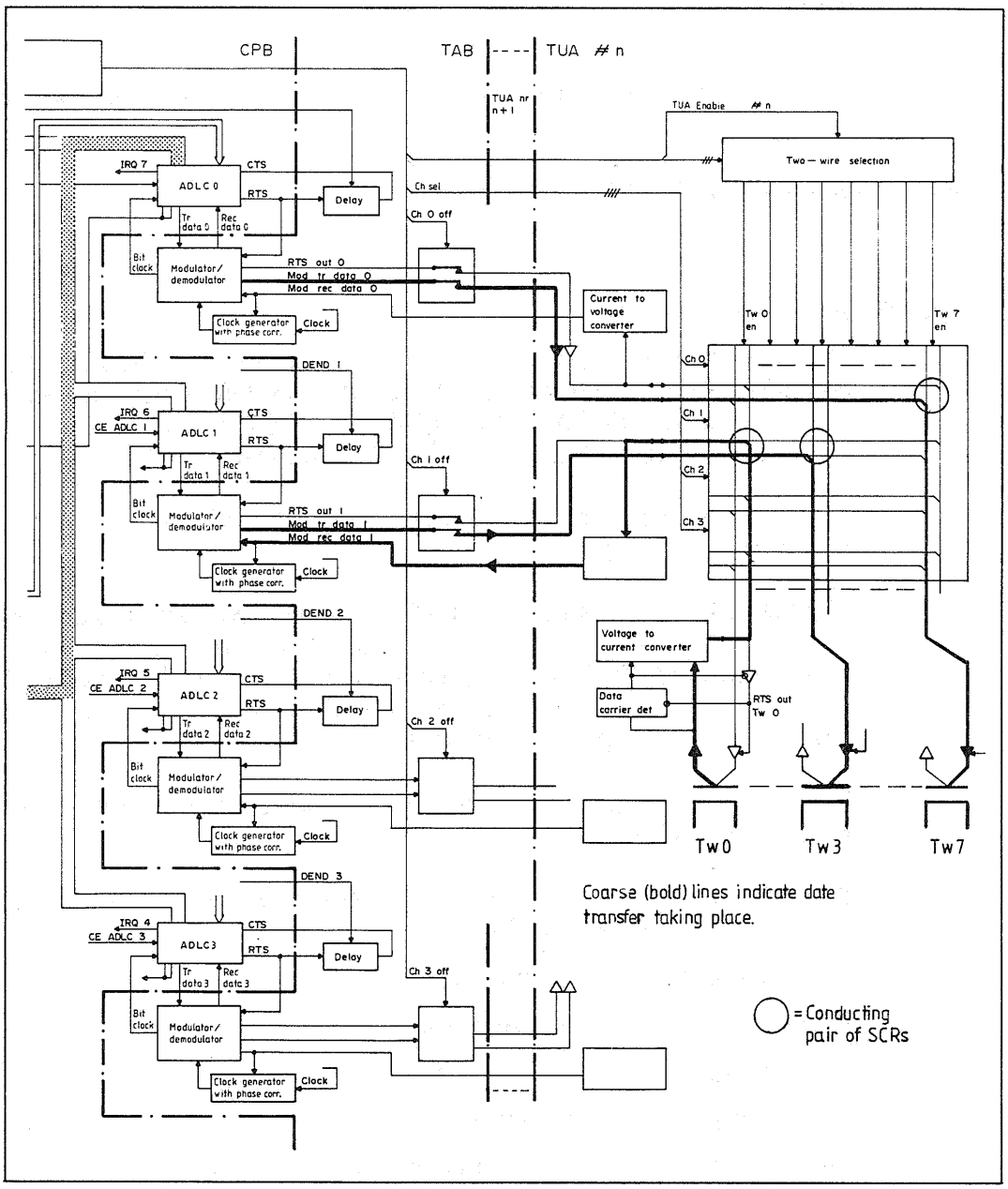


Fig. 5. Cluster data transfer example.

Detailed Description

This section will only cover more complicated circuits or special solutions.

Microcomputer PIA, MIC PIA, Functions

Input/Output Functions

The MIC PIA controls several central functions. See chapter Microcomputer; Peripheral Interface Adapter, PIA for definitions and a general survey. However, IRQA, CA1, IRQB, CB2, PB7, 5, and 4 are not used. (Neither CB1, PB6, 3, 2, and 1 for CPB E34060 0000.)

The CA2 output is used to generate interrupt 0 if CRA b3 is zero-set (CRA b5, 4 = $11_{(2)}$).

The MIC PIA interrupt flag (CRB b7) is set if the Reset button is depressed (as CB1 is then taken low). This must be checked by the program as IRQB is not connected. (This paragraph is not valid for CPB E34060 0000. The state of the Reset button is here fed to PA7 and can thus be checked by a MPU read of peripheral register A.)

The PA outputs/inputs are used in the following way (opposite polarity means "do not" or "not"):

PA7	Input	Hard-wired low. (1 = Reset button depressed in CPB E34060 0000.)
PA6	Output	1 = latch interrupt 2
PA5	Output	1 = interrupt slave I
PA4	Output	1 = latch interrupt 3
PA3	Output	1 = interrupt slave II
PA2	Input	0 = MCP connected and in test mode
PA1	Output	0 = lit Ready LED
PA0	Input	1 = parity error FF is set

The PB outputs/inputs are used in the following way:

PB6	Output	1 = activate (general) Reset. The output must be set low after power on. (The MPU and the MIC PIA are not reset by [general] Reset.) (This paragraph is not valid for CPB E34060 0000.)
PB0	Output	1 = reset and keep parity error FF reset.

PB3-1 are not used today. PB3 ought to and PB2-1 must, however, be low. (The last sentence is not valid for CPB E34060 0000.)

Reset Functions

The MPU is reset by a power on reset or a Reset button depression. The MIC PIA is only reset by the power on reset. The (general) Reset is activated at power on and stays active until MIC PIA PB6 is set low by the MPU.

The (general) Reset will not be activated at a Reset button depression. The MPU, however, will check the MIC PIA interrupt flag (CRB b7) and find that the button has been depressed. The MPU can then make a general reset by setting MIC PIA PB6 first high and then low.

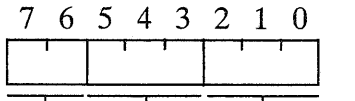
(The reset function differs in CPB E34060 0000. The [general] Reset will be activated at power on. A depression of the Reset button will lead to a non-maskable interrupt, NMI, which must be handled by the program as the MPU cannot generate a hardware [general] Reset.)

Crosspoint Selection Control

Crosspoint Control Signals

The control signals to the crosspoint system are generated in a crosspoint selection PIA (peripheral interface adapter, see Microcomputer chapter for a general description) and distributed via a decoder and a selector on the CPB (crosspoint selection logic on the block diagram) and a decoder on each TUA board.

The PIA generates an On signal from CB2 and a Channel off (Ch off) signal from CA2. All of peripheral register B is used as output with the following function:



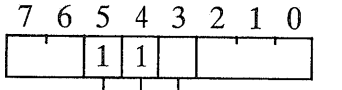
Two-wire address (Tw addr) decoded to one of eight two-wire enable (Tw en) signals when applicable TUA enable is present. $000_{(2)}$ = Tw en 0. $111_{(2)}$ = Tw en 7.

TUA select. When On is active these bits are decoded to one of five TUA enable signals. TUA enable 0 (TUA select = $000_{(2)}$) enables the decoding of Tw en 7 – 0 on the TUA 0, TUA enable 1 (TUA select = $001_{(2)}$) enables the decoding of Tw en 7 – 0 on the TUA 1, etc.

Channel select ($00_{(2)}$ = channel 0, $11_{(2)}$ = channel 3). These bits are used to control the distribution of On (called Ch sel 3 – 0 when fed to the TUAs and En [able] curr [ent] on the TAB and TUA logic diagrams) and Ch off to applicable channel.

Crosspoint Selection PIA Control

In order to get the above mentioned functions the control register A (CRA) of the crosspoint selection PIA must be set as follows:

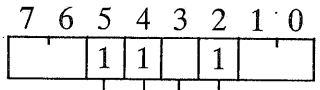


CA2 output (Ch off) follows what is written into this bit (if bits 5 – 4 are ones). 0 = selected channel is turned off. 1 = connections may be made on all channels.

CA2 follows CRA b3.

CA2 functions as output.

Control register B (CRB) should contain:



Peripheral register B accessible. This bit must of course be 0 at the initiation when the Data direction register B is set to all ones.

CB2 output (On) follows what is written in this bit (if bits 5 – 4 are ones).

CB2 follows CRB b3.

CB2 functions as output.

TAB Functions

The TAB contains for each one of the four channels (see Fig. 6) a clock generator with phase correction, a modulator/demodulator, a delay circuit, and an off switch. The off switches are treated under TUA Functions as they functionally belongs there.

The Hardware section under Internal Communication via Two-wire in the Communication chapter is preferably read before the following pages.

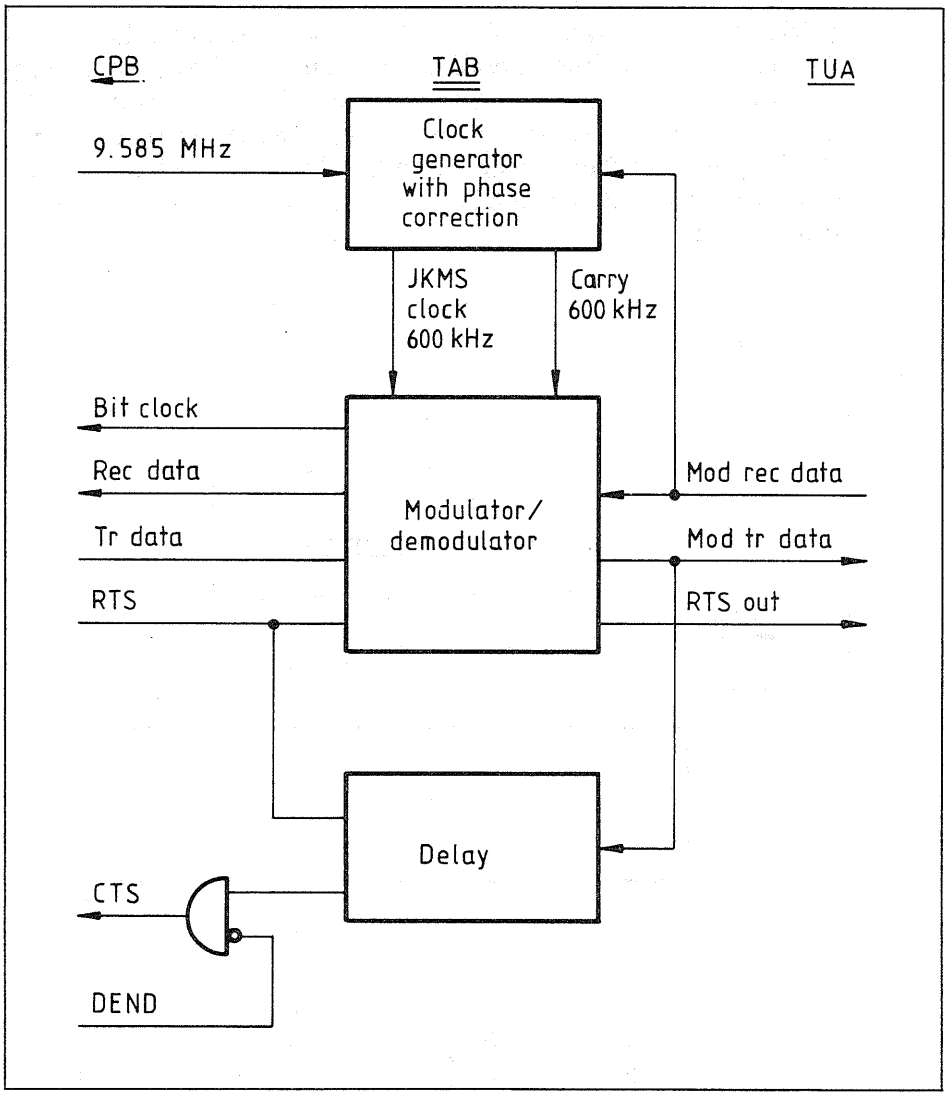


Fig. 6. ADLC/Crosspoint system interface.

Transmitting

Data to be transmitted (Tr data, from the ADLC) is presented in serial, not modulated, format to the modulator/demodulator. The modulator/demodulator converts the signal to a frequency shifted signal, Modulated transmitted data (Mod tr data), which then eventually may be fed to the two-wire(s).

Transmitted data is, when Request to send (RTS) is active, modulated in two steps. See Figs 7 and 8. A Bit clock (300 kHz) and a Transmitted data condition (Tr data cond) signal is generated in a ROM (and a D flip-flop as far as the Bit clock is concerned). The Bit clock causes the ADLC to send new data bits on the Transmitted data line. Transmitted data condition is either a high level (corresponding to Transmitted data low) or a copy of the Bit clock (corresponding to Transmitted data high). Transmitted data condition is then fed to a JKMS flip-flop which generates the frequency shifted signal, Modulated transmitted data. Note that the clock generator always feeds Carry and JKMS clock, thus even during transmission.

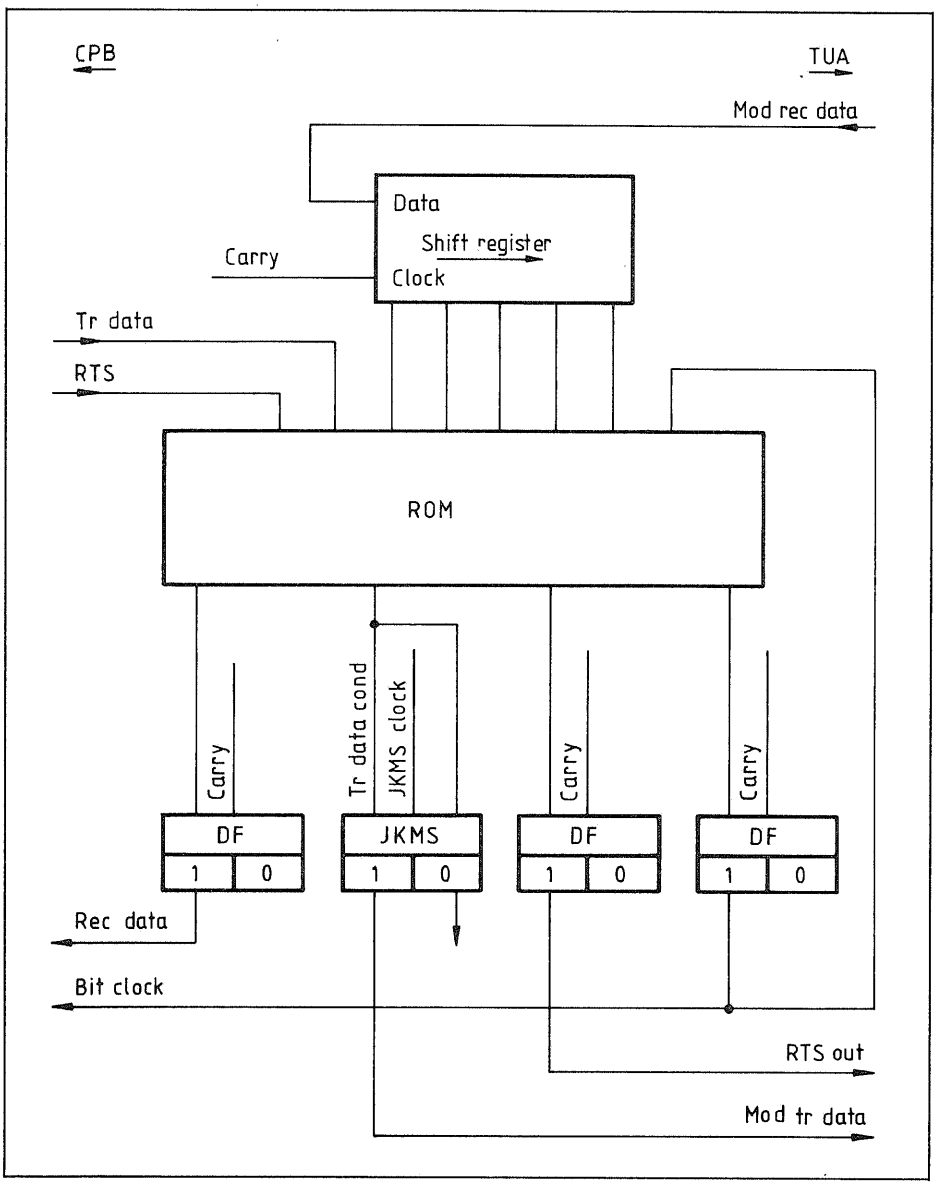


Fig. 7. Modulator/demodulator.

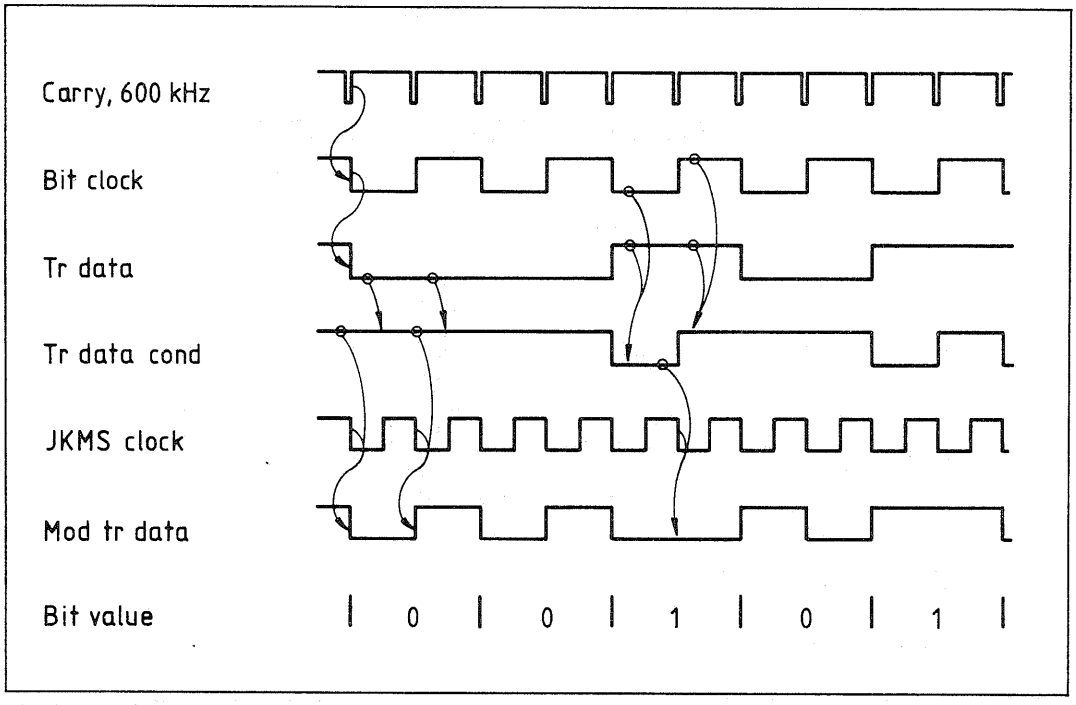


Fig. 8. Modulation timing diagram.

The Request to send output (TUA out) follows the Request to send input (RTS) in the transmit mode and is a function of Modulated received data in the retransmit mode.

Clear to send (CTS, see Fig. 6) follows Request to send input (RTS) with a delay of 32 periods of ones. This delay will enable the modem in the receiving unit (e.g. in a DU) to synchronize before the transmission is started. A frequency corresponding to the state of the Transmitted data input will be sent on Modulated transmitted data during the delay. An active DEND signal (at the end of a transmission sequence) from the DMAC (see Microcomputer chapter) will deactivate Clear to send.

Receiving

The Modulated received data (Mod rec data) from a two-wire is fed to the modulator/demodulator and the clock generator with phase correction.

The clock generator with phase correction consists of a counter which generates a 600 kHz Carry signal and a 600 kHz JKMS clock by dividing a 9.585 MHz clock by 16 when a phase correction logic is inactive. The phase correction logic is activated by positive transitions on the Modulated received data line and will then make the Carry and JKMS clock coincident to the center of the shortest received pulses. As can be seen from Fig. 9 the phase correction logic may compress or stretch the nominal 600 kHz period by 1/16. When a positive transition occurs during the first half of a JKMS clock period, the Carry and JKMS clock are accelerated, when the transition comes during JKMS clock high time the Carry and JKMS clock are slowed down.

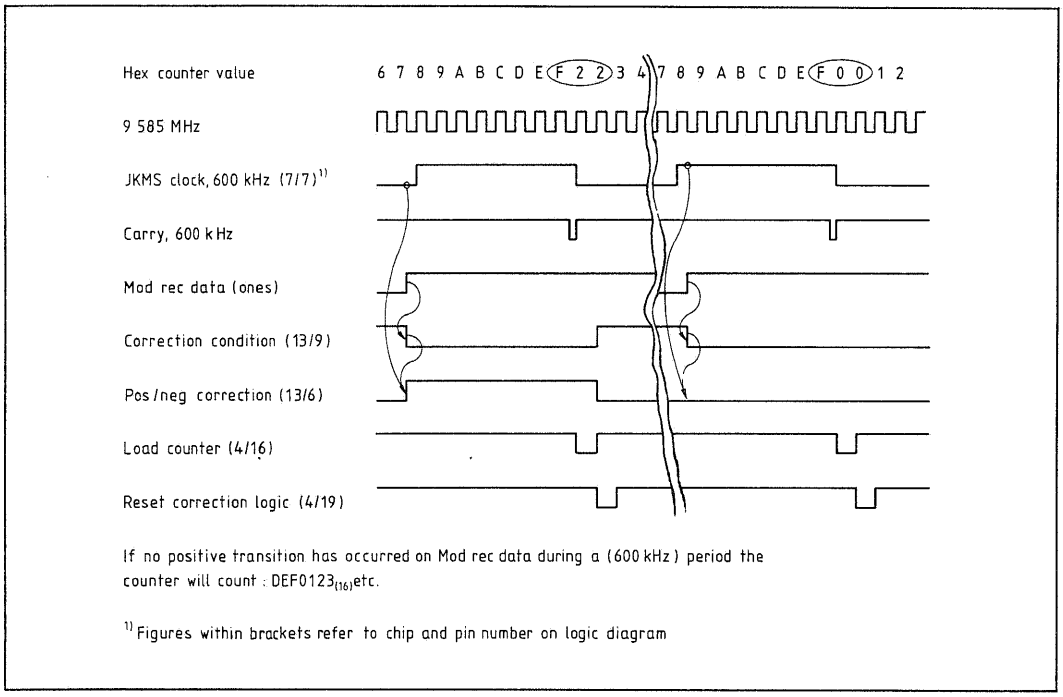
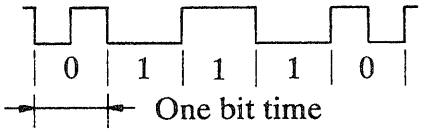


Fig. 9. Phase correction timing diagram.

The modulator/demodulator (see Fig. 7) works as follows. The Carry signal (600 kHz), mentioned above, clocks Modulated received data into a 5-bit shift register, which thus will contain the latest received five bit halves as the bit rate is 300 kHz. The outputs of the shift register are as well as the generated Bit clock (Transmitted data, and Request to send) fed to the address inputs of a ROM. The functions of the ROM in conjunction with one JKMS and three D flip-flops are to:

- Identify the bit limits and validate the sequence of ones and zeroes on Modulated received data, i.e. each bit must consist of an LH, HL, LL or HH and there must be a shift of polarity between the bits in order to be accepted:



- Generate a demodulated form of valid Modulated received data, i.e. Received data (Rec data, for the ADLC). Received data is always high when Modulated received data is invalid.
- Generate a 300 kHz Bit clock (for the ADLC) with a positive transition in the middle of each bit time (which may require a 180° change of phase).
- Remodulate valid Received data to Modulated transmitted data and generate RTS out. This function is only used in retransmit mode. The remodulation is done analogous to the modulation in transmit mode.

It will take one bit time from that a bit is clocked into the shift register until it is clocked out of the Received data flip-flop.

TUA Functions

Fig. 10 shows the circuits pertaining to one channel and one two-wire on a TUA board and the off switches on the TAB.

The connection through the crosspoint switch will be established when the Two-wire enable (Tw en) and Channel select (Ch sel) signals are activated, i.e. when the encircled double collector transistor in Fig. 10 is turned on (if Channel off [Ch off] is inactive). The SCRs will then be kept on by the holding current between the ~ +10 V to RC1 and RC2 in the current to voltage converter and the −12 V of the current sources in the voltage to current converter. The SCRs are turned off when the bases of the transistors T1 and T2 in the current to voltage converter are sunk to −12 V by an active Channel off signal. (No one of Modulated transmitted data, Request to send out, and Modulated received data can then be transferred through the crosspoint system.)

Data from the two-wire line is via the line transformers and line receivers fed to the receive gate and the data carrier detector. The detector will, if it accepts the signal, allow the signal to pass the receive gate and close te RTS gate which will disable the line driver and thus inhibit transmitting to this two-wire. The receive gate output affects, by the connection to the emitter of T3, the current through T3 (but not the voltage at the collector of T3), SCR1, T1 and RC1 (if the SCR1 is on). This creates a signal voltage across RC1 which is converted to a TTL level by a comparator and fed to the TAB as Modulated received data.

At sending data on the two-wire line, Request to send out and Modulated transmitted data control the voltages at the bases and thus also at the emitters of T1 and T2. These voltage differences are transferred through the crosspoint switch (if the SCRs are on). An active Request to send out enables the two-wire line driver, if the RTS gate is not closed by the data carrier detector. Modulated transmitted data is by the line driver fed to the line transformers and the two-wire line. (The currents through the SCR1, SCR2, T1, and T2 are not affected by the different voltage outputs from T1 and T2.)

The modulator/demodulator will always refresh and retransmit valid Received data to the crosspoint switch on the Modulated transmitted data line. Request to send out will also be active and the two signals are sent to all connected line interfaces (on the same channel), even the line interface which receives the original signal from the line, but the line driver in this one is disabled by the incoming signal from the two-wire. Note that at least one more than the receiving crosspoint must be connected in order to obtain an actual retransmission. Note also that Modulated transmitted data and Request to send out are transferred through the crosspoint system at the same time as Modulated received data is transferred through it in the opposite direction.

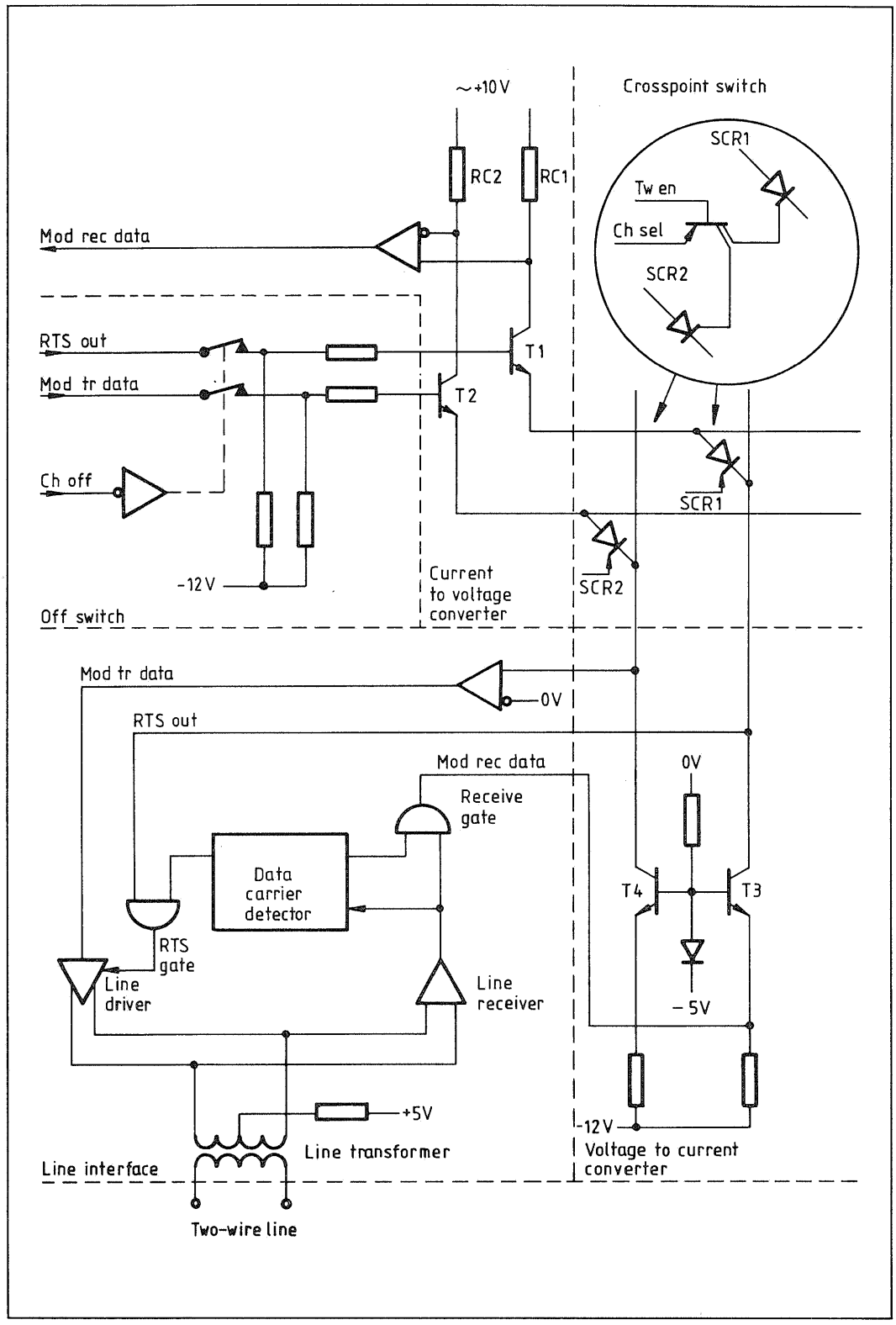
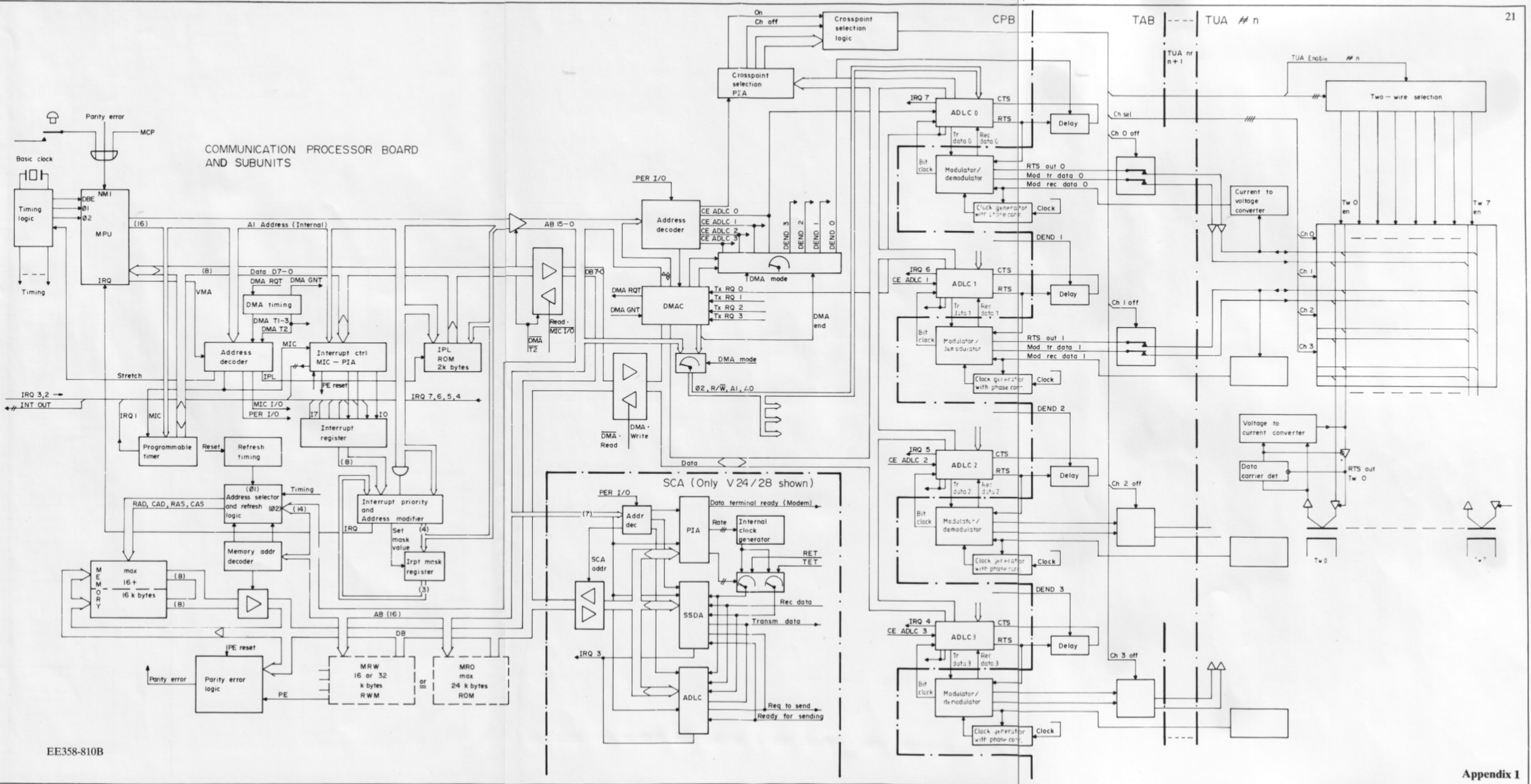


Fig. 10. Crosspoint system and two-wire line interface. (One channel, one two-wire.)



EE358-810B

Appendix 1

Communication Processor, Local

Contents

General

Communication Processor, Local, CPL 4102, is used for high speed communication between a host computer and a locally connected channel attached cluster of Alfaskop System 41 terminals.

A description of the components that are common to the two types of communication processors of Alfaskop System 41 is found in the chapters Microcomputer and Cluster terminal system, in this manual. The following components are special to CPL 4102:

CLM	CPL mechanics (see Mechanics below)
CCC 1, CCC 2	Channel communication controller boards. CCC 1 contains a high-speed microprocessor with associated 2k × 64-bit program storage in ROM. CCC 2 contains channel interface circuits, address decoding, and a line buffer for intermediate storage of communication data to/from the host computer.
CIB	CCC Interconnection board

This chapter contains:

- A description of the interface signals and sequences as specified by IBM
- A brief outline providing basic knowledge of the operation of CCC circuits and program at the block flow chart level
- Detailed description of important functions and circuits, and a detailed example of the microprogram execution.

Mechanics

CPL 4102 consists of the same basic assembly as CPR 4101. See Fig. 1. The difference is that CPL is equipped with CCC 1, CCC 2, and CIB instead of SCA/SCC. Another difference is the size of the cabinet. To simplify accomodation of the heavy cables to the computer, CPL is designed for floor mounting.

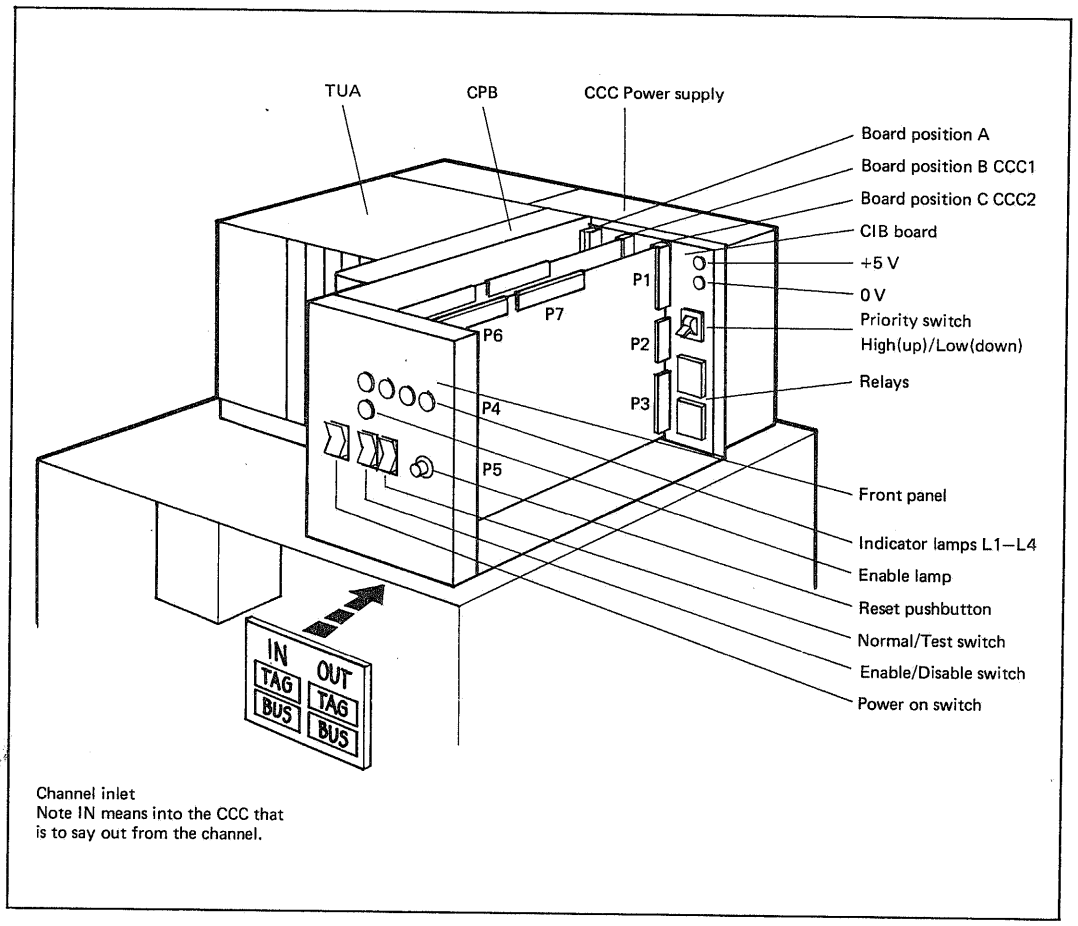


Fig. 1. CPL 4102 – mechanical design

Principles of Local Connection

General

The purpose of the channel communication controller, CCC, is to control the exchange of signals between the IBM I/O interface and the CPB interface.

This section outlines the main principles of interface operations between the CCC and the IBM channel. Detailed information about this interface can be found in the offical IBM document No GA 22 – 6974 – 0 entitled "IBM System /360 and System /370 I/O Interface Channel to Control Unit Original Equipment Manufacturers Information" The Alfaskop System 41 Reference manual (document No FE411-810) contains also detailed information about commands, orders and message layout.

The IBM computer has several channels for communication with I/O control units. Up to eight control units can be connected to one channel. One example showing CPL 4102 connected as a control unit is shown in Fig. 2. Normally, several other types of control units are connected to the same channel. The channels can be of three types: selector, byte-multiplexer and block-multiplexer. A byte-multiplexer channel can operate either in byte-multiplex mode or in burst mode. A selector channel operates only in burst mode. A block-multiplexer channel operates in burst mode but allows multiplexing between blocks.

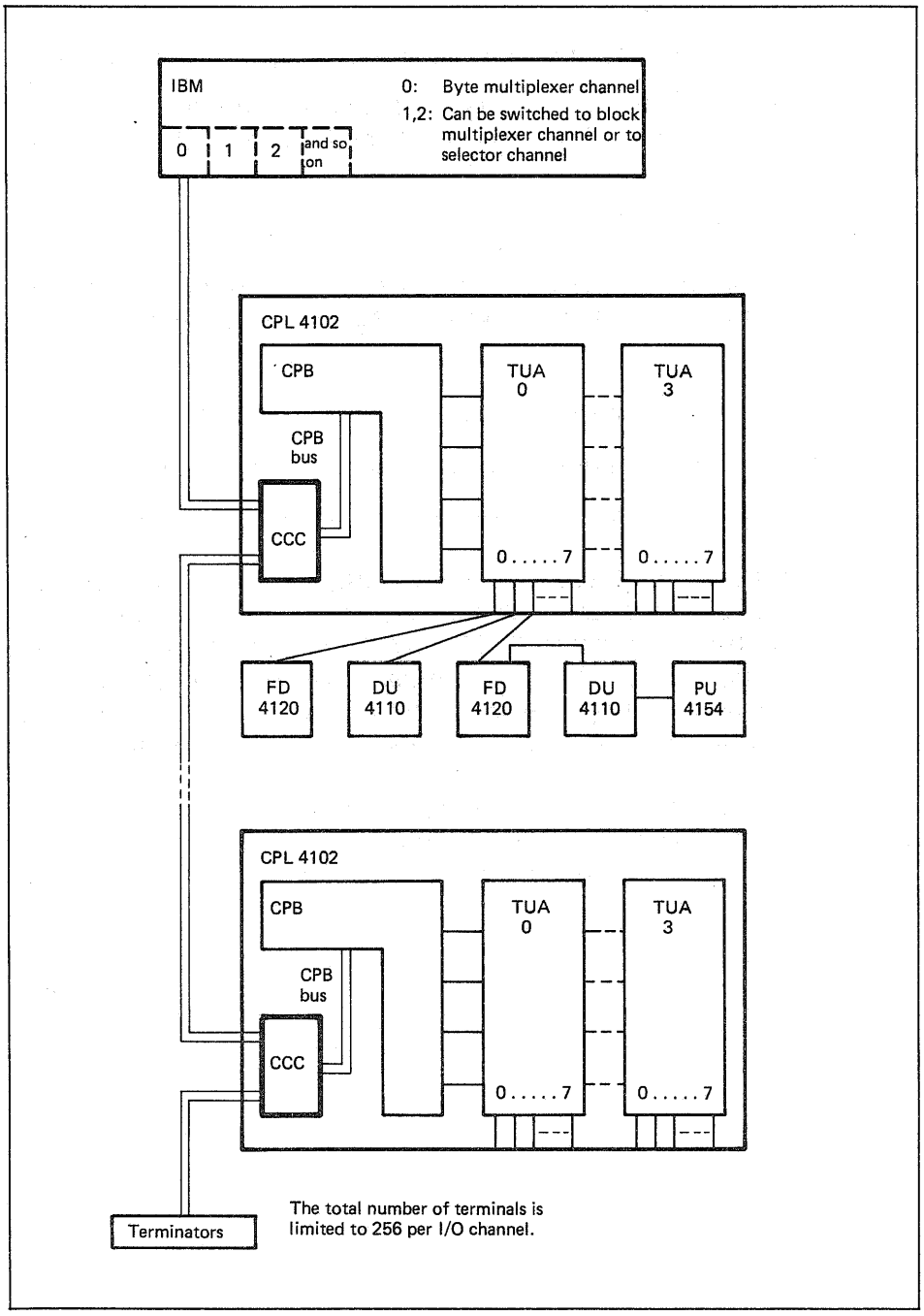


Fig. 2. Principles of connection to IBM channels

CPL 4102 can be connected to any of these channels. In multiplex mode, multiplexing can be made between bytes or blocks to or from different control units but not between bytes or blocks involving different devices within one control unit.

The channel program controls the various operations in the control unit by transmitting information across the I/O interface. This information consists of:

- An address byte that selects one control unit and one of the connected terminals
- Command bytes that specify which operation is to be carried out
- Data bytes which consists of the text that is to be displayed or printed out on the selected terminal or orders calling for the formatting of the buffer in the selected terminal
- Various control signals

In order to inform the channel program of general conditions in the terminal system, the control unit generates status bytes on the following occasions:

- In response to a command
- When an operation that has been called for has been executed
- When a terminal wishes to communicate with the computer

If two or more control units are attached to the interface, only one unit can communicate with the computer at any one time. Selection of a control unit is controlled by a signal (passing serially through all control units) that permits, sequentially, all units to respond to the signals provided by the channel. In this way, the different control units may be selected on a priority basis. CPL 4102 is normally placed at low priority.

Except for the above mentioned selection control signal, all communications to and from the computer channel occur over a common bus, i.e., any signal provided by the channel is available to all control units.

Input/Output Interface Lines

The I/O interface connects a computer channel to control units. External cables physically connect all control units in a chain with the first control unit connected to the channel. See Fig. 3.

The I/O interface lines and their uses are:

Name of line	Abbreviation	Uses
Bus out position P Bus out position 0 Bus out position 1 Bus out position 2 Bus out position 3 Bus out position 4 Bus out position 5 Bus out position 6 Bus out position 7	Bus out P Bus out 0 MSB Bus out 1 Bus out 2 Bus out 3 Bus out 4 Bus out 5 Bus out 6 Bus out 7 LSB	Bus out, used to transmit information (data, I/O device addresses, commands, control orders) from the channel to the control unit.
Bus in position P Bus in position 0 Bus in position 1 Bus in position 2 Bus in position 3 Bus in position 4 Bus in position 5 Bus in position 6 Bus in position 7	Bus in P Bus in 0 MSB Bus in 1 Bus in 2 Bus in 3 Bus in 4 Bus in 5 Bus in 6 Bus in 7 LSB	Bus in, used to transmit information (data, selected I/O device identifications, status information, sense data) from the control unit to the channel.
Address out Address in Command out Status in Service out Service in	Adr out Adr in Cmd out Sta in Srv out Srv in	Tags, used for inter-locking and controlling information on the bus, and for special sequences.
Operational out Operational in Hold out Select out Select in Suppress out Request in	Opl out Opl in Hld out Sel out Sel in Sup out Req in	Selection controls, used for the scanning of, or the selection of, attached I/O devices.
Clock out	Clk out	Metering control

Note that the out lines are always used for signals from the computer and in lines for signals from a control unit.

The tags, selection controls and metering control are lines in the same cable, called the tag cable.

Note: When a signal is said to be »up», »down», »raised», etc, reference is made only to the logic state of the signal, i.e. up (raised) means a logical one while down (dropped) means a logical zero. Do not confuse with actual voltage levels.

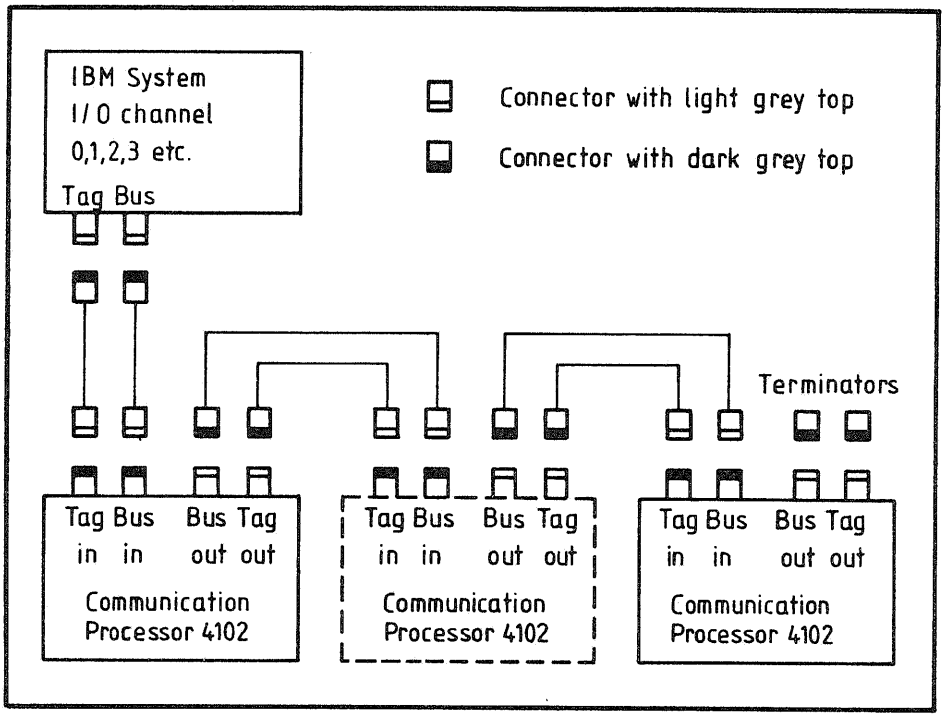


Fig. 3. Connections to IBM channel

Tag Lines

The tag lines are used to define the period during which information on Bus out or Bus in is valid. The address, status, data, etc byte is placed on the appropriate bus just prior to the rise of the tag line in question, but the bus information is not valid until the tag is up. Fig. 4 shows the transfer directions for different kinds of information.

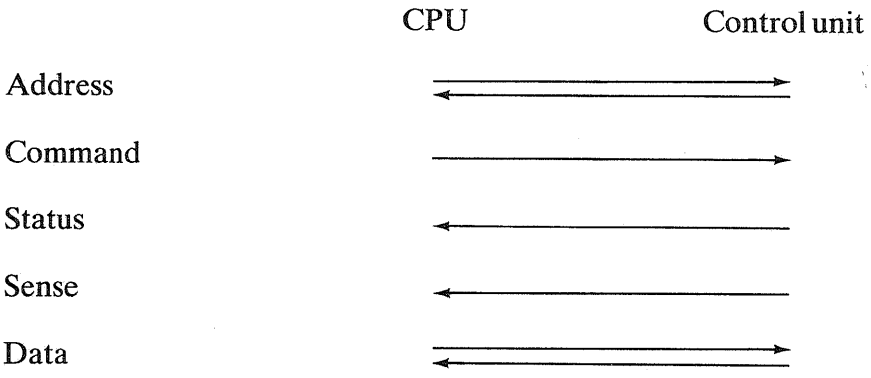


Fig. 4. Directions of information transfers.

Address out, Adr out is a signal to all connected control units, which indicates that the channel has placed an address on the Bus out. The address byte has been placed on the bus just prior to the rise of Adr out.

Address in, Adr in is a signal to the channel from the control units, indicating that a control unit has placed the address of the selected I/O device on Bus in just prior to the rise of Adr in.

Command out, Cmd out is a signal to all control units, indicating that the channel just has placed a command byte on Bus out. Command out can also indicate "proceed" or "stop".

Status in, Sta in is a signal from a selected control unit to the channel, indicating that it has placed a status byte on Bus in.

Service in, Srv in is a signal from a selected control unit to the channel, indicating either that the control unit has placed a byte on Bus in or that it is ready to accept a byte on Bus out.

Service out, Srv out is a signal from the channel meaning a response on Srv in or Sta in, indicating that the channel has accepted the byte on Bus in or has provided a byte on Bus out as a result of Srv in.

Selection control lines

Select out, Sel out and Select in, Sel in are in contrast to all other lines connected so that a closed loop is formed. It runs from the computer through all control units via a terminator block back through all control units to the channel. See Fig. 5.

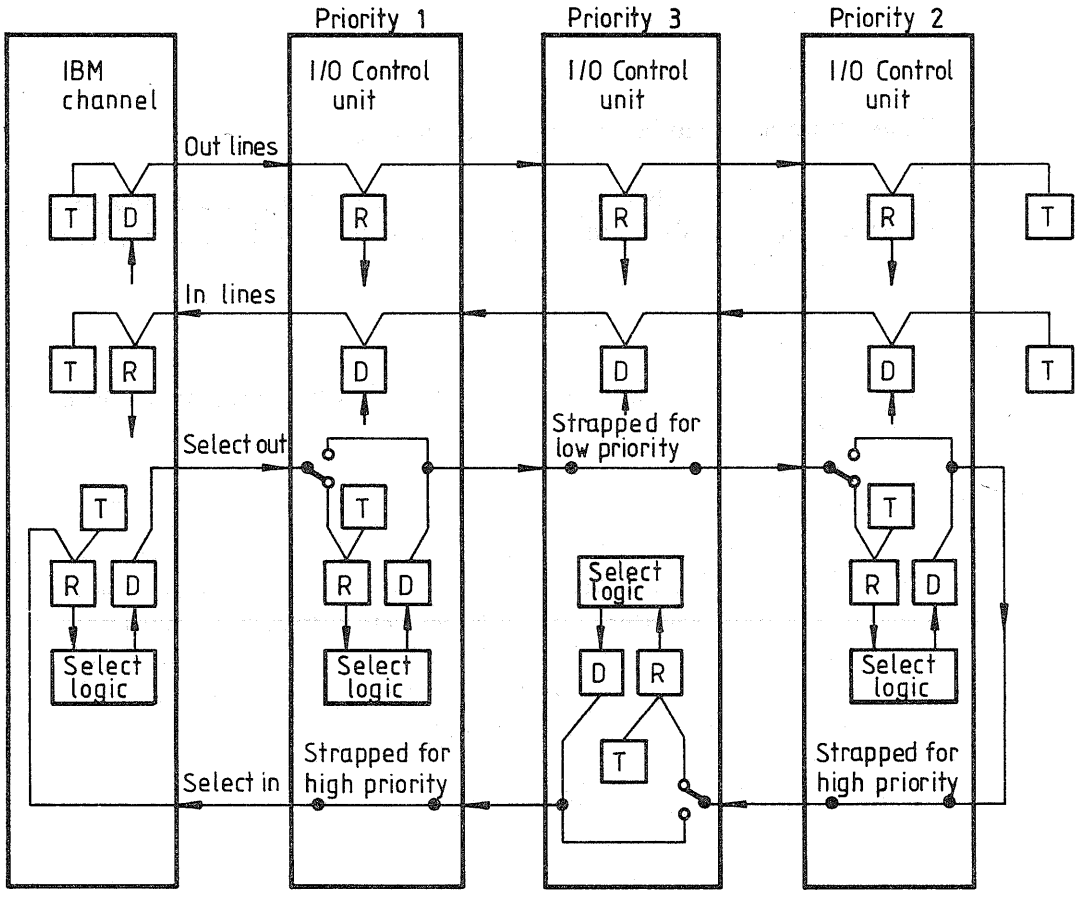


Fig. 5. Interconnections to the IBM I/O interface

The series connection of the control units means that Sel out is first raised at the first control unit after the channel. If the control unit does not wish to communicate with the channel, it generates propagated Sel out which is sent to the next control unit in the priority chain, etc.

Hold out, Hld out is a signal from the channel to all control units, which always appears together with Sel out. It is used for synchronization and for minimizing the propagation at the fall of Sel out by simultaneously clearing Sel out from all control units.

Operational out, Opl out. No lines from the channel, except Sup out, are valid until Opl out is up. Opl out down will cause general reset in CPL 4102 except when Sup out is up.

Operational in, Opl in is a signal to the channel, confirming that a control unit has been selected. The signal must stay up during the selection. Opl in can only rise when incoming Sel out is up and propagated Sel out is down, i.e. when Sel out is not sent to the succeeding control unit.

Suppress out, Sup out is a signal from the channel providing:

- Suppress data
- Suppress status
- Command chaining
- Selective reset

These functions will be explained further on.

Request in, Req in is a signal from a control unit, indicating that the control unit is ready to provide status or data and therefore awaits a selection sequence.

Metering control lines

Clock out, Clk out, when raised (by the channel) indicates activity on the buses.

Bus in and Bus out lines

Each bus is a set of nine lines consisting of eight information lines and one parity line. When a byte, transmitted over the interface, consists of less than eight information bits, as is the case with ASCII characters, the bits are placed in the highest-numbered contiguous bit positions of the bus i.e. as the least significant bits. Unused lines present logical zero. The parity bit (odd parity) always appears in the parity position.

Bus line EBCDIC	Binary position value	Hexadecimal position value	
P	P	P	
0	128	8	
1	64	4	High order
2	32	2	
3	16	1	
4	8	8	
5	4	4	Low order
6	2	2	
7	1	1	

Bus out is used to send addresses, commands, orders and data to the control units.

Bus in is used to send addresses, status information, detailed status information (sense) and data to the channel.

Detailed information about addresses, commands, status and sense can be found in the Reference Manual (document No FE411 – 810) sections Commands and orders and Local operation.

Communication Sequences

In the following description the data exchange is illustrated using timing diagrams and sequence diagrams showing small parts (sequences) of the interface procedures. All these diagrams are to be read in a similar way and therefore only the initial selection in burst mode is described in detail.

Burst mode and multiplex mode channels

Burst mode implies that a complete data transfer is made, from initial selection, through the transfer of data and including ending sequence. This means that one whole message or block is transferred in one row, without being interrupted by any other communication over the interface.

The channel decides which mode to operate in (burst or multiplex) and burst mode is normal at high speed I/O devices. A control unit can force burst mode on a multiplex channel by holding up Opl in. However, CPL 4102 can not force burst mode. The forced burst mode can be overridden by an interface disconnect sequence from the channel.

Multiplex mode implies that only one byte at a time is transferred to or from a certain device. The channel thereby shares its time between a number of I/O devices as in the following example:

Initial selection of A, initial selection of B, initial selection of C, byte to A, byte to C, byte to B, byte to A, byte to B, etc, ending sequence C, byte to A, ending sequence B, ending sequence A.

A burst mode channel, which is not capable of working in multiplex mode forces burst mode by holding up Sel out.

Data transfer from channel to control unit, burst mode

When the channel wants to transfer data to an I/O device, it performs an initial selection sequence (see Figs 6 and 7) to start the operation. Such a sequence starts with the channel placing the actual device address on Bus out and then raising Adr out, Hld out, and Sel out. The control unit then raises Opl in, which in this case means that the address on Bus out was recognized and that Sel out has not been propagated to the succeeding control unit. The channel then drops Adr out.

To indicate to the channel that the correct control unit has been selected, the control unit then places its address on Bus in and raises Adr in.

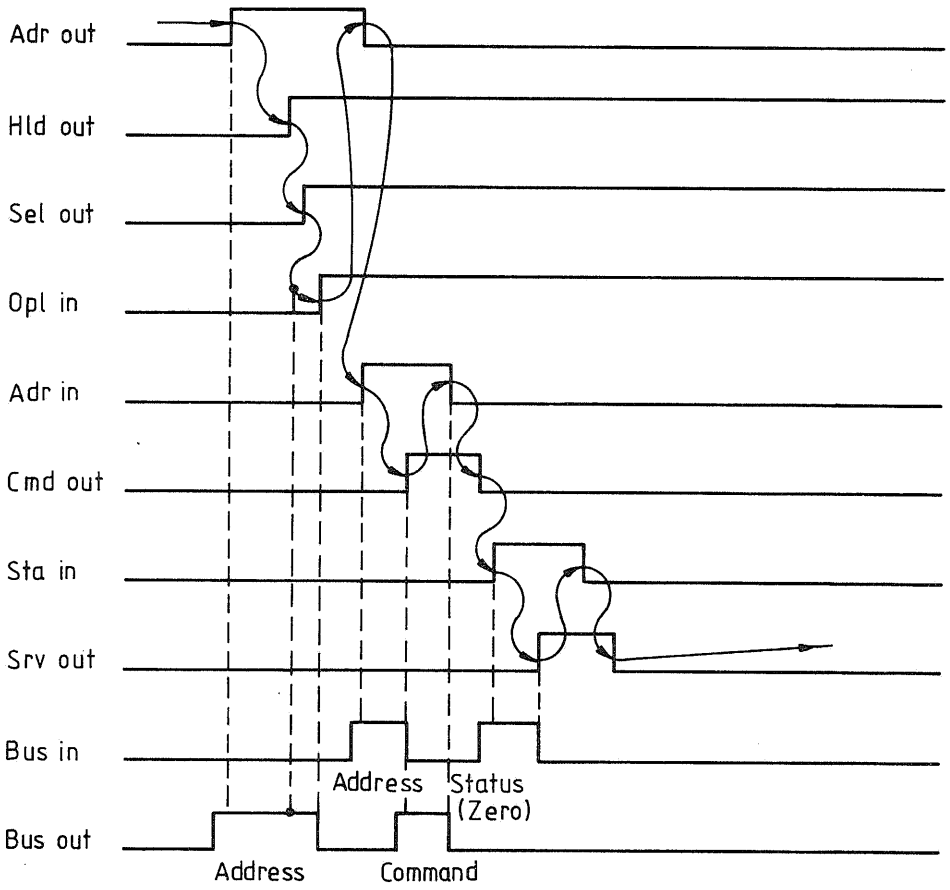


Fig. 6. Initial selection timing, burst mode

This is accepted by the channel when Cmd out rises, which also indicates that the command byte on Bus out is valid. The command byte informs the control unit what it is expected to perform.

The control unit accepts the command and indicates this by dropping Adr in, whereupon the channel drops Cmd out.

The selected control unit now places a status byte (zero) on Bus in and raises Sta in (see "Local operation: Initial Status" in the Reference manual). This is to inform the channel that the I/O device is ready to accept a message.

The channel acknowledges the status byte by raising Srv out, whereupon the control unit drops Sta in and the channel drops Srv out.

The communication then begins with a data transfer sequence (see Figs 8 and 9) during which write control character, buffer addresses, text bytes etc. are transferred to the control unit (and the I/O device). The control unit raises again Srv in when the channel has sent the last data byte, but this time the channel responds by raising Cmd out. This indicates that an ending sequence shall be made.

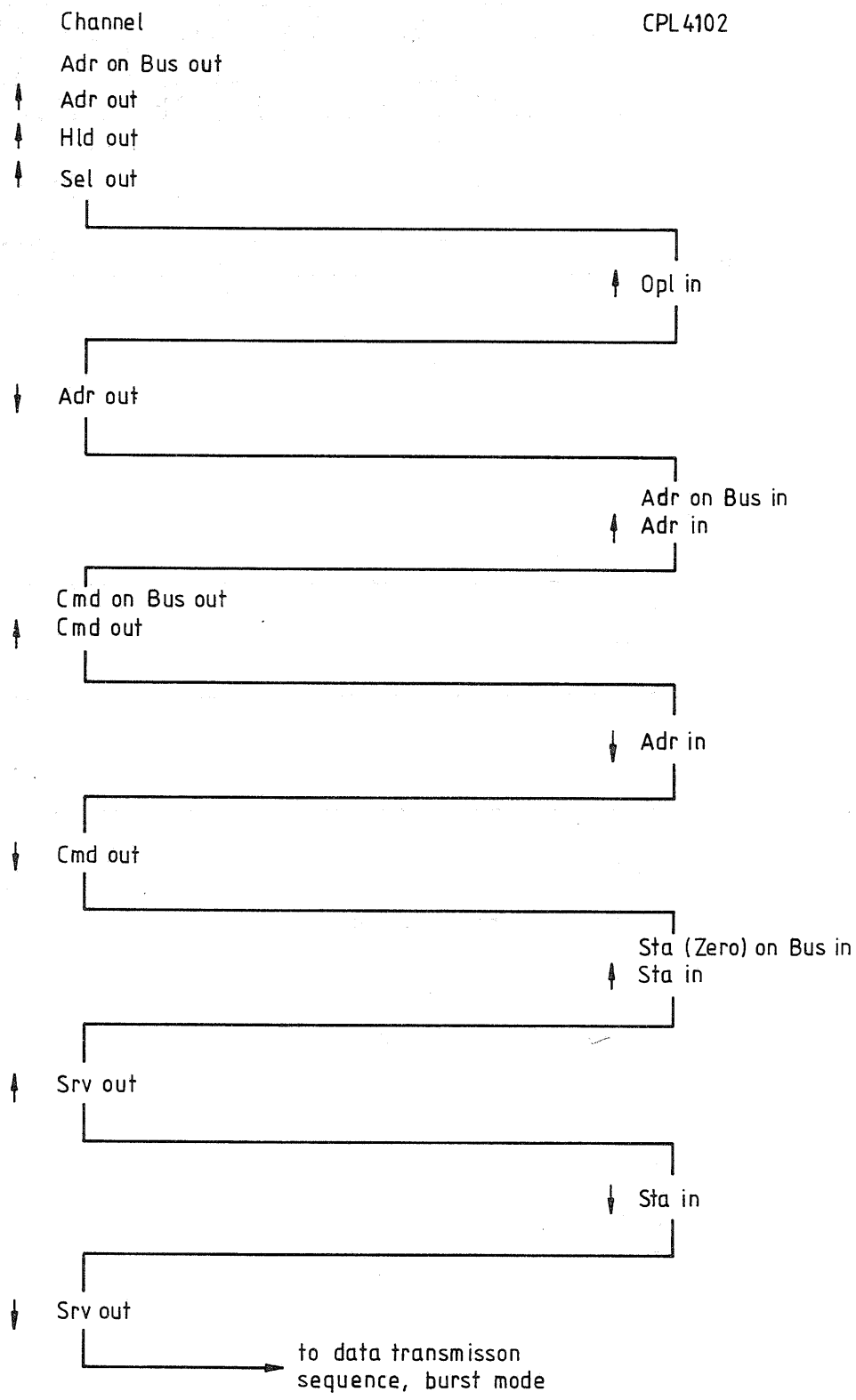


Fig. 7. Initial selection sequence, burst mode

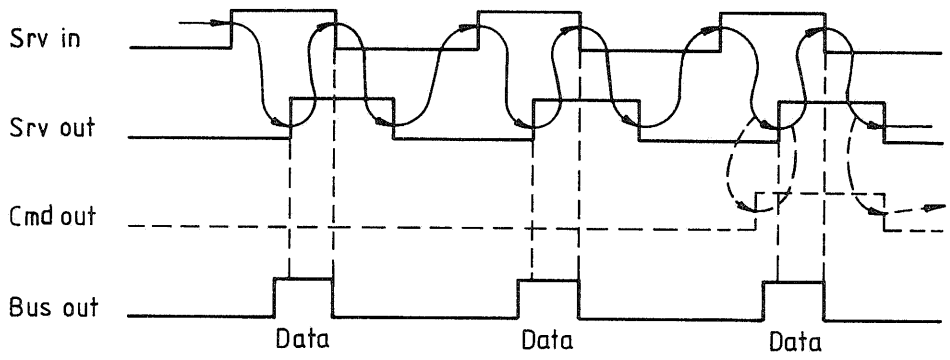


Fig. 8. Data to control unit transfer timing, burst mode

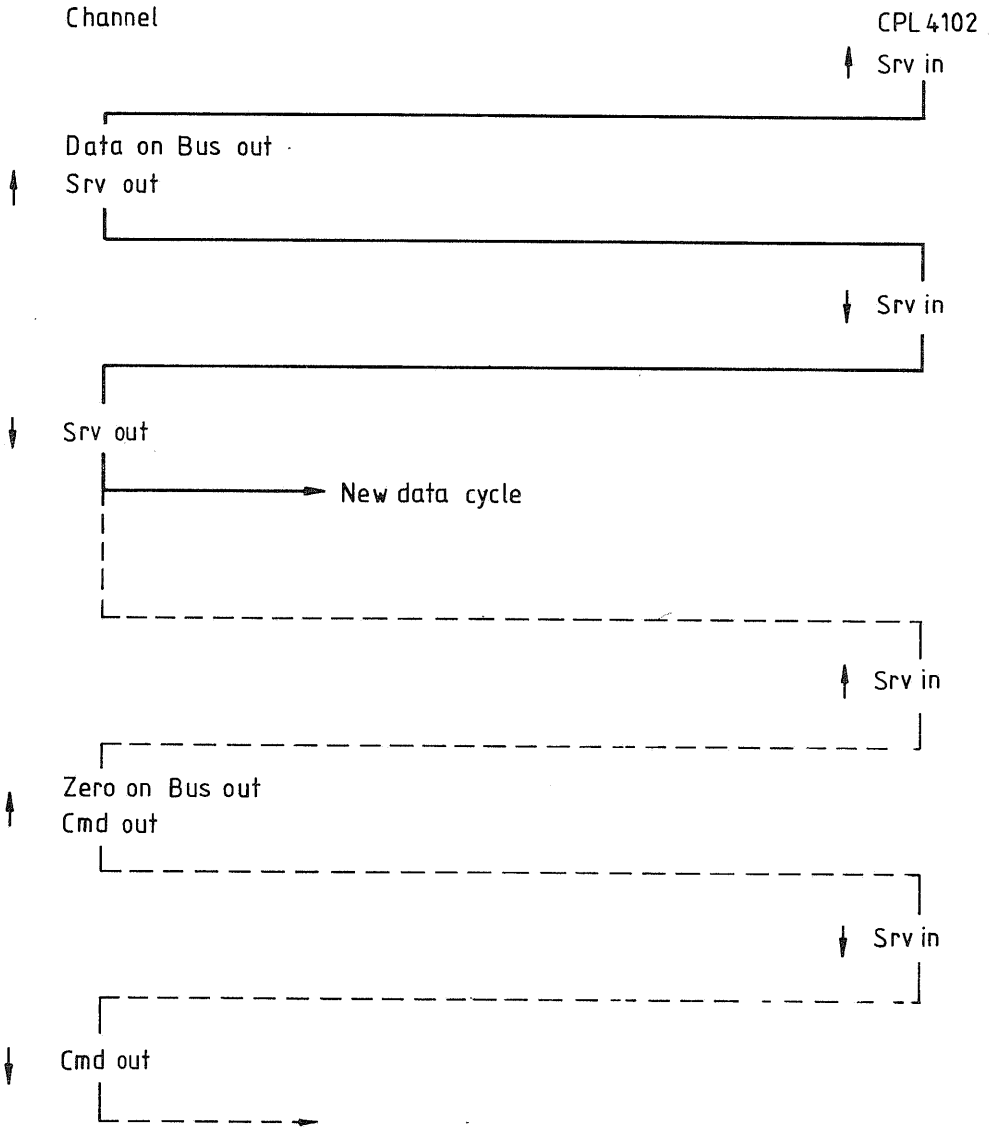


Fig. 9. Data to control unit transfer sequence, burst mode

During the ending sequence (see Figs 10 and 11) the control unit transfers its synchronous ending status (CE) to the channel, whereupon Hld out, Sel out, Opl in drop. When the writing operation in the I/O device is finished, the control unit sends an asynchronous status (DE). (See Figs 18 and 19.) In this sequence the raise of Cmd out indicates; proceed.

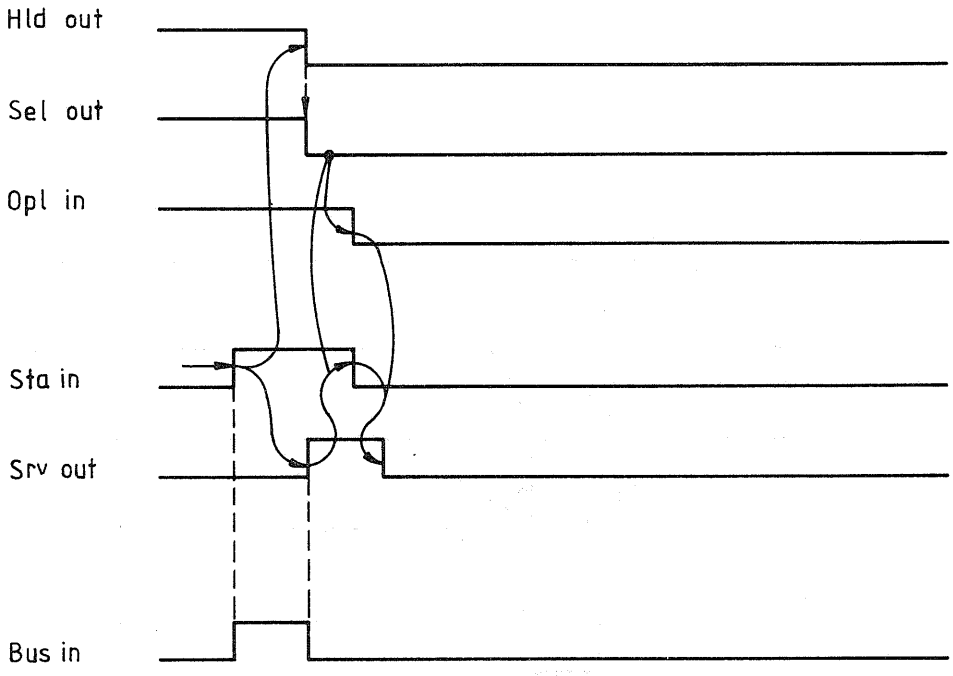


Fig. 10. Ending timing, burst mode

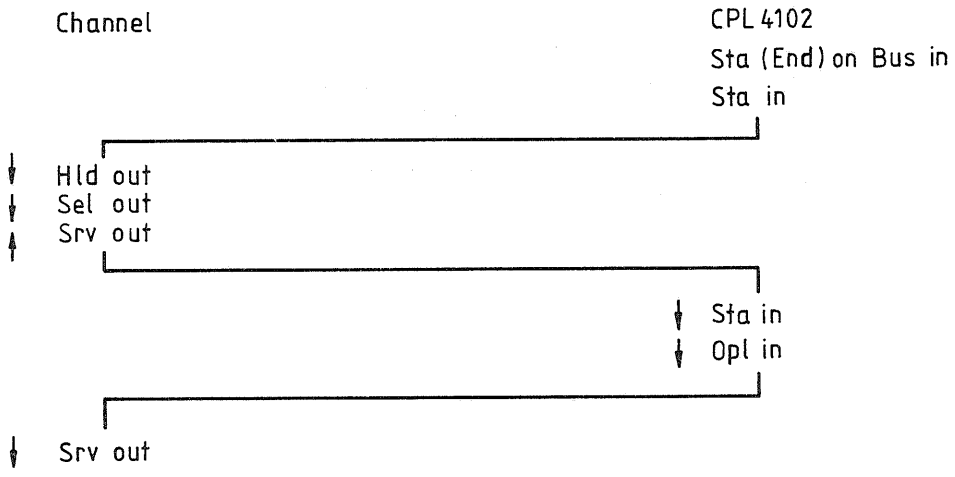


Fig. 11. Ending sequence, burst mode

Data transfer from channel to control unit, multiplex mode

First the channel makes an initial selection (see Figs 12 and 13). In multiplex mode, the Hld out and Sel out drop as soon as the control unit has raised Adr in. Therefore, only the command can be sent in the initial selection sequence.

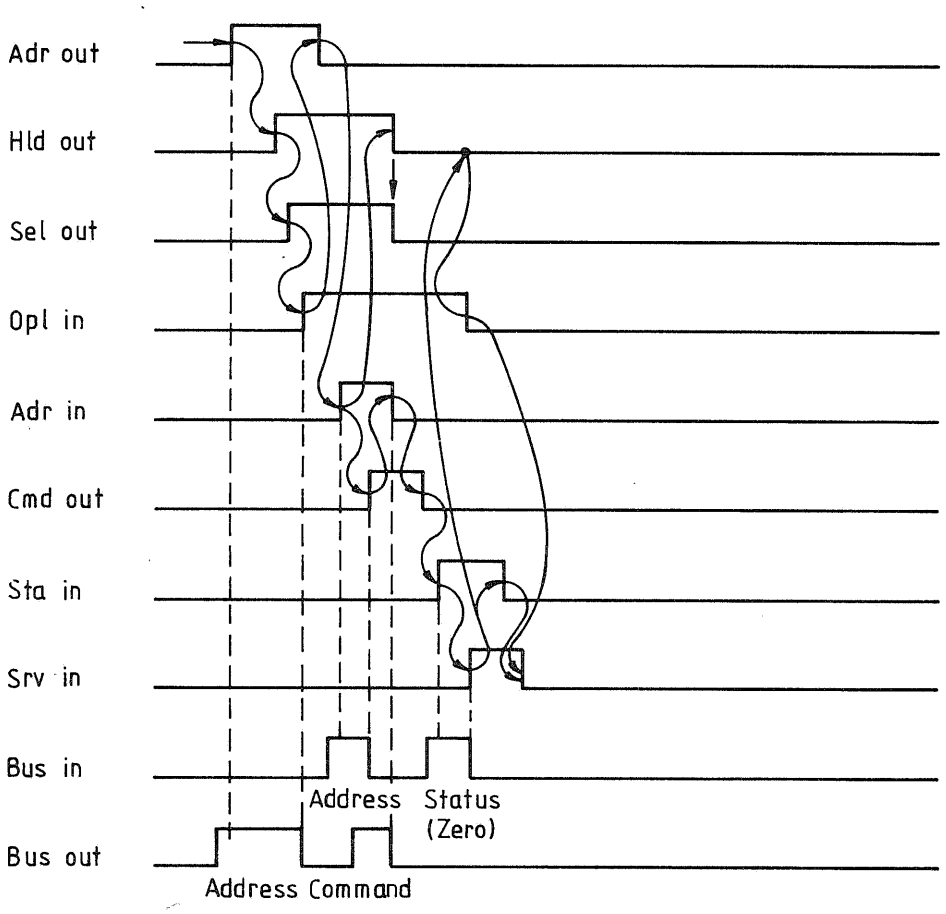


Fig. 12. Initial selection timing, multiplex mode

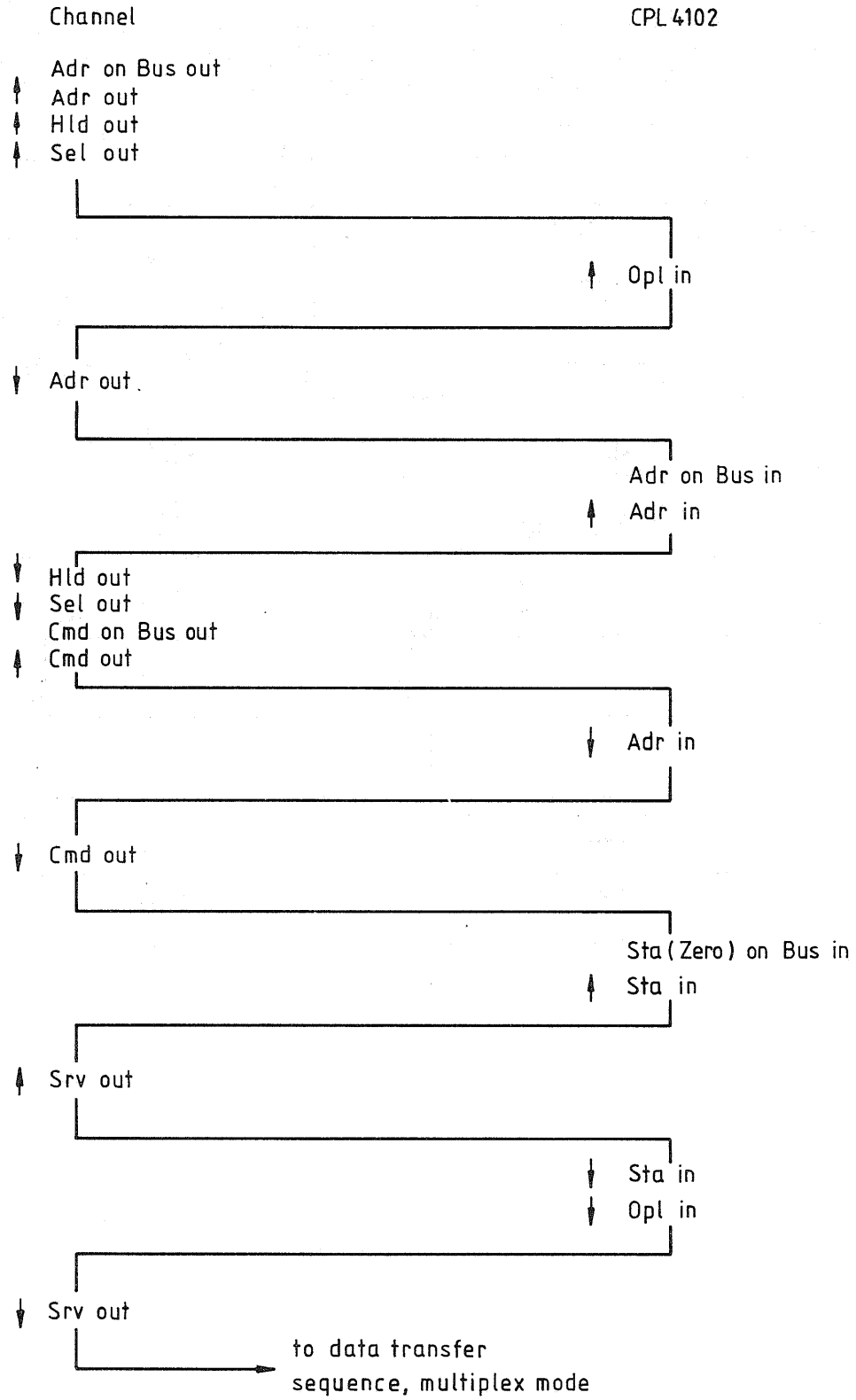


Fig. 13. Initial selection sequence, multiplex mode

As soon as the control unit is ready to accept a byte, it raises the Req in (see Figs 14 and 15). The Cmd out tag is raised to indicate proceed, no command is transferred since that was made in the initial selection. One byte is transferred each time Req in is raised.

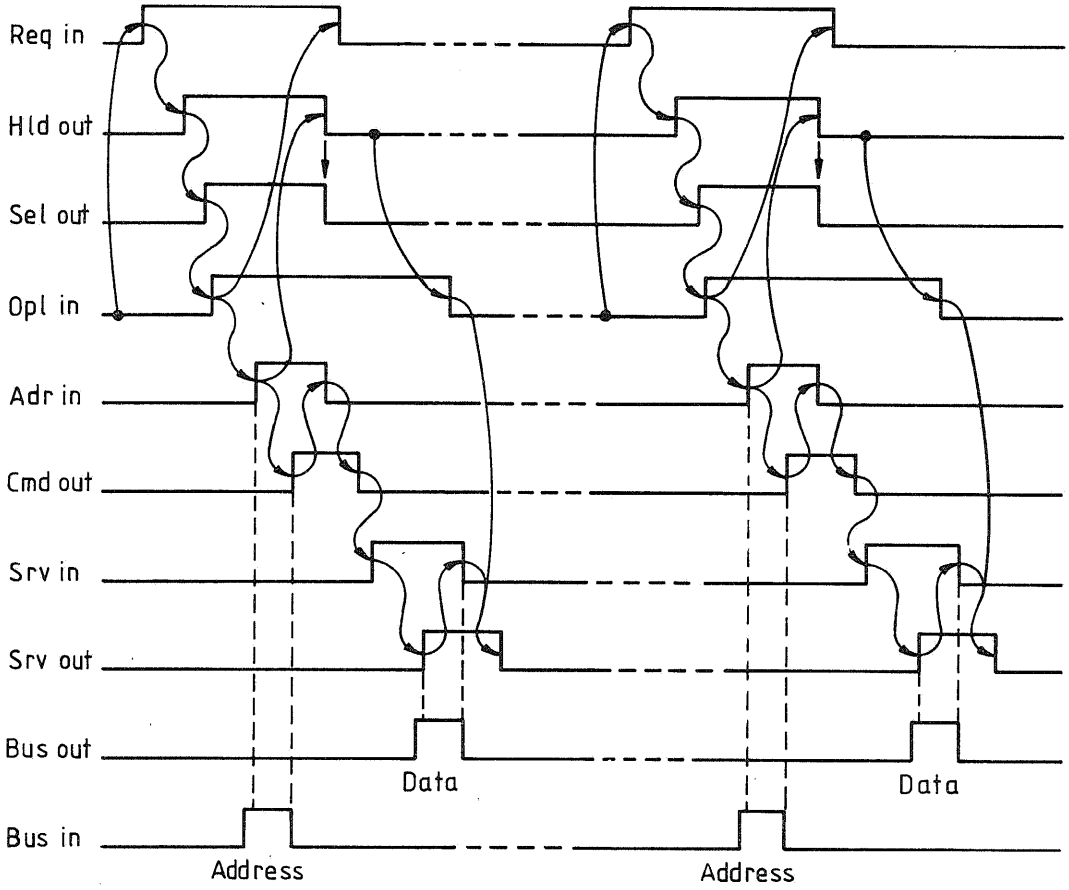


Fig. 14. Data to control unit transfer timing, multiplex mode

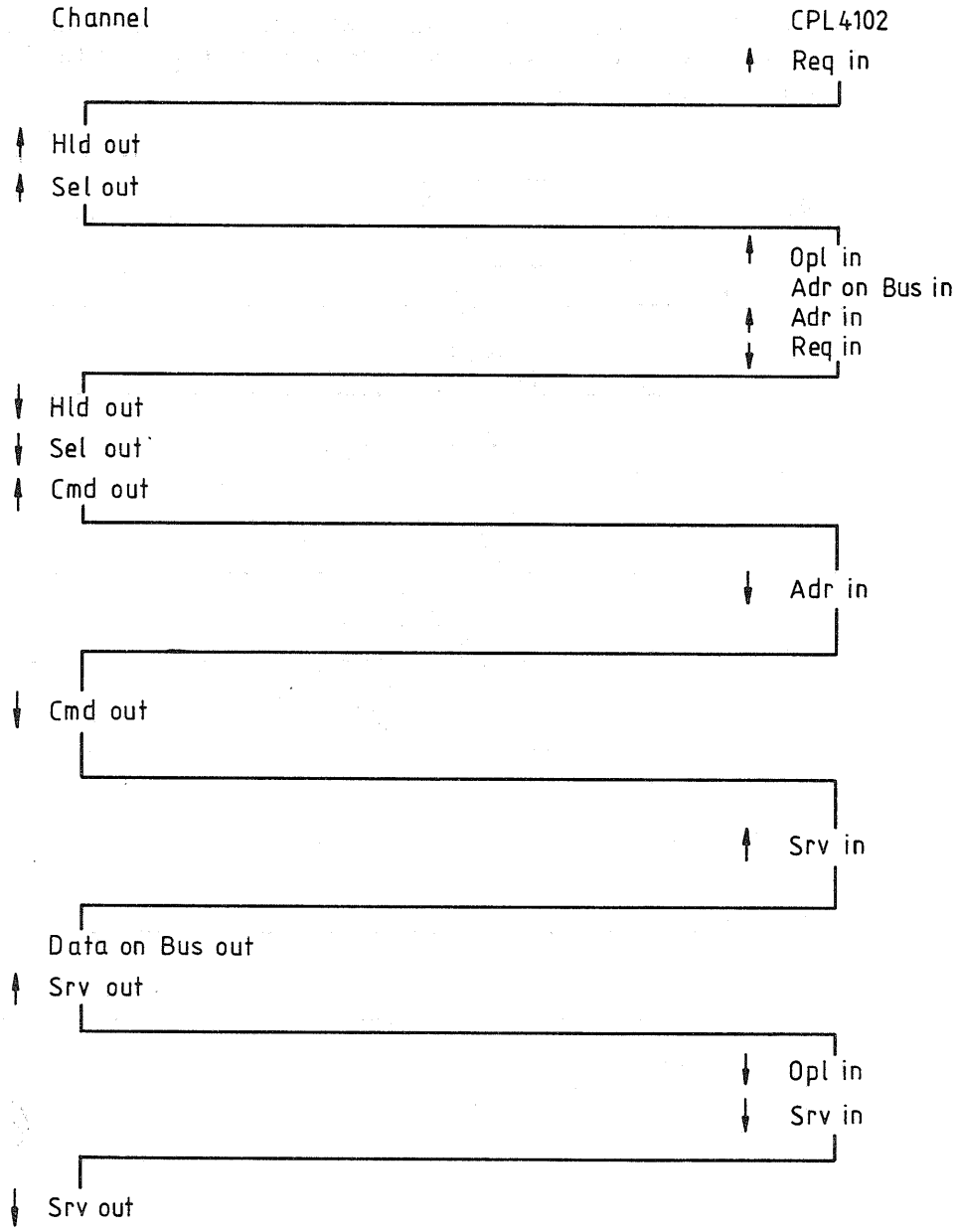


Fig. 15. Data to control unit transfer sequence, multiplex mode

When the channel has sent all bytes to the control unit, it performs the interface stop sequence (see Figs 16 and 17). In this sequence the channel raises the Cmd out twice, the first time to indicate proceed and the second time to indicate stop.

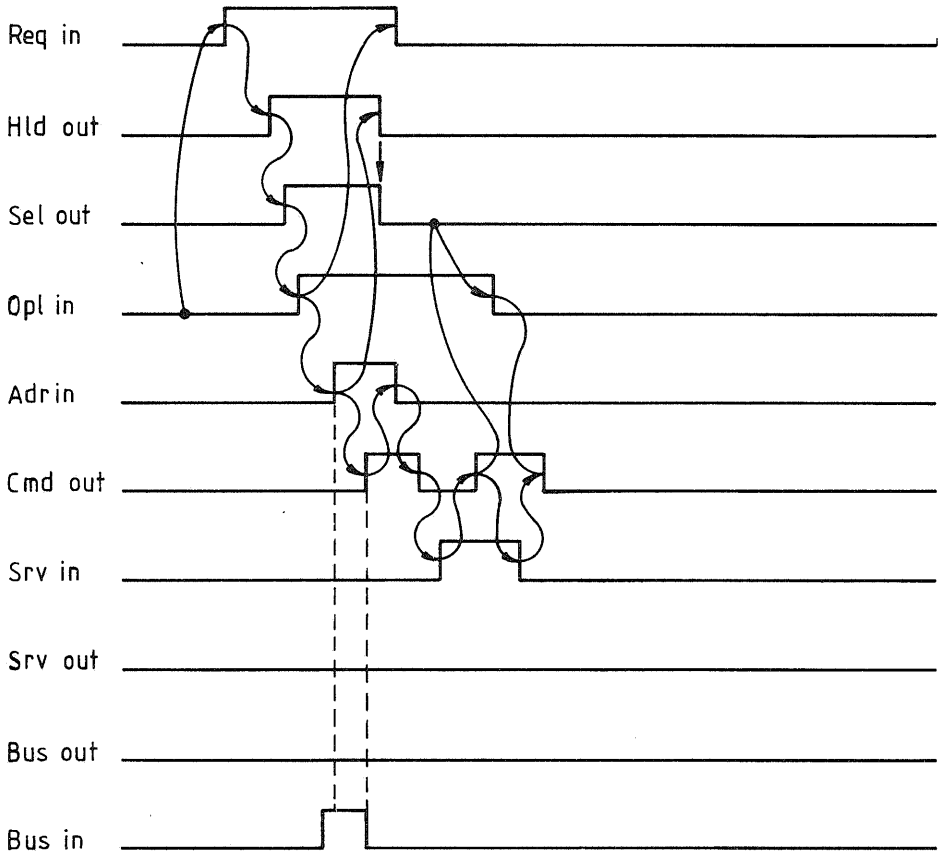


Fig. 16. Interface stop timing, multiplex mode

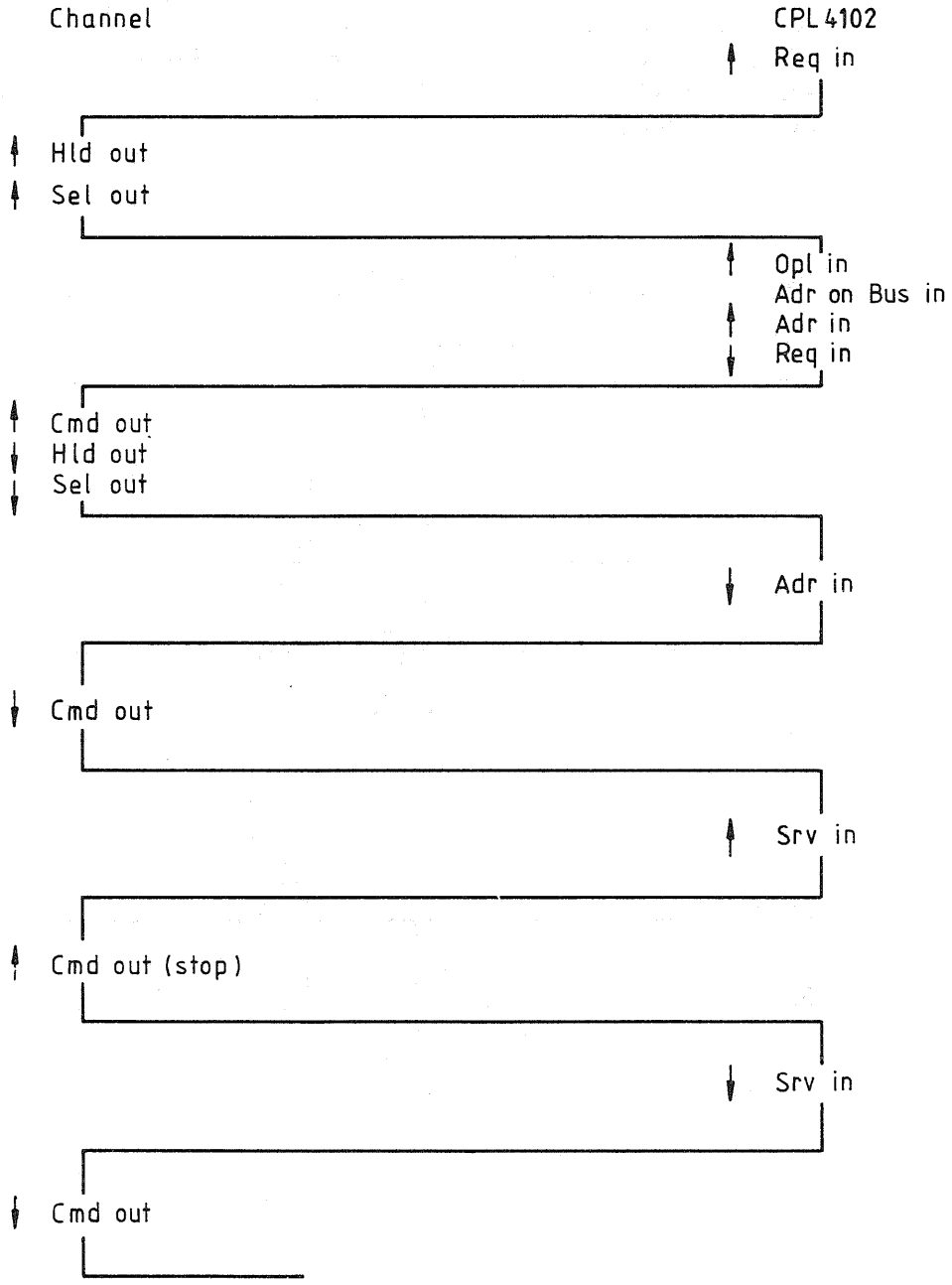


Fig. 17. Interface stop sequence, multiplex mode

The ending sequence in multiplex mode after write type operations are two status transfers according to the status sequence (see Figs 18 and 19). The first transfer includes the synchronous ending status (CE) and the second, which is performed after the operation is concluded in the I/O device, includes DE.

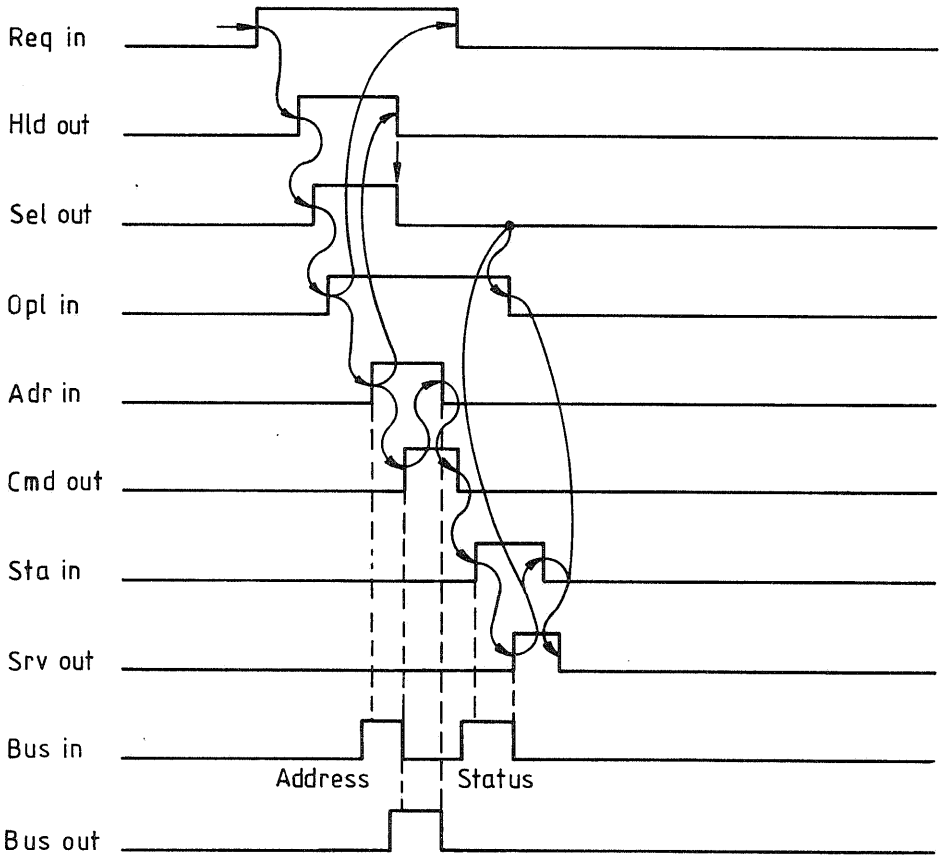


Fig. 18. Status timing

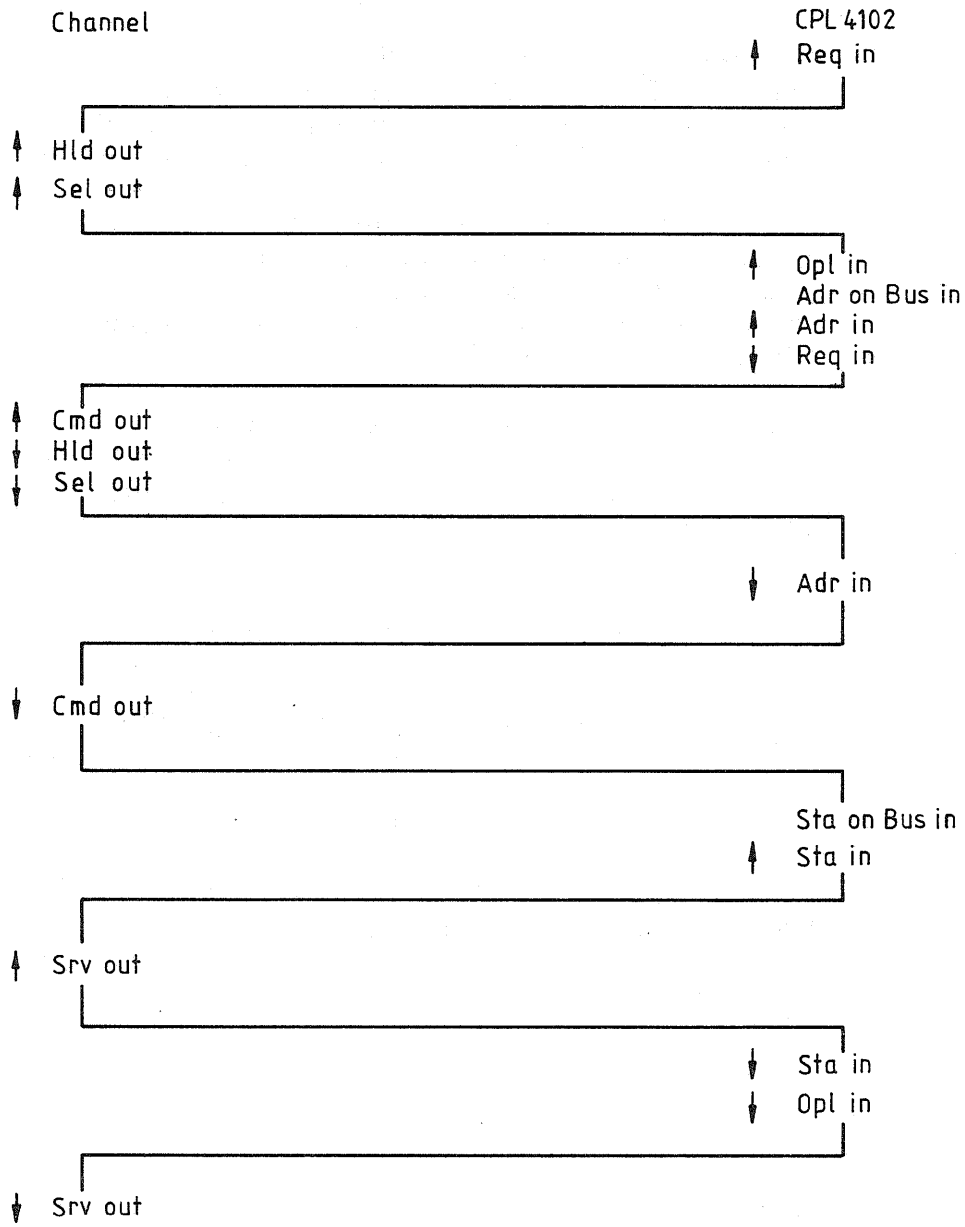


Fig. 19. Status sequence

Data transfer from control unit to channel, burst mode

A transfer from a control unit to the channel always starts with the control unit raising Req In (see Figs 18 and 19). This causes the channel to raise Hld out and Sel out. The control unit thereby identifies itself by the address and then feeds status Attention, indicating a request for service.

The channel thus performs an initial selection (see Figs 6 and 7) transferring a read type command. Thereafter, the communication commences with a data transfer sequence (see Figs 20 and 21). When the last byte is sent, the control unit places the ending status on the bus (see Figs 10 and 11) whereby the communication ends.

The channel can at any time during the communication break the transfer by raising Cmd out instead of Srv out and thereby indicate stop. The control unit then places the ending status (CE + DE) on the bus (see Figs 10 and 11).

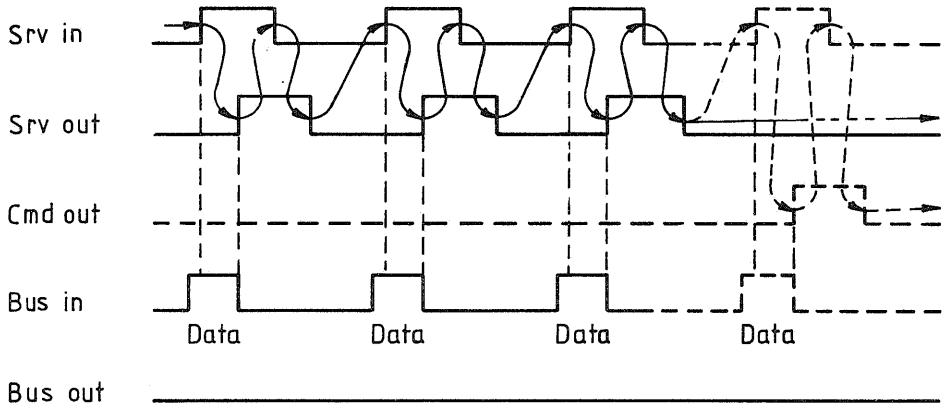


Fig. 20. Data to channel transfer timing, burst mode

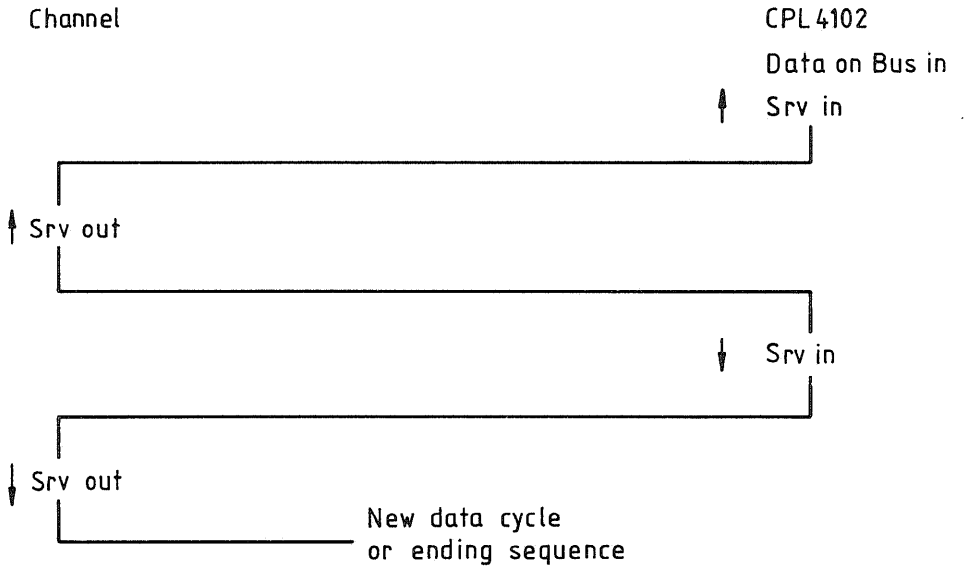


Fig. 21. Data to channel transfer sequence, burst mode

Data transfer from control unit to channel, multiplex mode

A data transfer to the channel in multiplex mode also starts by a status transfer (see Figs 18 and 19). Then follows an initial selection (see Figs 12 and 13) and data transfers (see Figs 22 and 23). When the last byte is transferred, the control unit sends the ending status (CE + DE) according to the status sequence (see Figs 18 and 19).

The channel can at any time break the transfer by raising Cmd out instead of Srv out (see Figs 16, 17, 22 and 23). The byte sent from the control unit is in such case neglected. The control unit then transfers ending status (CE + DE, sequence in Figs 18 and 19).

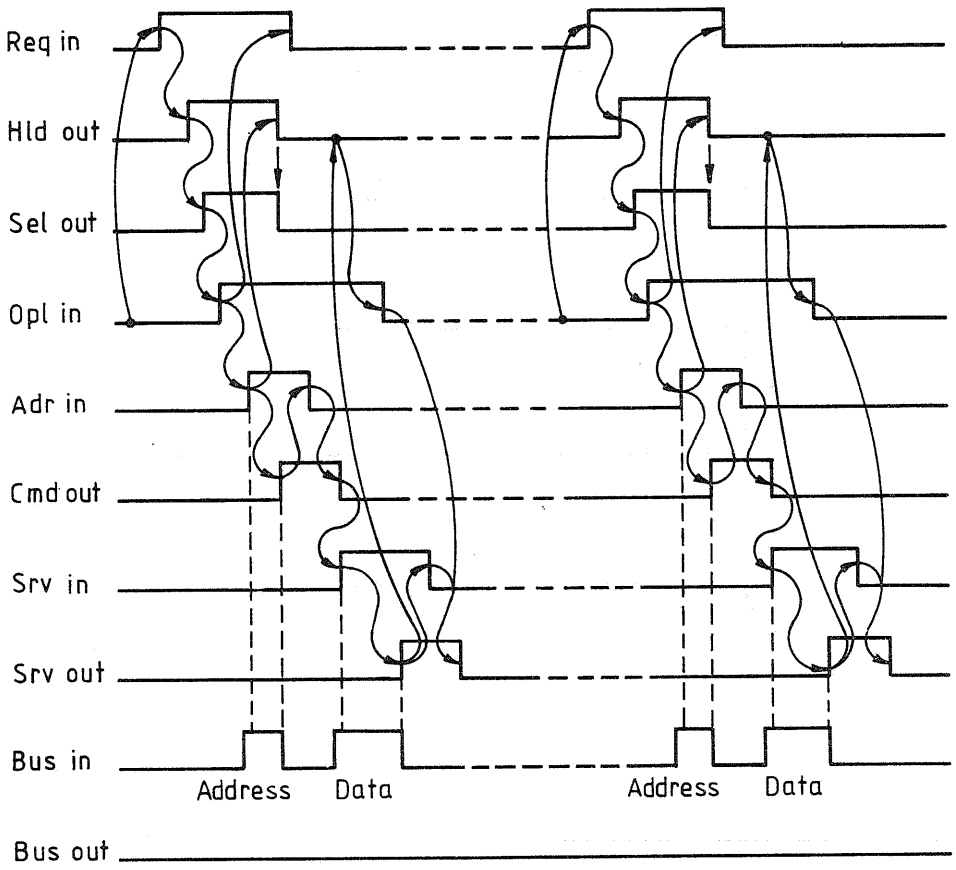


Fig. 22. Data to channel transfer timing, multiplex mode

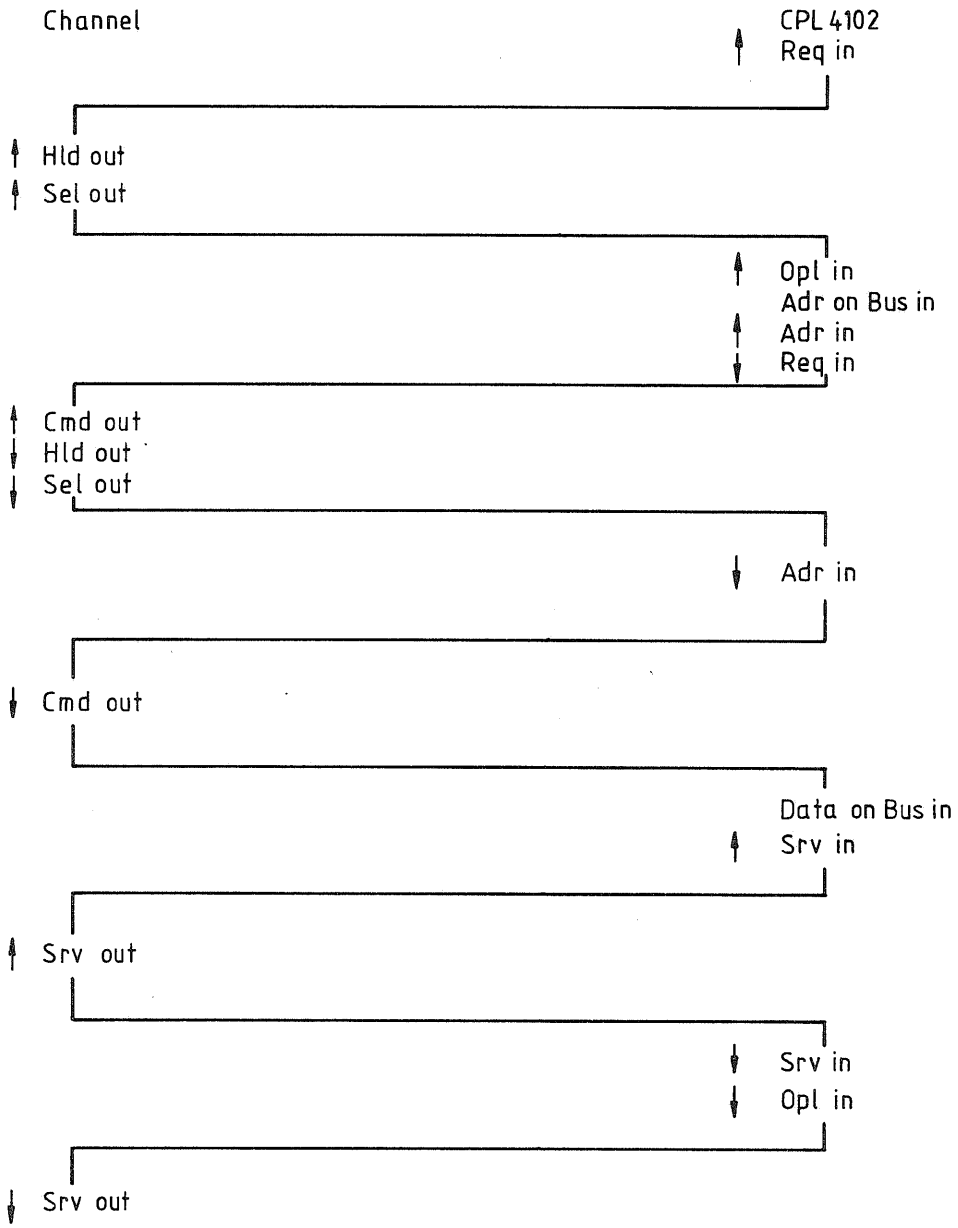


Fig. 23. Data to channel transfer sequence, multiplex mode

Control unit busy (short busy)

If the channel tries to address an I/O device, the control unit of which is busy or has a status concerning another I/O device waiting, a control unit busy sequence is made (see Figs 24 and 25).

When the control unit at hand recognizes the address on Bus out, it feeds control unit busy status on Bus in and rises Sta in. The channel thereby drops Adr out when the busy status has been recognized.

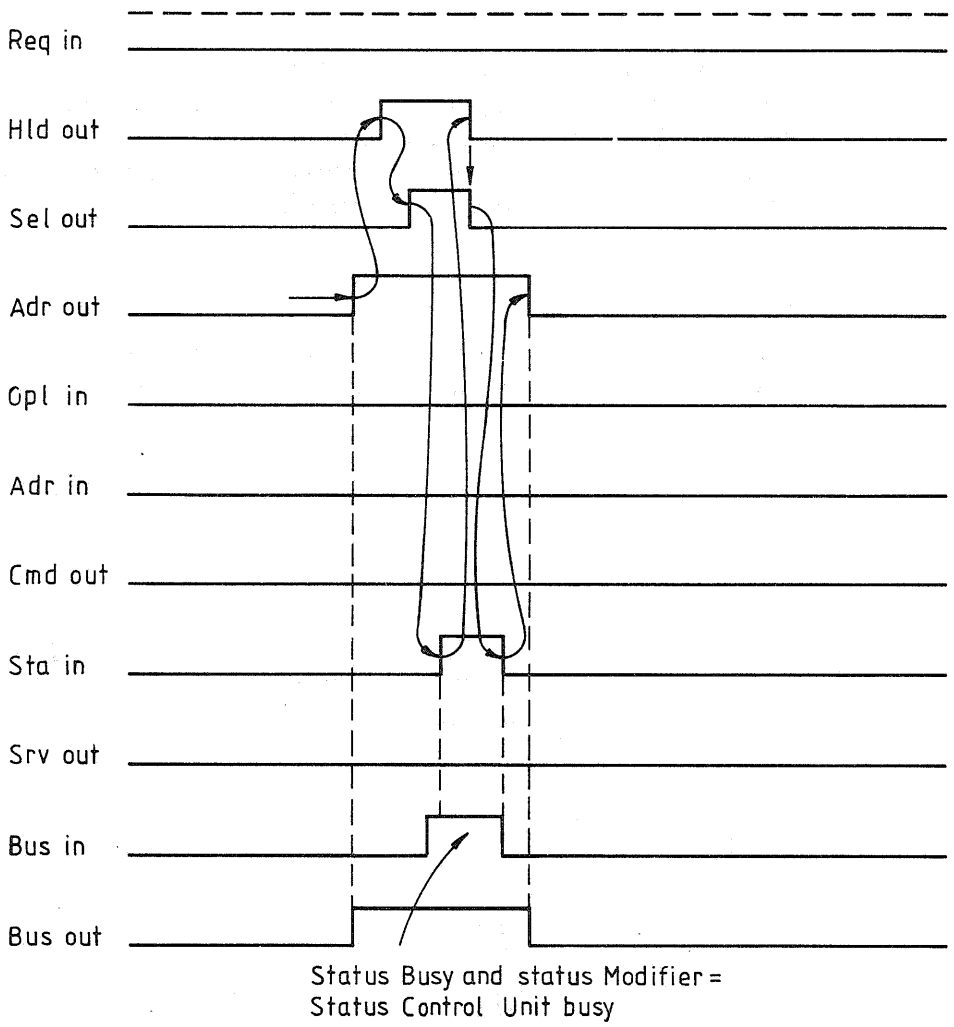


Fig. 24. Control unit busy timing

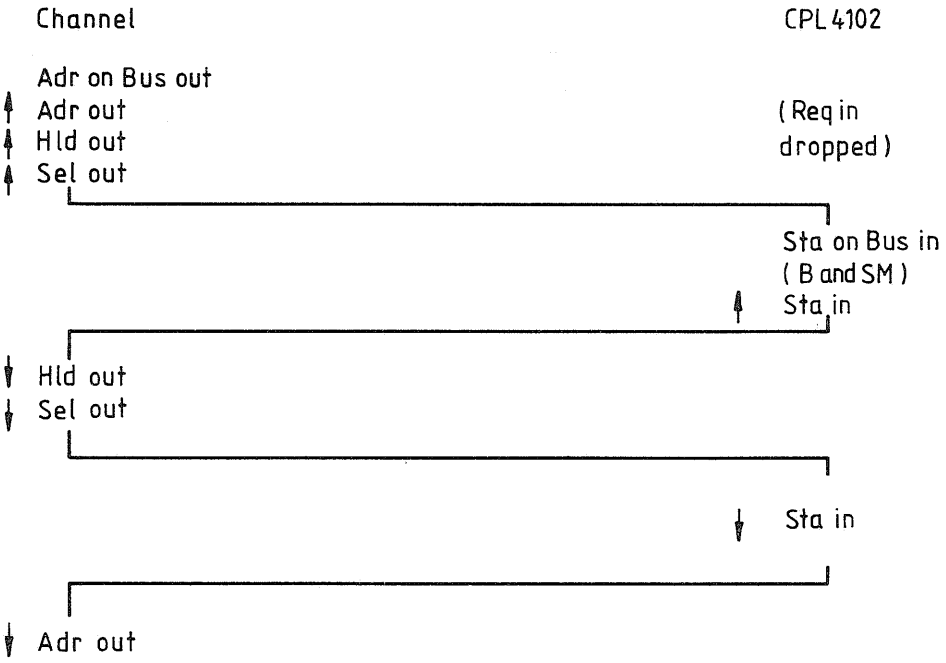


Fig. 25. Control unit busy sequence

Device busy

If the channel tries to address an I/O device, which is occupied by e.g. a print operation or by a pending (waiting) status, busy status is placed on Bus in and Sta in is raised as a response to the command from the channel during the initial selection. The situation is shown in Figs 26 and 27. Note that the command is not accepted.

Suppress data

If the channel is not able to accept a new data byte it can suppress the next byte if it raises Sup out before Srv out is dropped.

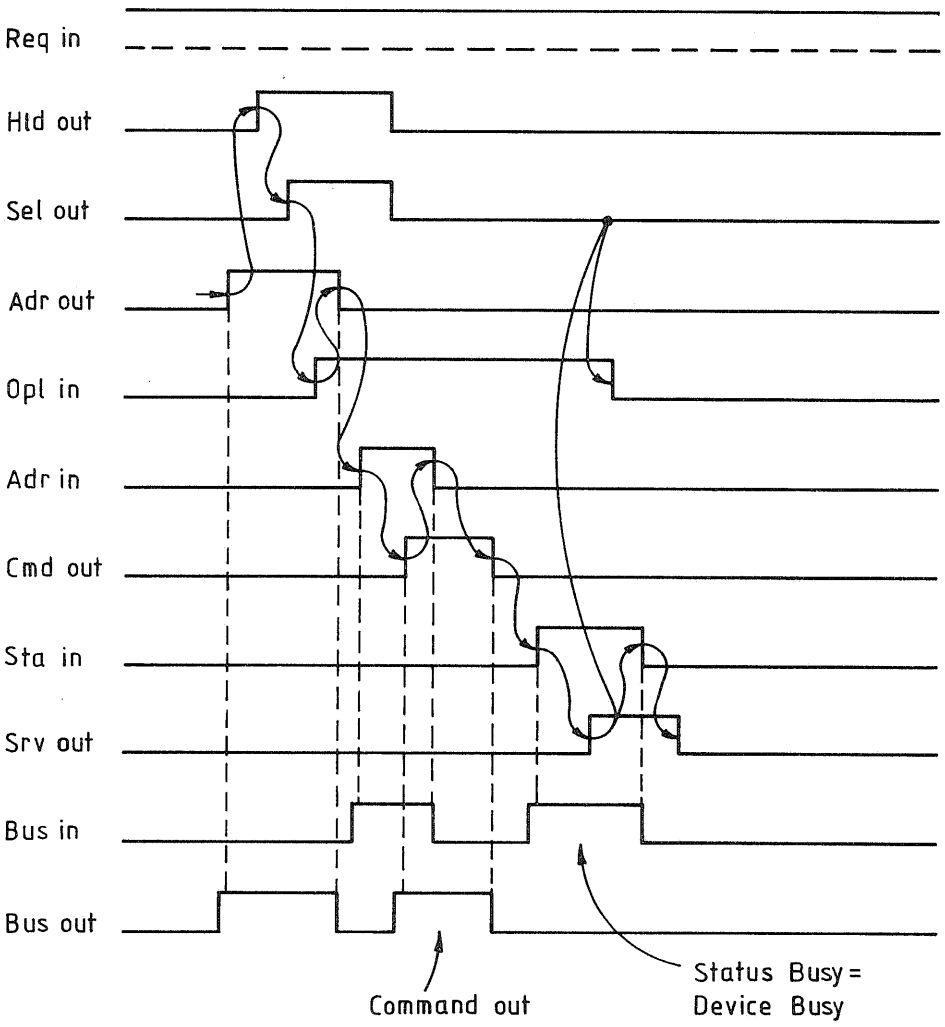


Fig. 26. Device busy timing

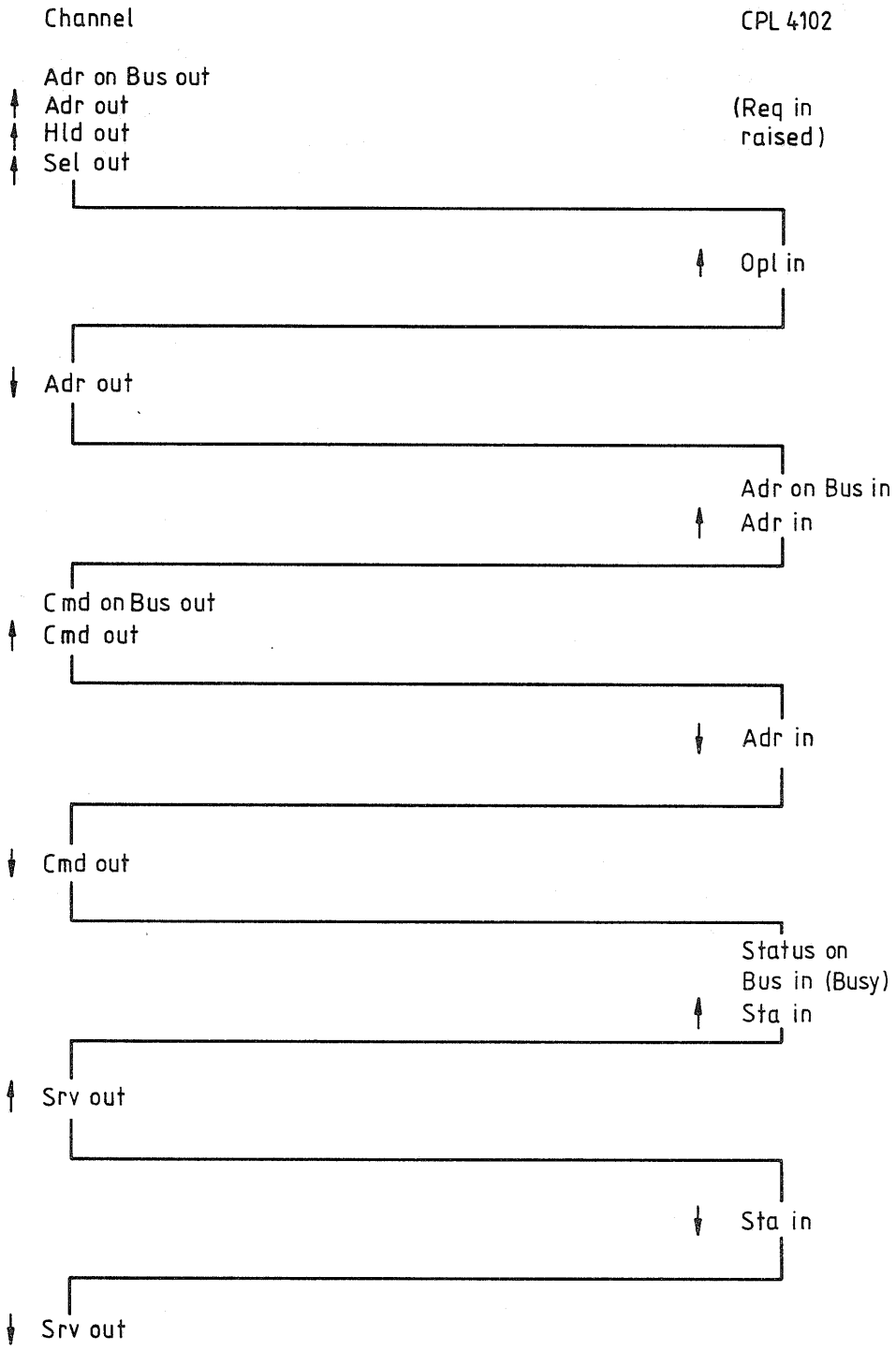


Fig. 27. Device busy sequence

Stack status and suppress status

Whenever the channel is unable to handle a status, it may raise Sup out. The control unit must then drop Req in if the status that it wants to transfer is suppressible. Status containing CE or DE is normally not suppressible until it is stacked. To stack status Cmd out is raised (see Figs 28 and 29). Note that in burst mode, Sel out and Hld out must be dropped before the raise of Cmd out.

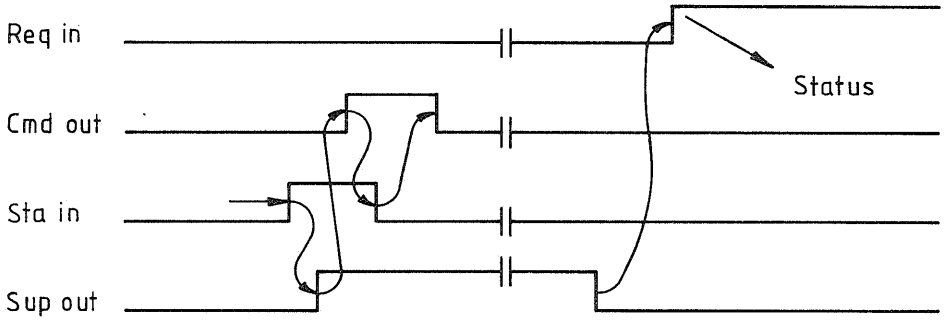


Fig. 28. Stack status and suppress status timing

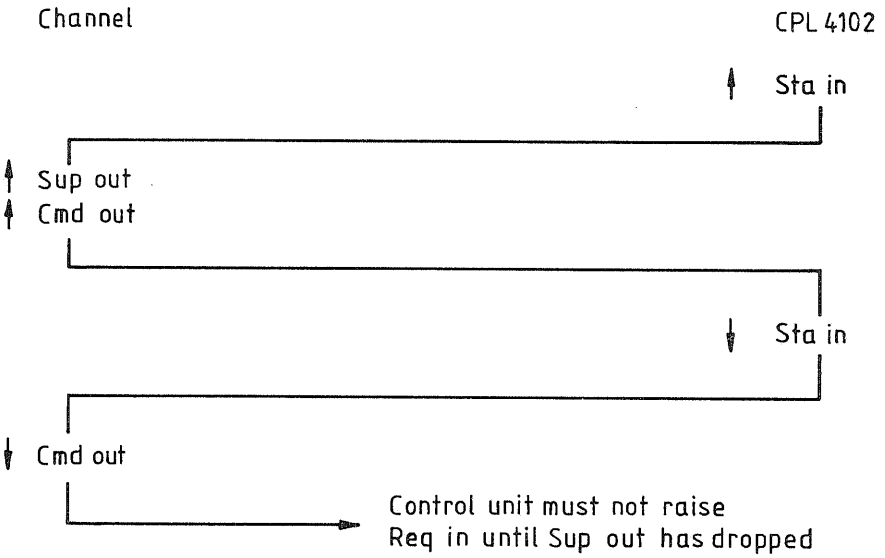


Fig. 29. Stack status and suppress status sequence

Selective reset

The I/O device being selected can be reset by the channel if it raises Sup out and drops Opl out. This affects the status of the selected I/O device and concludes the CPL operations (DE is not sent). Other units do not react to the drop of Opl out since Sup out is up.

Command chaining

The channel can execute a number of different operations in sequence at the same terminal. This is made by command chaining which implies that the channel stops any attempt from the control unit to initiate a selection regarding any of its own I/O devices until all chained operations are completed. This is achieved if the channel raises Sup out when the control unit generates end status. Sup out must remain up until the control unit raises Opl in within the following initial selection.

Interface disconnect

The channel can force the control unit to remove all signals from the interface by having Adr out up when Sel out and Hld out are dropped (see Fig. 30). The control unit will then normally raise Req in for subsequent sending of the ending status.

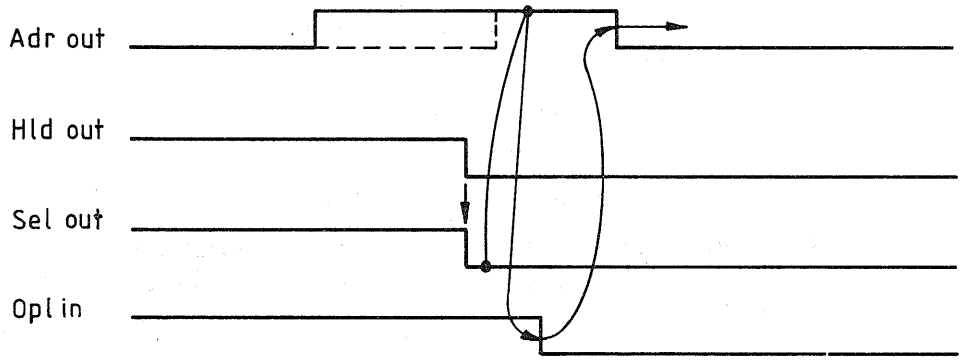


Fig. 30. Interface disconnect timing

Brief Outline

General

Basic design

The two printed circuit boards of the channel communication controller, CCC, contain interface control circuits and data buffering between the channel and the communication processor board, CPB. The block diagram, Fig. 31, shows that the various function blocks communicate via two bus systems, the CCC bus, internal to the CCC; and the CPB bus, between the CCC and the CPB. A third printed circuit board, CCC interconnection board, CIB, is not shown in Fig. 31. The CIB is basically a backplane board with interconnections.

CCC 2 includes:

- Receivers and parity check for incoming bus, control, and tag signals from the channel.
- Drivers and parity generation for outgoing bus, control, and tag signals to the channel.
- Select logic and address decoder to determine if the control unit is selected by the channel.
- Line buffer, to store and forward data to/from the channel.
- Address counter, to address the line buffer during data transfers (read or write) between the channel and the line buffer. (During read/write between the line buffer and the CPB, the line buffer is addressed from the CPB address bus.)
- Reset logic for power on and reset detection.
- A test connector (P4) for connection of a recorder for analysis of bus and tag activity.

CCC 1 includes:

- Microprogram controller, controlled by instructions from the microprogram and by test conditions, generating the address to the microprogram PROM.
- Microinstruction PROM storing 2k 64-bit instructions forming the CCC microprogram.
- Pipeline register, storing an instruction while the result of the previous instruction is taken care of, and while the next instruction is made available at the output of the microinstruction PROM.

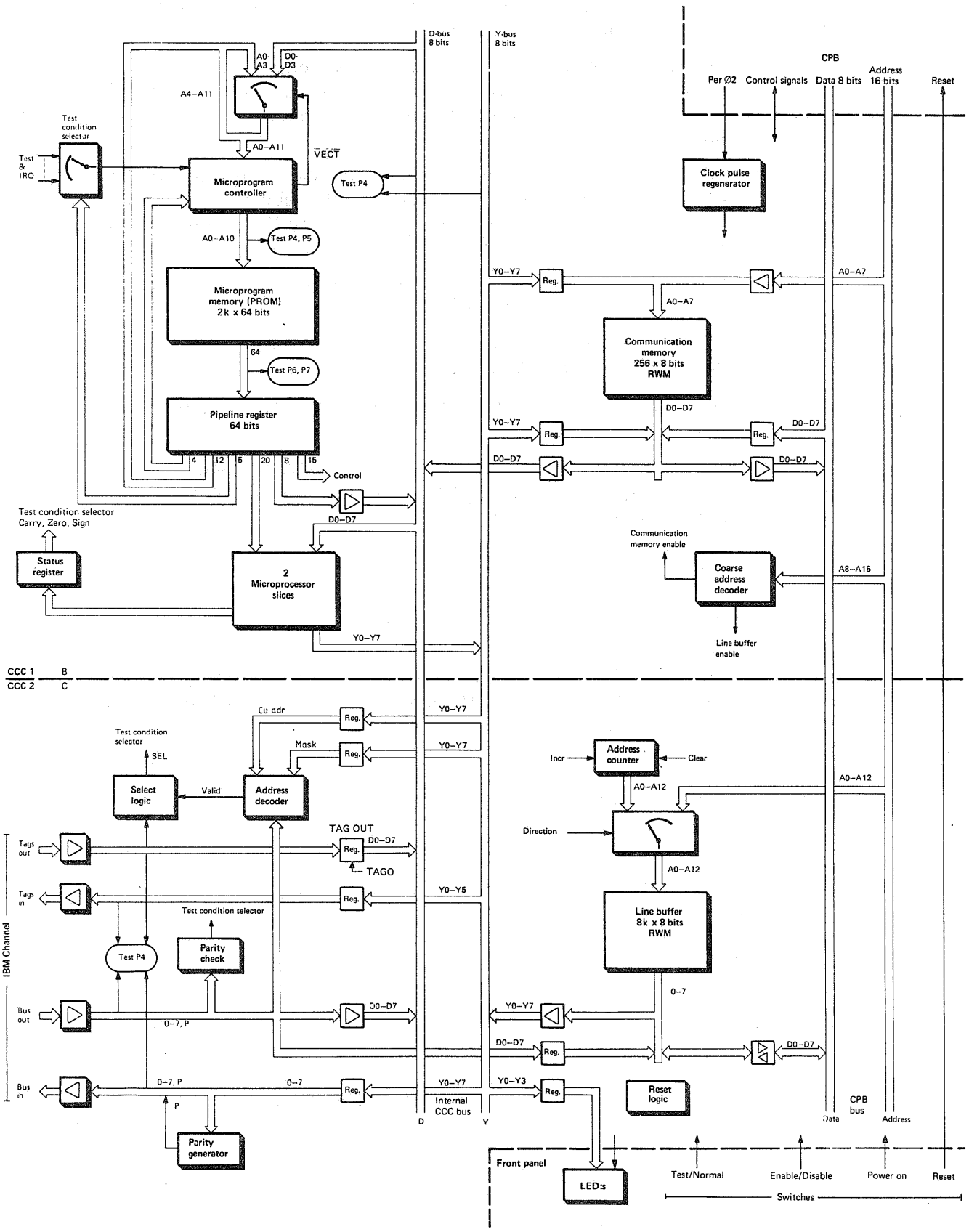


Fig. 31. Channel communication controller, block diagram

- Two 4-bit microprocessor slices containing ALU functions for manipulation of data on the CCC bus.
- Communication memory, addressed from either of the two bus systems, storing status information from displays, printers, flexible disk units, and other information used to establish and maintain communication.
- Coarse address decoder, sensing the eight most significant bits of the CPB address bus. This decoder delivers an enable signal to the line buffer control circuits if it is addressed, or to the communication memory control circuits if it is addressed.
- Clock pulse regenerator, synchronizing the CCC clock to the CPB clock by means of a phase-locked loop.
- A test connector (P4) for connection of a recorder for analysis of activity on internal address and data buses.
- A test connector (P5) for external addressing of the microprogram memory.
- Test connectors (P6, P7) for connection of external PROM storage. The internal microprogram PROM chips are automatically disabled when external PROMs are connected.

CIB includes:

- Relays to provide automatic propagation of Select out at disabled state or power off.
- High/Low priority switch, determining if Select out shall be caught on its way out from the channel (high priority) or on its way back from the terminator (low priority).

CPL panel controls and indicators

The CPL panel (see Fig. 1) holds the following controls:

- Power on switch (mains voltage switch)
- Enable/Disable switch
- Normal/Test switch
- Reset switch (reset function via CPB)

All indicator lamps are lit at power on. When the lamps go off they indicate:

- Lamp 1: CCC program is started
- Lamp 2: Result of internal CCC processor test is "Go"
- Lamp 3: Result of internal test of status memory is "Go"
- Lamp 4: Result of CCC-CPB communication test is "Go"

The Enable lamp is controlled by the CCC via relays on the CIB. The power on switch contains a built-in indicator lamp.

CPB indicators

Two of the five indicator lamps on the CPB front are of interest in CPL 4102:

- Lamp 1: Blinks when CPL has started with an internal poll to find the system disk. Glows steadily when the CPL has the operating system loaded into its memory
- Lamps 2 – 4: Not used
- Lamp 5: Lit directly by +5 V from the power supply

CCC Basic functions

The main function of the CCC is to serve as a buffer between the high-speed channel and the devices connected to the CPL unit. Fig. 32 provides a summary of the CCC function.

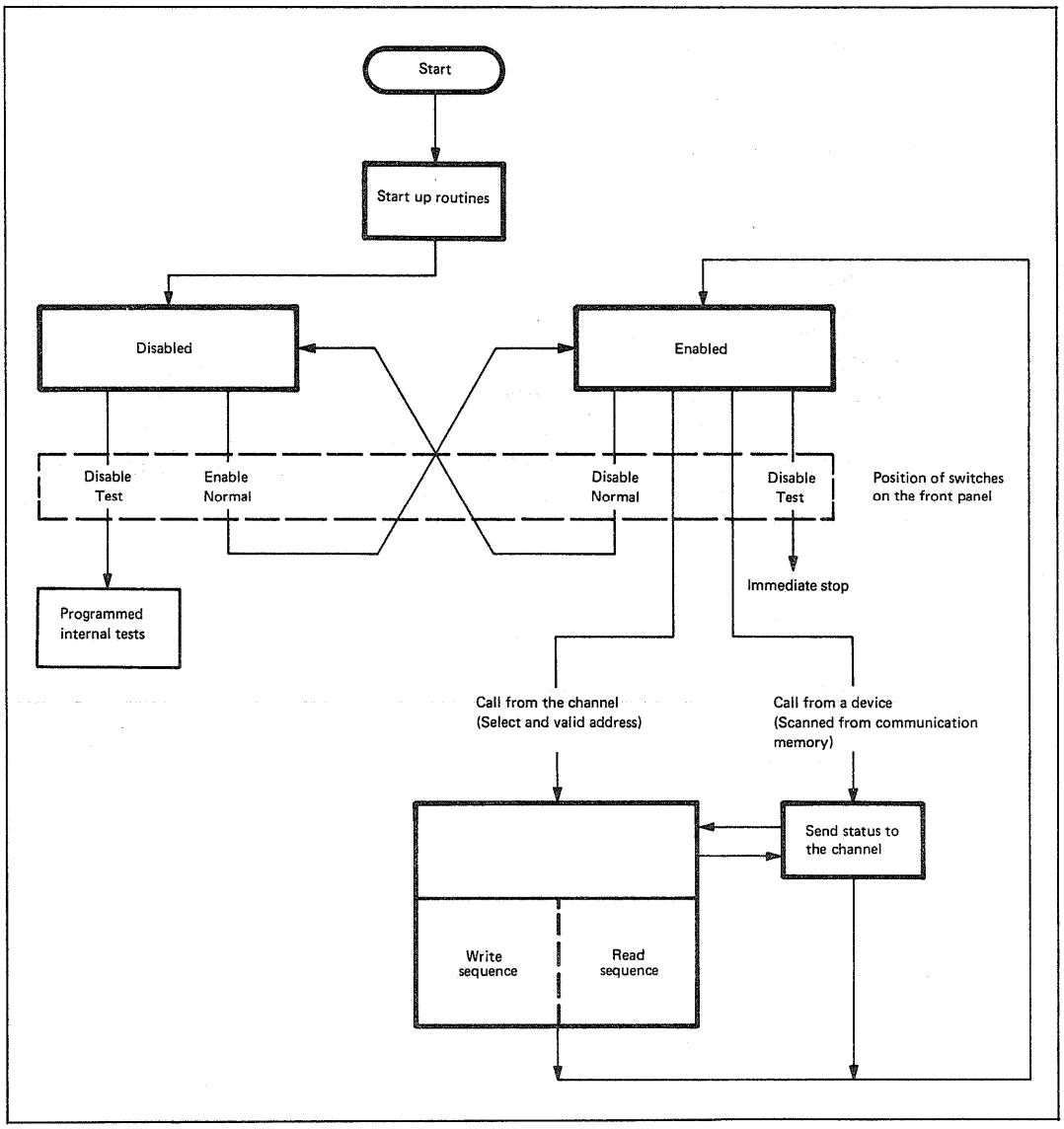


Fig. 32. CCC basic function

Start-up routines

Immediately after power on to the CPL, the CCC starts to perform internal tests such as processor tests and RWM tests. During this time the CPB has read in the system program from the system disk, and is ready to interface the CCC. The CCC then tests the communication to the CPB, via interrupts and the communication memory.

When all tests have been carried out correctly (all lamps on the front panel are switched off), initiation of the CCC starts. This means that the specific address of the CPL unit is loaded into an address decoder, so that the channel can address the CPL unit. Moreover all commands and status bytes to and from the devices (display units and printers connected to the CPL) are reset.

The CCC is then ready to start operation and enters either the enabled or disabled state, depending on the switch on the front panel. In the disabled state, programmed internal tests can be carried out (Normal/Test switch on the front panel in position test). Normally the CCC is in the enabled state.

In the enabled state the CCC senses the following:

- Call from the channel
- Call from a device
- Enable/Disable switch

Call from the channel

If the Select in signal and a valid address appears on the channel, the CCC immediately catches the select signal and starts communication. If a Write command is detected (data from the channel to a device), the CCC stores the command to the device in the communication memory and the data in the line buffer memory. At communication end an interrupt is sent to the CPB, telling that data has been received from the channel to a certain device.

The CCC then enters a wait state (a sort of enabled state), while the CPB transfers command and data from the CCC memory to the selected device.

The CPB then returns with an acknowledge to the CCC, telling that data has been received correctly. The CCC then calls the channel and transfers the acknowledge to the computer. Thereby a write sequence is ended and the CCC returns to the enabled state.

Call from a device

If a device wants to call the channel, the CPB stores a status in the communication memory, which is scanned by the CCC in the enabled state. When the status is detected, the CCC switches over the line buffer memory to the CPB, so that data to the channel can be loaded in the memory. Moreover the CCC sends a request to the channel, telling that the CPL has got a message for the channel. The CCC then enters a wait state (enabled state), waiting for sending the message.

Then when the channel is calling back for a read sequence, the CCC sends the data stored in the line buffer memory. At end the CCC tells the CPB that the data is sent, by changing status in the communication memory. Thereby a read sequence is ended and the CCC returns to the enabled state.

Enable/Disable switch

In the enabled state the CCC also senses the Enable/Disable switch on the front panel. If the switch is set in position disable the CCC enters the disabled state as soon as the program permits.

CCC Interfaces

The interface to the channel is described under "Principles of local connection".

The interface to the CPB contains the following signals:

Direction CPB–CCC	Abbreviation	Description
⟹	A-bus (A0–A15)	Address bus
⟺	D-bus (D0–D7)	Data bus
→	Per Ø2	Clock pulse
→	R/W	Read/Write signal
→	VMA	Valid memory address
←	IRQ CCC (IRQ3)	Interrupt request
→	IRQ CPB (Int. slave 1)	Interrupt request
→	COL	Column (address valid)
→	NMI	Non-maskable interrupt (reset function)

CCC Processor

The CCC processor handles the transfer of messages between the IBM channel and the CCC line buffer. This includes evaluation of tag signals from the channel, routing of data, and control signals and generation of tag signals to the channel. The processor consists of the following circuits (see Fig. 33):

- Microprogram controller
- Microprogram memory 2k × 64 bits (PROM)
- Pipeline register 64 bits, for latching of instructions
- Two 4-bit bipolar microprocessor slices
- Selectors for jump selection.

The processor uses two 8-bit buses, a D-bus and a Y-bus. The D-bus is used for routing of data into the microprocessor, whereas the Y-bus is used for data from the microprocessor.

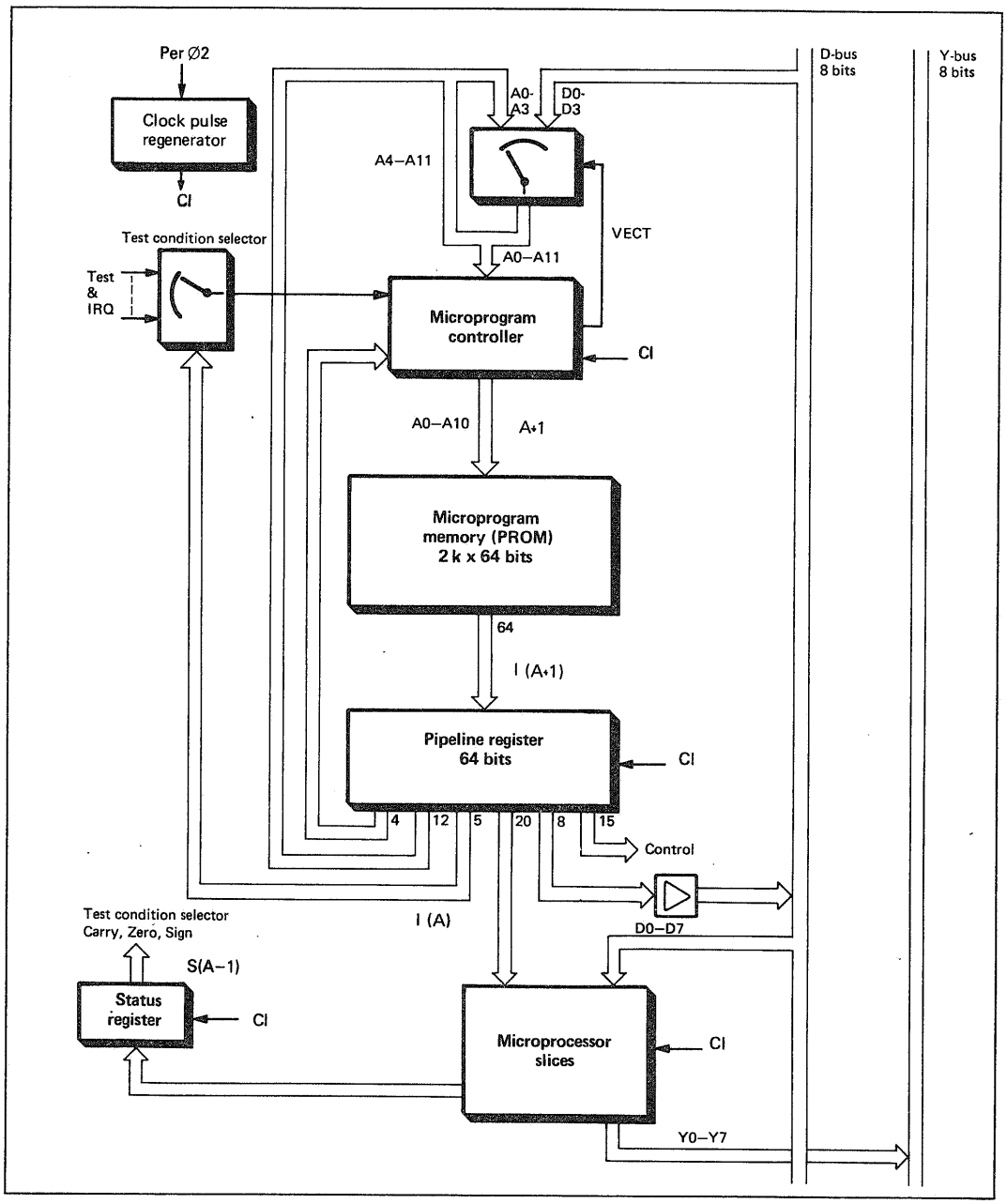


Fig. 33. CCC processor

The two cascaded microprocessor slices form an 8-bit high-speed processor array containing an 8-function ALU, 16 addressable registers, and various other circuits. The processor array is mainly used for logical manipulations, data testing, etc.

The system clock is generated by a circuit which is phase-locked to the CPB clock. The frequency is four times the reference frequency Per Ø2.

Microinstructions

Each of the microinstructions stored in the microprogram memory is 64 bits wide. The memory has room for 2048 such microinstructions. They are clocked out into the 64-bit pipeline register by the system clock. A summary of the microinstruction interpretation is found in Appendix 1.

The out lines of the pipeline register are connected as follows:

- 4 lines are used to instruct the microprogram controller how to determine the address of the next microinstruction. There are 16 such instructions. The simplest of these instructions is Continue, which means that the next sequential microinstruction is to follow. The other instructions often result in a jump from one place in the microprogram to another.
- 12 lines are used to provide the immediate address required in conjunction with some of the jump instructions.
- 5 lines are used to select the source and polarity of the test signal which is used by the conditional instructions
- 20 lines are connected to the two microprocessor slices. These lines include instructions concerning operation of the arithmetic logic units (ALU) in the slices.
- 8 lines are directly connected to the internal D-bus. These lines carry immediate data.
- 15 lines (a 16th line is included in the 20 lines to the processor above) are carrying control information. This includes:
  - six directly usable control signals such as read/write and enable signals for the line buffer and the communication memory, change signal for line buffer access, and a direct Service in signal to the channel
  - three groups of three signals, each group connected to a one-of-eight decoder. The decoded output signals from the first group determine the source of data to be applied to the D-bus. Decoded signals from the second group determine the destination of the data on the Y-bus. The third group provides various other control signals.

Processor architecture

For maximum speed, the CCC processor uses a so called one-level pipeline based architecture. This means that the microprogram memory and the microprocessor slices are in parallel speed paths.

At any time during microprogram execution, three microinstructions must be considered: the current one at address A, the previous one at address A−1, and the next one at address A+1. Each system clock pulse:

- Loads the result, S(A−1), of the previous microinstruction into the status register
- Loads the current microinstruction, I(A) into the pipeline register.

Depending on the microprogram controller instruction (4 bits) included in I(A), the address A+1 of the next microinstruction will be generated. It can be:

- The immediate address included in the current microinstruction
- An address provided by the microprogram controller as the result of previous activities (e.g. from an internal stack or microprogram counter).

The address generation is depending on the microprogram controller instruction, on five other bits of the current microcinstruction selecting test condition, and on test results from various CCC circuits, including S(A−1), stored in the status register. Four of the 16 microprogram controller instructions are unconditional, i.e. independent of test conditions.

During the time after one clock pulse and before the next one, the following sequential events occur:

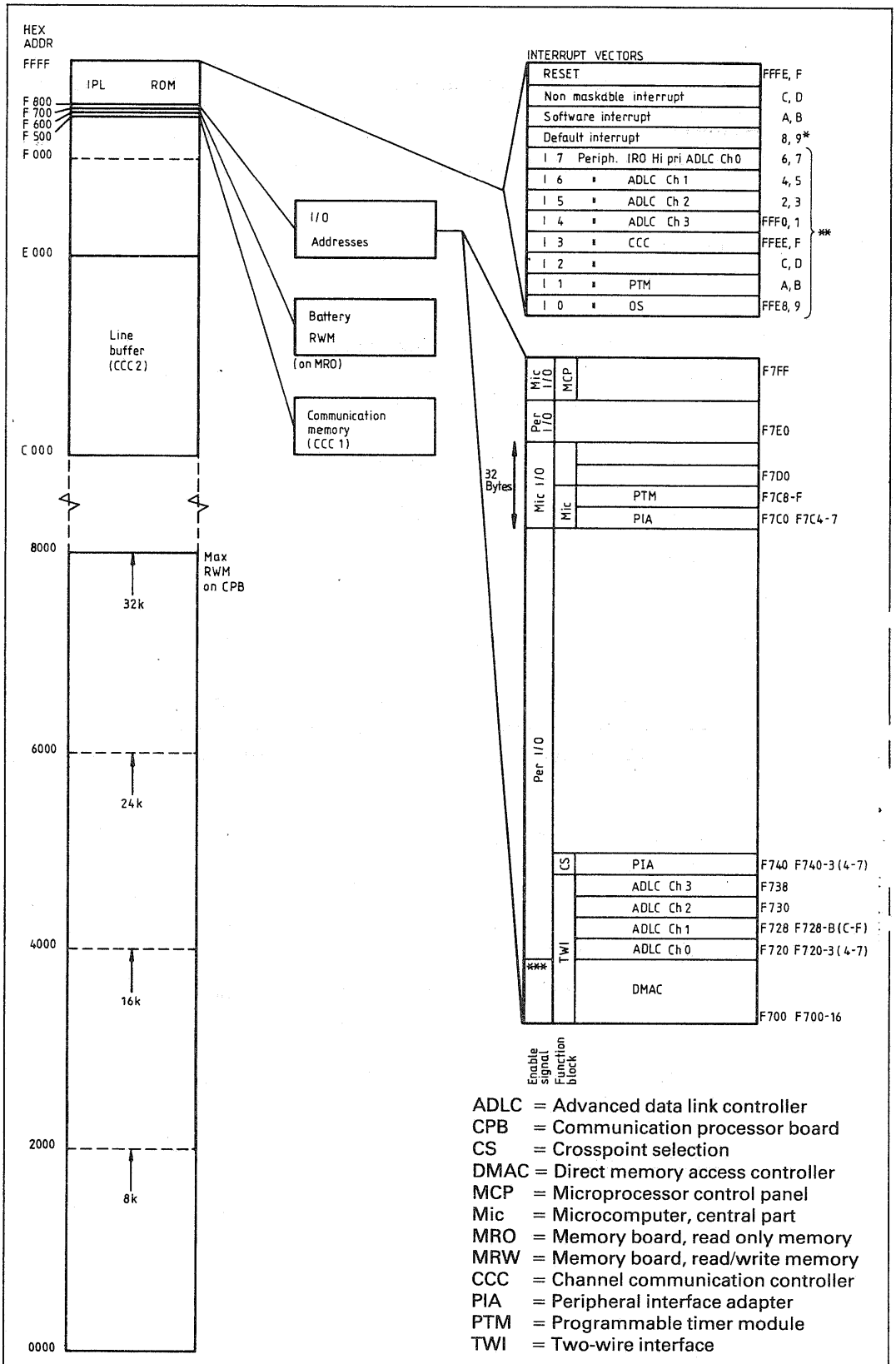
- I(A) becomes available at the pipeline register output. Thus a new instruction is immediately applied to the microprogram controller. At the same time, a test select code is applied to the test condition selector.
- After a delay, the test input selected is applied to the microprogram controller.
- The microprogram controller processes the instruction and test result, and generates an address A+1 which is applied to the microprogram memory.
- After the PROM access time, a new microinstruction is available at the memory output, and applied to the pipeline register input.

Memory Map

Fig. 34 presents the memory map of CPL 4102. (Do not mix up with the microprogram memory for the CCC processor on the CCC 1 board.)

Communication to/from the host computer takes place via a line buffer on the CCC 2. This line buffer occupies 8k bytes. Note that the corresponding area in the basic memory (RWM) on the CPB must be blocked by strapping on the CPB.

Information for communication control, such as status for the connected devices, is stored in a special communication memory on the CCC 1. This storage occupies 256 bytes.



* MPU reading FFF8 – 9 will make contents of two of the cells FFE8 – F9, depending on highest priority valid interrupt, transferred into the MPU.

** A read or, preferably, write operation into one of the cells FFE8 – FFF7 will result in a setting of a corresponding mask value. As an example, FFEE or FFEF **from MPU** will disable interrupts of priority < 3.

*** Not present during one of the three DMA cycles DMA T1 – T3.

Fig. 34. CPL memory map

Detailed Description

Detailed information on the microprocessor slice (2901A) and the micro-program controller (2910) can also be found in National Semiconductor: IDM 2900 family microprocessor data book or Advanced Micro Devices: Bipolar microprocessor logic and interface data book.

Microprocessor Slice

General

The four-bit bipolar microprocessor slice for high-speed operations can be cascaded to any desired word length. In the CCC application, two slices are put together to form a word length of eight bits.

The microprocessor slice (Fig. 35) consists of:

- A 16-word 4-bit two-port RAM
- A high-speed ALU
- A Q register
- Circuits for shifting, decoding and multiplexing.

The 9-bit microinstruction word is organized into three groups of three bits each. The groups select ALU source operands, ALU function, and ALU destination register. Three flag outputs are used: Zero, Carry and Sign.

The three key elements in the block diagram are the 16-word RAM, the Q register and the ALU.

Random Access Memory (RAM)

Data in any of the 16 words of the RAM can be read from the A port of the RAM as controlled by the 4-bit A address field input.

If the same address is applied to the B address field, the same data will be available at the B port. Different addresses at A and B address inputs will yield different data at the two output ports, i.e. simultaneous access to two RAM words.

When new data is to be written into the RAM, the B address input must be used. The RAM data input is driven by a 3-input multiplexer. This configuration is used to shift the ALU output data (F) if desired. the RAM input data always comes from the ALU.

The RAM has two shift inputs/outputs which are intended for cascaded operation.

A summary of the use of the RAM addresses can be found in Appendix 2.

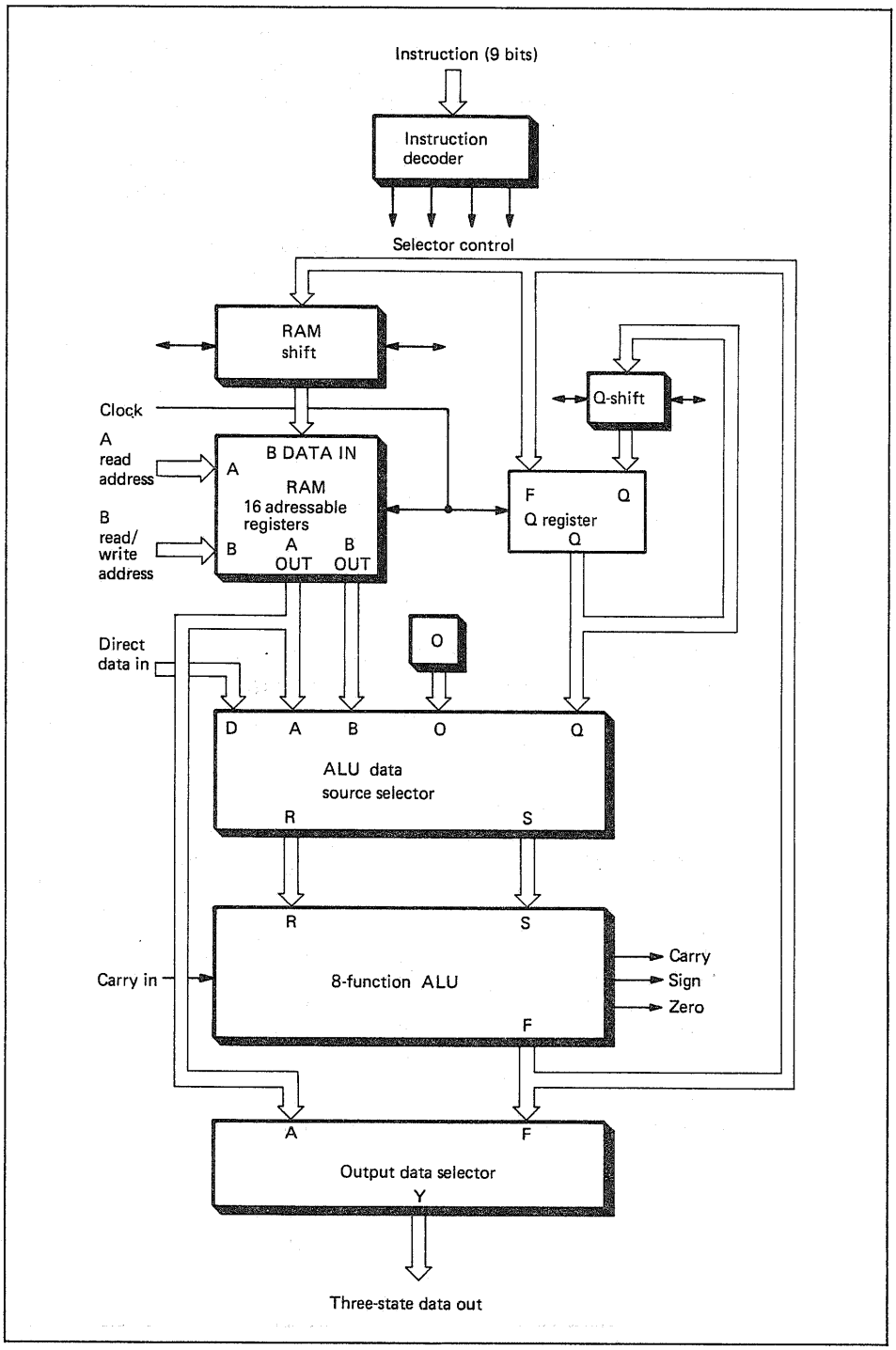


Fig. 35. Microprocessor slice

Q register

The Q register is a separate 4-bit register intended for arithmetic operations and for use as an accumulator or holding register. Its input field is driven by a 3-input multiplexer, which allows direct up/down shift operations.

For cascaded operation, the Q register has two shift inputs/outputs simular to those of the RAM.

Arithmetic Logic Unit (ALU)

The ALU has two input fields: R and S. The data sources for the R and S fields are determined by instruction bits I2 – I0 (ALU source). The following table shows the combinations available:

Micro code					ALU source operands	
Mnemonic	I2	I1	I0	Octal Code	R	S
AQ	L	L	L	0	A	Q
AB	L	L	H	1	A	B
ZQ	L	H	L	2	Z	Q
ZB	L	H	H	3	Z	B
ZA	H	L	L	4	Z	A
DA	H	L	H	5	D	A
DQ	H	H	L	6	D	Q
DZ	H	H	H	7	D	O

A = RAM A-port
B = RAM B-port
Q = Q register
D = Direct data input
Z = 0 (fixed logic zero)

The ALU function, i.e. the operation performed on the operands applied to R and S inputs, is determined by instruction bits I5 – I3 (ALU Function). The following table shows the eight functions (three binary arithmetic and five logic functions):

Micro code					ALU	
Mnemonic	I5	I4	I3	Octal Code	Function	Symbol
Add	L	L	L	0	R Plus S	R + S
Subr	L	L	H	1	S Minus R	S – R
Subs	L	H	L	2	R Minus S	R – S
Or	L	H	H	3	R or S	R ∨ S
And	H	L	L	4	R and S	R ∧ S
Notrs	H	L	H	5	$\overline{R}$ and S	$\overline{R}$ ∧ S
Exor	H	H	L	6	R ex-or S	R ⊻ S
Exnor	H	H	H	7	R ex-nor S	$\overline{R \veebar S}$

∨ = Boolean inclusive or
⊻ = Boolean exclusive or
∧ = Boolean and
$\overline{R}$ = Complement of R

The ALU data output (F) can be routed to several destinations. Eight combinations are possible, as defined by instruction bits I8 – I6:

Micro code					RAM function		Q-REG function		Y output
Mnemonic	I8	I7	I6	Octal code	Shift	Load	Shift	Load	
Qreg	L	L	L	0	X	None	None	F→Q	F
Nop	L	L	H	1	X	None	X	None	F
RAMA	L	H	L	2	None	F→B	X	None	A
RAMF	L	H	H	3	None	F→B	X	None	F
RAMQD	H	L	L	4	Down	F/2→B	Down	Q/2→Q	F
RAMD	H	L	H	5	Down	F/2→B	X	None	F
RAMQU	H	H	L	6	Up	2F→B	Up	2Q→Q	F
RAMU	H	H	H	7	Up	2F→B	X	None	F

X = Don't care. Electrically, the shift pin is a TTL input internally connected to a three-state output which is in the high-impedance state.

B = Register addressed by B inputs.

Up is towards MSB, Down is toward LSB.

For example, Qreg results in two things:

- The ALU output (F) is loaded into the Q register on the next high-to-low clock transition.
- The ALU output (F) is available on the three-state data output port (Y).

RAMQD results in several events:

- The ALU output (F), shifted one step down (= F/2), is loaded into the RAM position addressed by the B address field on the next clock pulse.
- The Q register output (Q), shifted one step down (= Q/2), is loaded into the Q register on the next clock pulse.
- The ALU output (F) is available on the three-state data output port (Y).

It should be noted that carry in modifies the arithmetic functions but not the logic functions.

Three status flags from the ALU are used in the CCC application: Carry, Sign, and Zero. The Sign flag is the most significant ALU output (bit F3), which can be used to determine positive or negative results without enabling the three-state output.

Cascaded operation

In CCC two microprocessor slices operate in cascade. The result is that 8-bit data words can be processed. Fig. 36 shows how the slices are cascaded. Clock, address, and instructions are applied in parallel, whereas Carry, RAM-shift, and Q-shift terminals are connected in series. The 4-bit data in/out fields are taken together to form 8-bit data fields.

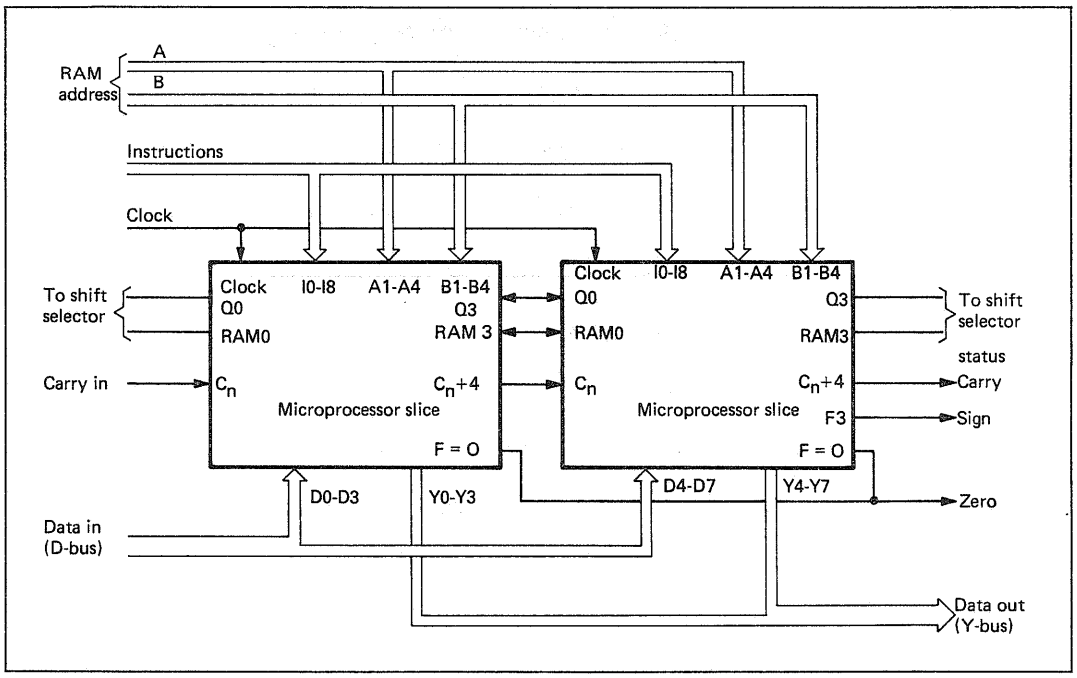


Fig. 36. Cascaded microprocessor slices

The RAM and Q-register shift circuit is shown in Fig. 37. The shift selectors connected to the outer ends of the array determine the type of shift operation to occur. Depending on the S0 – S1 shift function code in conjunction with instruction bit I7 a zero, a one, or an overflowing bit from the RAM or Q-register is entered into the "empty" bit position. Fig. 38 describes the shift possibilities available.

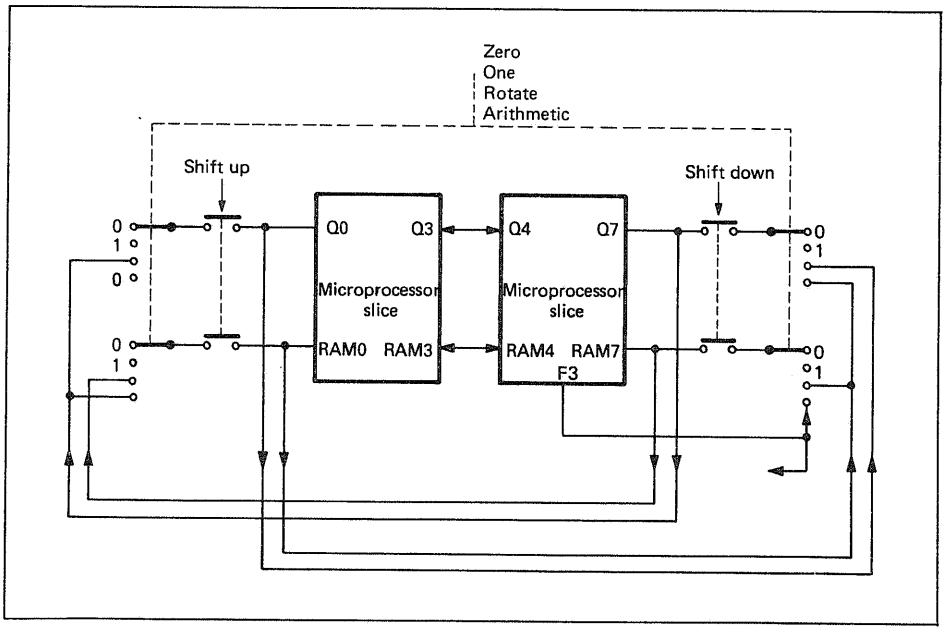


Fig. 37. Shift selectors

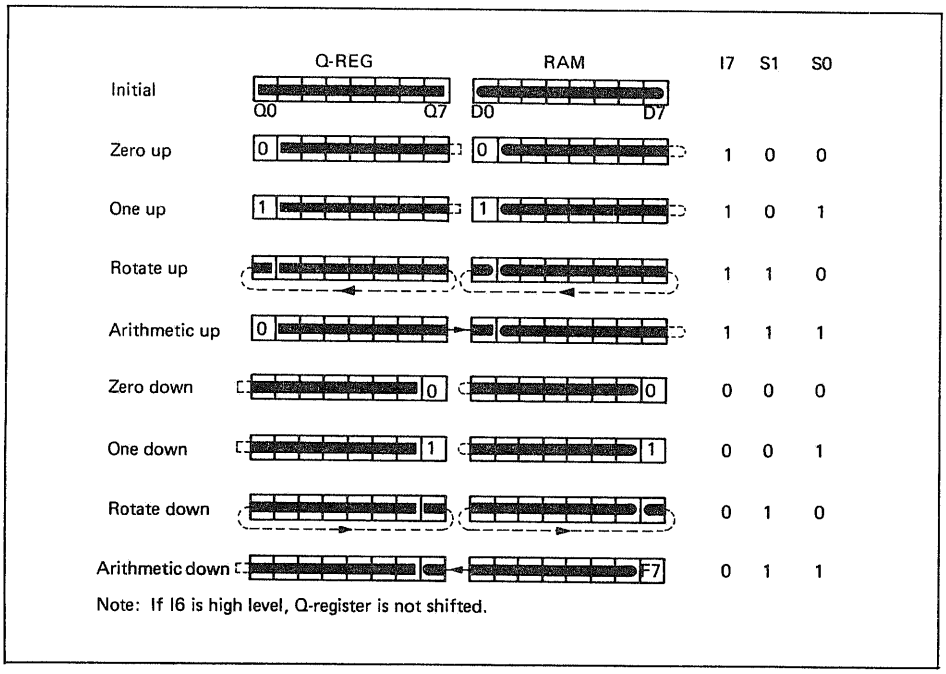


Fig. 38. Shift instructions

Microprogram controller

General

The microprogram controller is an address sequencer intended for controlling the sequence of execution of microinstructions stored in a microprogram memory. It provides for sequential access and conditional branching. The microprogram controller generates a 12-bit address.

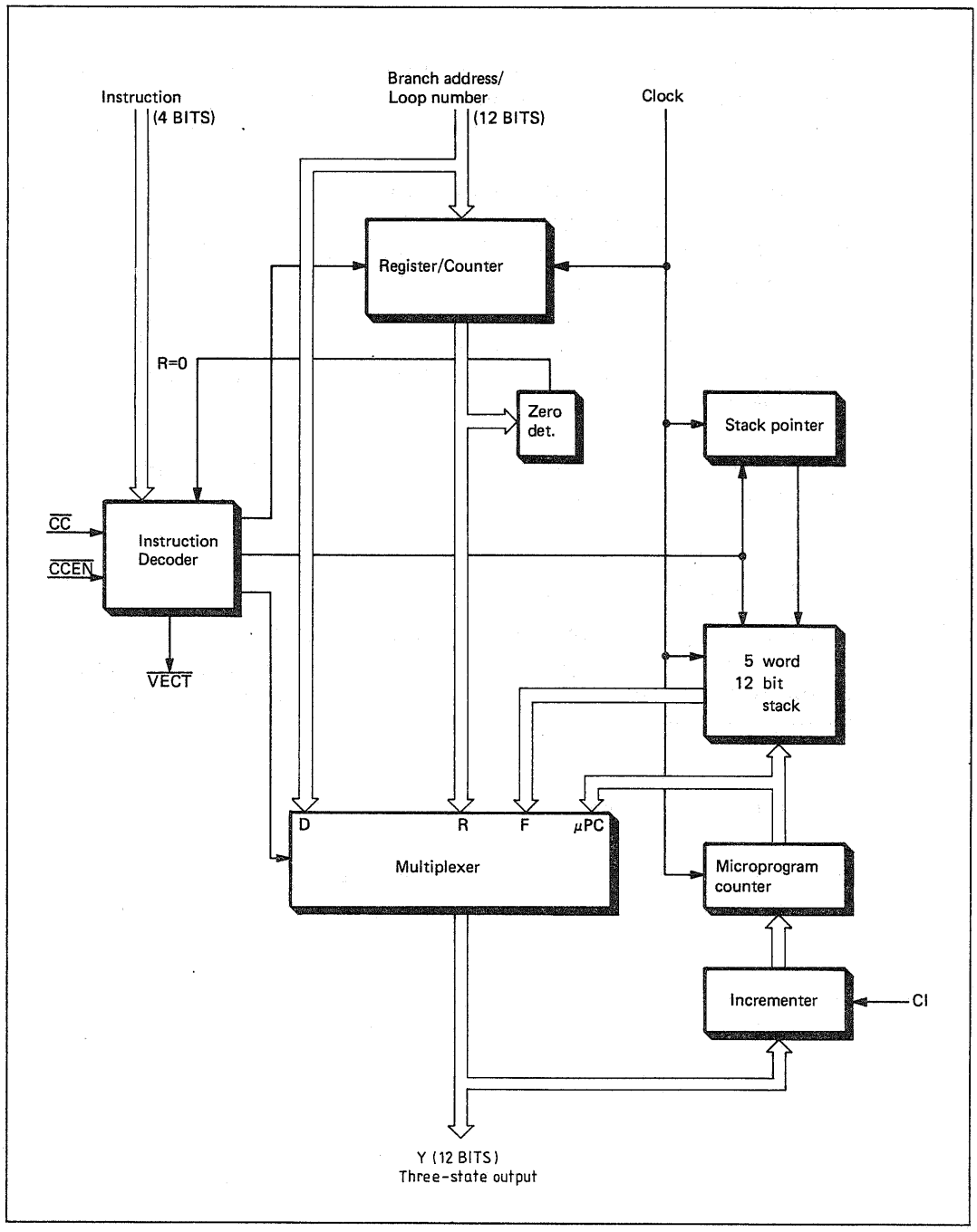


Fig. 39. Microprogram controller

The microprogram controller (Fig. 39) contains an instruction decoder, and a multiplexer which selects the output address from one of the following four sources:

- Directly from the branch address field of the pipeline register (see CCC processor)
- Register/Counter
- Five-level stack
- Microprogram counter

The operation is governed by 16 instructions, most of which are conditional with regard to external test conditions, state of register/counter (zero or not) or both.

When carry in (CI) to the incrementer is high, the next clock puls loads the current output address plus one into the microprogram counter. When CI is low, the output address is loaded unaltered into the counter.

The microprogram counter is the source for sequential stepping in the microprogram. It also provides return addresses to be pushed into the 5-word stack. The stack is controlled by a stack pointer which always points to the last word written.

The register/counter is intended for loop counting. A preloaded loop number is decremented down to zero, which is used as a microinstruction test criterion.

The CC and CCEN inputs are used for testing of external conditions. See section Test condition logic.

Microprogram controller instructions

The following table provides a summary of the 16 instructions available. Note that actual push/pop operations in the stack, and loading/decrementing of the register counter occur on the next low-to-high clock transition following the current instruction.

Microprogram controller instructions

I3–I0	Mnemonic	Name and description
0	JZ	Jump zero Jump to address zero. The stack is cleared.
1	CJS	Cond jsb pl Jump to subroutine at the address provided by the pipeline register, if the test condition is true. A return address is pushed into the stack. If the test condition is false; the next address is supplied by the microprogram counter.
2	JMAP	Not used.
3	CJP	Cond jump pl Jump to the address provided by the pipeline register, if the test condition is true. If the test condition is false; the next address is supplied by the microprogram counter.

4	PUSH	Push/Cond ld cntr Push the next address into the stack. If the test condition is true; load the number from the pipeline register into the register/counter. Then the next instruction.
5	JSRP	Cond jsb r/pl If the test condition is true; jump to subroutine at the address provided by the pipeline register. If false; jump to subroutine at address provided by the register/counter. In both cases, the return address is pushed into the stack.
6	CJV	Cond jump vector If the test condition is true; jump to the address provided by the pipeline register (8 most significant bits) and the D-bus (4 least significant bits). If false; the next address is supplied by the microprogram counter.
7	JRP	Cond jump r/pl If the test condition is true; jump to the address provided by the pipeline register. If false; jump to the address provided by the register/counter.
8	RFCT	Repeat loop, cntr=0 If the register/counter contains non-zero value; go to the address provided by the stack, and decrement the register/counter. If zero; go to the next address, supplied by the microprogram counter, and pop the stack.
9	RPCT	Repeat pl, cntr=0 If the register/counter contains non-zero value; go to the address provided by the pipeline register, and decrement the register/counter. If zero; go to the next address, supplied by the microprogram counter.
A	CRTN	Cond rtn If the test condition is true; jump back to the return address, supplied by the stack, and pop the stack. If false; go to the next address, supplied by the microprogram counter.
B	CJPP	Cond jump pl & pop If the test condition is true; jump to the address provided by the pipeline register, and pop the stack. If false; the next address is supplied by the microprogram counter.
C	LDCT	Ld cntr & continue Load the contents of the pipeline register into the register/counter. Then the next instruction.

D	LOOP	Test end loop If the test condition is true; go to the address provided by the microprogram counter, and pop the stack. If false; go to the address provided by the stack.
E	CONT	Continue Continue to the next address, provided by the microprogram counter.
F	TWB	Three-way branch If the test condition is true; go to the next address provided by the microprogram counter, and pop the stack. If false, and the register/counter contains a non-zero value; go to the address provided by the stack. If false, and the register/counter contains zero; go to the address provided by the pipeline register, and pop the stack.

Test condition logic

The microprogram controller uses two test condition inputs, CC and CCEN for conditional branching. Five bits in the microinstruction, Cond 3-0 and Condpol, determine the parameter to be tested, and the signal polarity desired for a pass decision. Fig. 40 describes the function of the test condition logic.

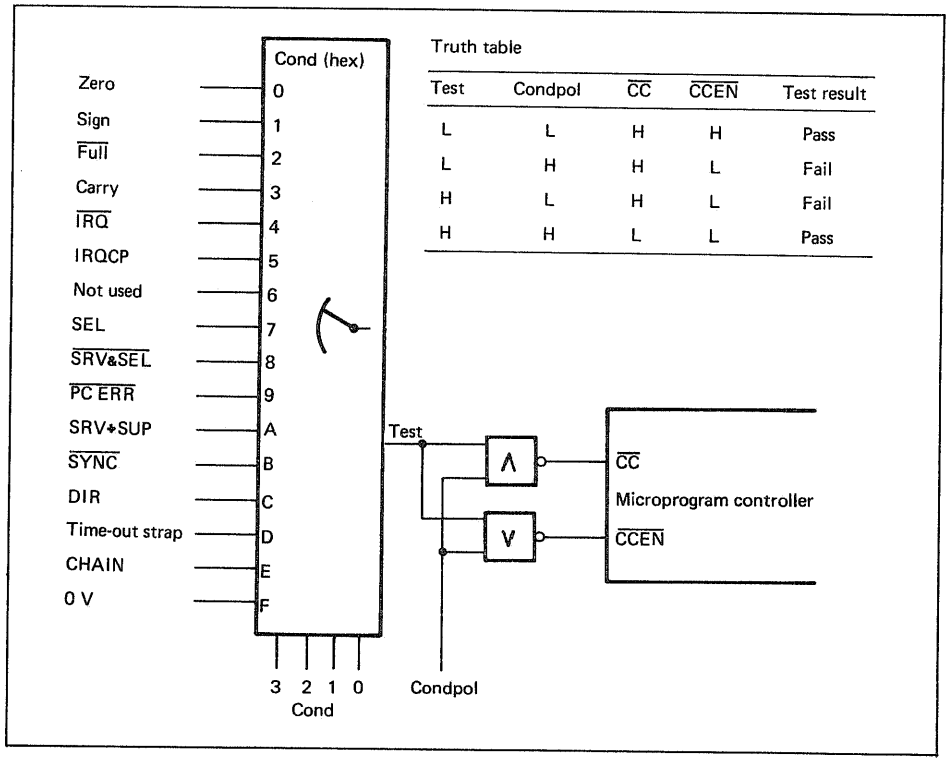


Fig. 40. Test condition logic

The various test signals are described in the following table:

Cond 3–0	Signal	Description
0	Zero	The ALU outputs contain all zeros.
1	Sign	The most significant ALU bit is 1.
2	Full	The microprogram controller stack is full.
3	Carry	Carry out from the ALU.
4	IRQ	IRQ from the CPB or SEL from the channel or DIS from the front panel Enable/Disable switch.
5	IRQ CP	IRQ from the CPB.
6	Not used	
7	SEL	Selected from the channel.
8	SRV + SEL	Service out or Select out from the channel.
9	PC ERR	The parity check has detected an error.
A	SRV & SUP	Service out and Suppress out from the channel.
B	SYNC	Special pulse related to the system clock (see Clock pulse circuits).
C	DIR	Direction signal which determines if the CCC or the CPB has access to the line buffer (High level means that the CCC has access).
D	Time-out strap	Normally +5 V
E	CHAIN	The channel has ordered command chaining.
F	0 V	Used for unconditional branching.

Clock Pulse Circuits

The control pulses required for the CCC operation are derived from the CPB system clock Per Ø2. Since the CCC memories (line buffer and communication memory) can be accessed from the CPB as well as from the CCC, clock and control signals must be synchronized.

The clock pulse circuits on the CCC 1 are based on a phase-locked loop. Per Ø2 (approximately 1 MHz) is used as a reference frequency. A voltage controlled oscillator operates at a frequency that is eight times higher then Per Ø2. This 8 MHz clock signal (2Cl) is divided in three flip-flops down to the Per Ø2 frequency, and applied to the phase detector for comparison.

The flip-flops along with additional circuitry generate a few other clock signals. Fig. 41 shows the relationship between these signals.

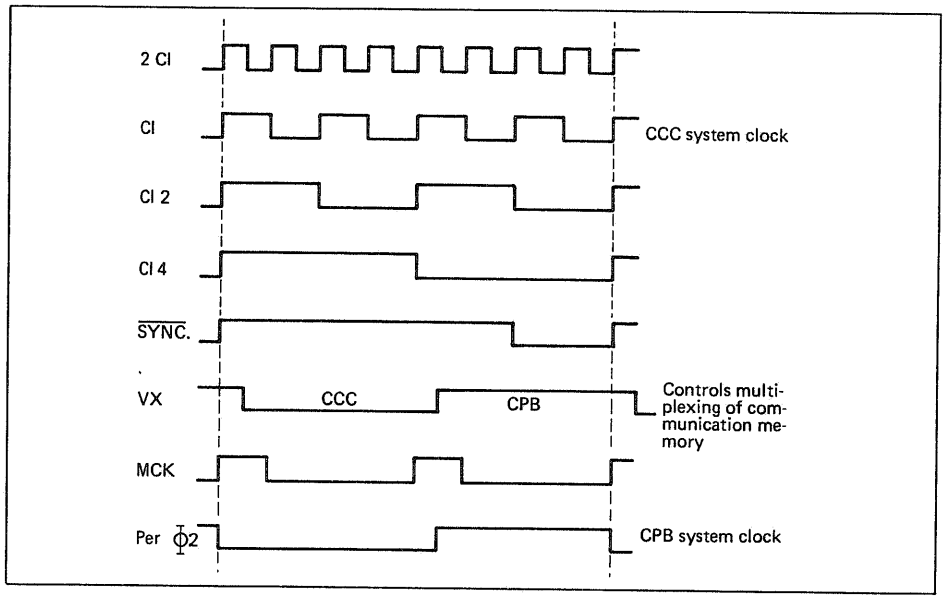


Fig. 41. Clock signals

Microprogram Memory

The microprogram memory consists of 16 1k × 8-bit bipolar PROM chips. It is possible to connect an external memory to replace the internal memory, via connectors P6 and P7 on CCC 1. The internal PROM is automatically disconnected, since a PROM enable input is tied to ground when the external PROM is connected.

The 64-bit instructions obtained from the microprogram memory are loaded into the pipeline register by the CCC system clock CI. A register, latching the 4-bit instruction to the microprogram controller, can be reset separately. The reset signal comes from the channel interface logic (CCC 2), as the result of reset or system reset. A three-state register connected to the D-bus latches the immediate data byte. A summary of the microinstruction interpretation is found in Appendix 1.

Line Buffer

Line buffer design

The line buffer is an 8k bytes read/write memory using 16 1k × 4-bit static RAM chips. Its purpose is to provide intermediate storage of data to and from the channel. The line buffer can be accessed by the CPB as well as by the CCC. Fig. 42 shows the basic configuration.

The access is controlled by the signal DIR, which determines the selection of address, enable, and write enable source: CCC or CPB.

When the CCC has control, the line store is addressed by the address counter. A read or write sequence starts with a clear (CLR) signal to the counter. CLR is decoded from the current microinstruction. Then, for each byte read or written, the microprogram generates increment (INC) pulses to the address counter.

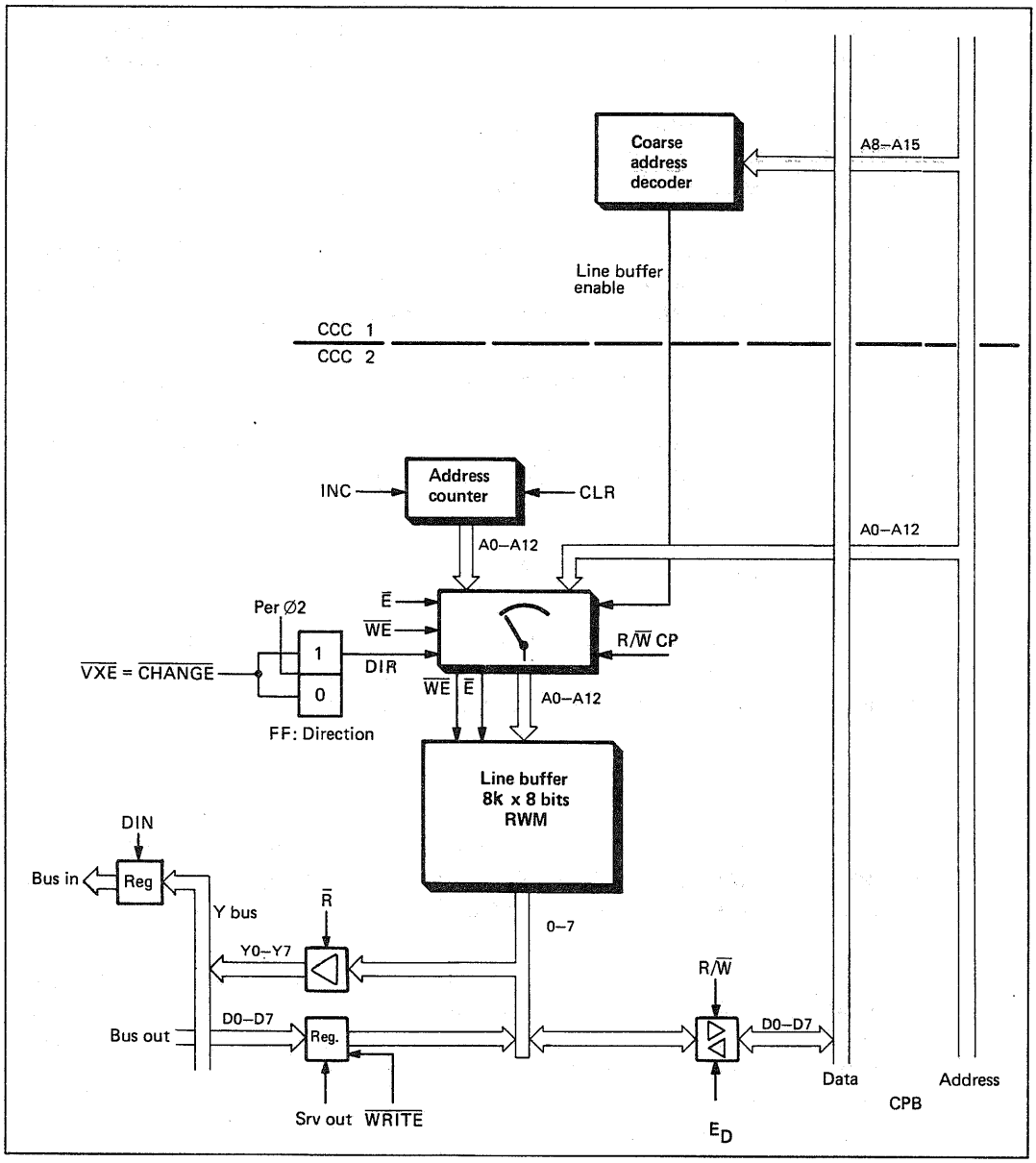


Fig. 42. Line buffer

In CCC mode, the enable signals E and WE are obtained from the current microinstruction. E is basically the logic OR of the READ and WRITE bits.

The write enable signal WE is derived from WRITE, but the timing is adjusted to provide some margin with respect to clock transitions. This allows data to settle prior to write. The timing of WE is shown in Fig. 43.

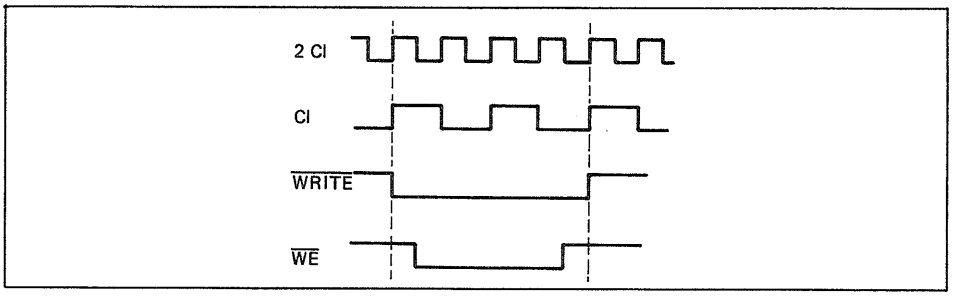


Fig. 43. WE timing

Data to the channel is routed to Bus in via the Y-bus. R from the microinstruction controls the three-state buffer to the Y-bus. DIN (Data in), decoded from the microinstruction, clocks the data into the Bus in latch.

Data from the channel is clocked from Bus out into a latch by Srv out (a tag out signal). Write from the microinstruction enables the data from the latch to be applied to the data I/O ports of the line buffer.

When control of the line store is to be turned over to the CPB, the CHANGE bit in the microinstruction goes low. This allows the direction flip-flop to change state on the next CPB clock pulse.

In CPB mode, the line buffer occupies part of the system memory map (Fig. 34). The buffer is addressed by the CPB address bus, bits A0-A12. The enable signal is generated in the coarse address decoder. Here it is determined that the current address is in the line buffer.

A signal R/W (strobed by COL) (both from the CPB bus) is used as the write enable (WE) signal in CPB mode.

Data to/from the CPB data bus is routed via a bidirectional buffer controlled by R/W from the CPB. The buffer is enabled if the address is in the correct range, and the line buffer is in CPB mode.

Line buffer handling

The line buffer is normally set in direction CCC. When the CPB accesses the memory (transfer of data to/from a device) the line buffer functions as an ordinary RWM memory to the CPB.

Before the CCC can transfer data from the line buffer to the channel, the line buffer must be set in direction CCC (tested on the DIR signal) and the address counter must be reset. In the first instruction, data from the memory is transferred to the Bus in of the channel via a three state buffer (enabled by R from the microinstruction) and a register (clocked by DIN from the microinstruction).

In the next sequential instruction the signal Srv in to the channel is raised and the address counter is incremented (INC from microprogram). Srv in is raised directly by a signal Srv in from the pipeline register.

The counters of data length (Counth and Countl) in the microprocessor RAM registers is then decremented and tested for carry. (No carry if all data is sent).

Then the condition Sel out or Sup out raised is tested as a sign that the channel has received the data byte. The signals Sel out or Sup out in tags out are ored by hardware and fed to the test condition selector.

When the test condition is passed, Srv in (from the pipeline register) is immediately dropped and a new data byte from the line buffer is transferred to the Bus in of the channel.

This means that the minimum time for sending a data byte is equal to four microinstructions (clock periods), which is equal to a maximum transmission rate of 1 Mbyte/s.

Communication Memory

Communication memory design

The communication memory is a 256 bytes read/write memory using two 256 × 4-bit static RAM chips. Its purpose is to store system and device status, and other data governing the communication process. It can be accessed by the CPB as well as by the CCC. Fig. 44 shows the basic configuration.

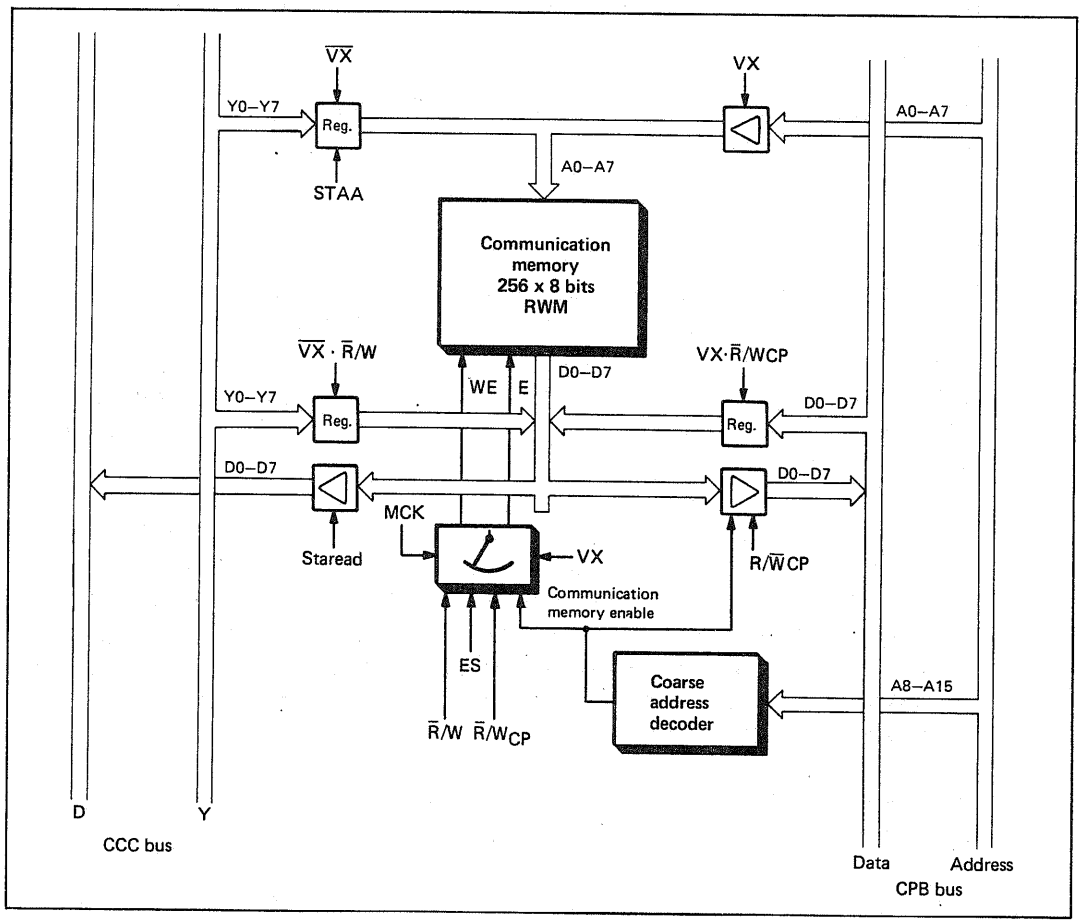


Fig. 44. Communication memory

Access to the memory is controlled by the signal VX. As can be seen in Fig. 41, VX is a clock signal with the same frequency as the CPB system clock Per Ø2. The CCC has access to the communication memory during the first half-cycle of VX, the CPB during the second half-cycle. This means that the CCC has only access during two of four CCC instruction cycles whereas the CPB has access during all CPB instruction cycles.

When the CCC has control, the address is taken from the Y-bus via a buffer register. The signal STAA (decoded from the microinstruction) loads the address into the register. VX enables the three-state output.

The enable signals E and WE are taken via a selector controlled by VX. In CCC mode, the E signal is derived from ES which is one bit in the microinstruction. Similarly, the WE signal is derived from the R/W bit in the microinstruction. For correct timing this signal is gated with MCK, to provide some margin for settling times. MCK is a special clock signal. See Fig. 41.

In CCC read mode the signal Staread, decoded from the microinstruction, enables the output buffer to the D-bus.

In CCC write mode the Y-bus data is loaded into a register which is enabled by VX and R/W (from the microinstruction).

In CPB mode, the communication memory occupies part of the system memory map (Fig. 34). The memory is addressed by the CPB address bus, bit A0 – A7. The enable signal E is generated in the coarse address decoder. Here it is determined that the current address is in the communication memory.

Write enable WE in CPB mode is derived from CPB R/W gated with MCK.

Communication memory handling

Since the communication memory (see Figs 31 and 44) is shared in time multiplex between the CCC and the CPB, the CCC must check if the memory can be acceessed. This is done by testing the SYNC signal in the test condition selector. The SYNC signal is generated by the clock pulse regenerator and indicates to the program that the CCC has access to the communication memory for two clock periods.

The microinstruction which tests SYNC is then repeated until SYNC becomes valid. The same instruction also clocks the memory address to a register between the Y-bus and the address lines of the communication memory by means of the control signal STAA. The address to the communication memory is often fetched from the microinstruction in the pipeline register (immediate data) and fed via the D-bus, through the processor to the Y-bus.

Then when SYNC becomes valid the microprogram controller outputs the address of the next microinstruction.

If a read sequence is executed, the three-state buffer of the communication memory, is enabled (by Staread decoded from the pipeline register) for two sequential microinstructions (set up time for the memory). Data is then fetched via the D-bus in the last of these two microinstructions. Data can either be stored in any register of the processor, or just pass through the processor simultaneously as the flags are affected (Zero, Carry etc).

The CPB can access the communication memory at any time, since the CCC clock frequency is four times higher than the CPB clock frequency, and the CPB only uses the two free CCC clock periods for access.

Channel Interface Logic

Select

The Select out/in signals from the channel, in contrast to all other bus/tag lines, are not connected directly to the line receiver on the CCC. Instead, they are connected via a priority switch to two relays. See Fig 45.

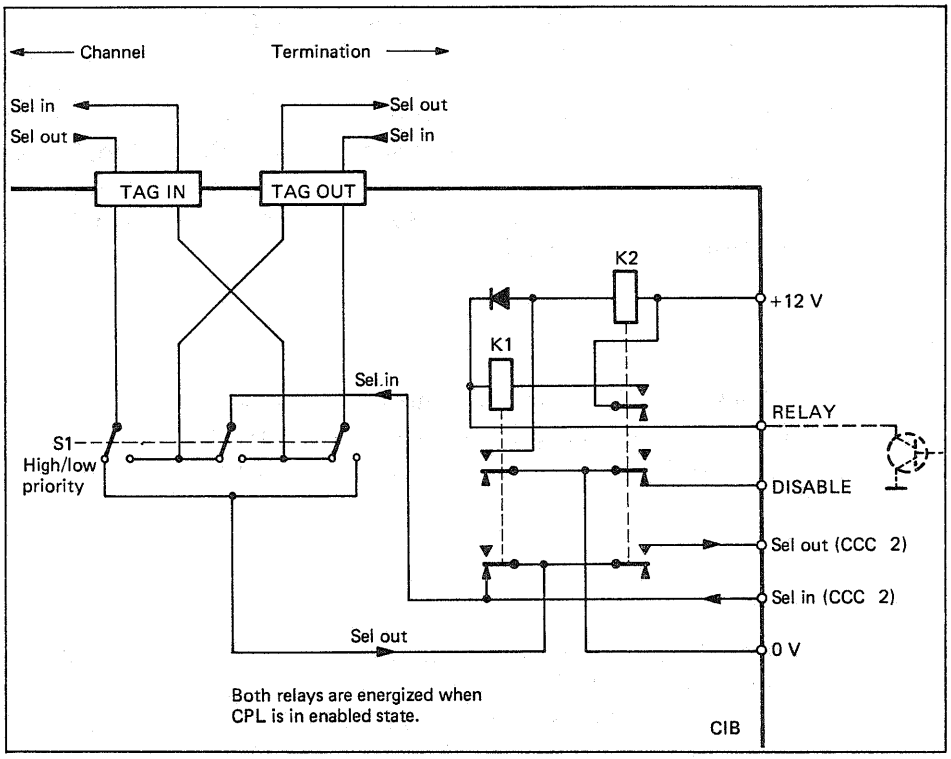


Fig. 45. Select circuits

When the CPL is switched off or in disabled state, both relays are off. This means that a Select out from the channel (or a Select in from the termination side, if low priority) is automatically propagated. In the enabled state, both relays are energized. This means that the select signal can be sent to the line receiver on the CCC. If the selection attempt proves unsuccessful, the CCC generates a Select in signal which is propagated "down the line" to the next control unit.

Select logic

The purpose of the select logic is to respond quickly to a selection attempt from the channel See Fig. 31. The circuits generate a SEL signal if the attempt is successful. Unsuccessful attempts result in a propagated select signal, sent out from the line driver.

Select out, Hold out and Address out together form the condition for a selection attempt. The address decoder determines if the address on Bus out is within the valid address range. This range is determined by the CPL system software which (in advance) has loaded a control unit address and a mask value into the two registers connected to the address decoder. The mask value depends on how many devices the cluster contains, and is used to determine the number of address bits (on Bus out) to be decoded. For example: For 16 devices the mask value is $F0_{(16)}$ and bits 7 – 4 are decoded.

The Select out and Hold out signals gated together are applied to a gate/flip-flop network where possible glitches are suppressed. Valid address and Select/Hold out together form the signal SEL. This signal is sensed by the microprogram via the test condition selector and indicates that the selection sequence can proceed.

Another situation when Select out occurs is as a response to Request in. The microprogram can generate two types of Request in signals; suppressible and non-suppressible. These signals are labelled Req 2 and Req 1, respectively. Request in to the channel is an or function of Req 1 and Req S (= Req 2 gated with Suppress out). The same or function also provides a condition for the signal SEL. In order to block cases, when the channel raises Select out when addressing another unit (i.e. not as a response to Request in), the signal Address out is used to block the generation of SEL in such cases.

Chaining

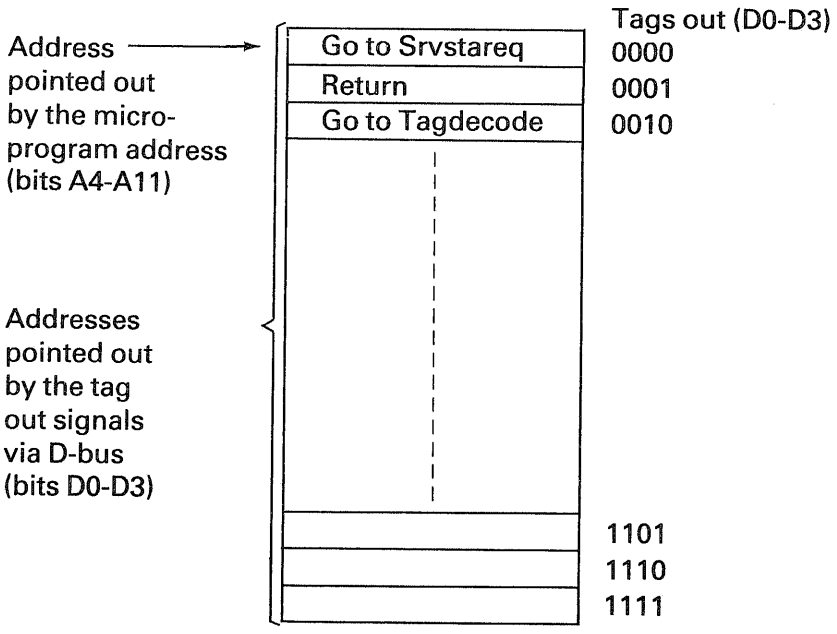
A signal CHAIN (to the test condition selector) is generated if Suppress out is up when Service out goes up as a response to status Channel end or Device end. Chaining stops as soon as Suppress out drops for minimum 250 ns.

Decoding of Tags in

To make the decoding of tags and commands as fast as possible, a special microprogram table is used. The microprogram instruction sends the start address of the table (A4 – A11) to the microprogram controller, (see Fig. 46) but also activates the VECT signal. The VECT signal switches over the address selector to the microprogram controller, so that the four lowest address bits to the microprogram controller are selected from the D-bus, bit 0 – 3. The same instruction also enables the tag out register by the TAG O signal, which is decoded from the pipeline register. The important tag out signals then become available on the D-bus.

This means that the start address of the 16 byte wide decode table (see below) is pointed at by the microprogram instruction, and one of the 16 instructions is pointed at directly by the additional four tag out bits. The microinstructions in the table can either order a jump to a certain instruction or return to the decode table again.

Tag decode table



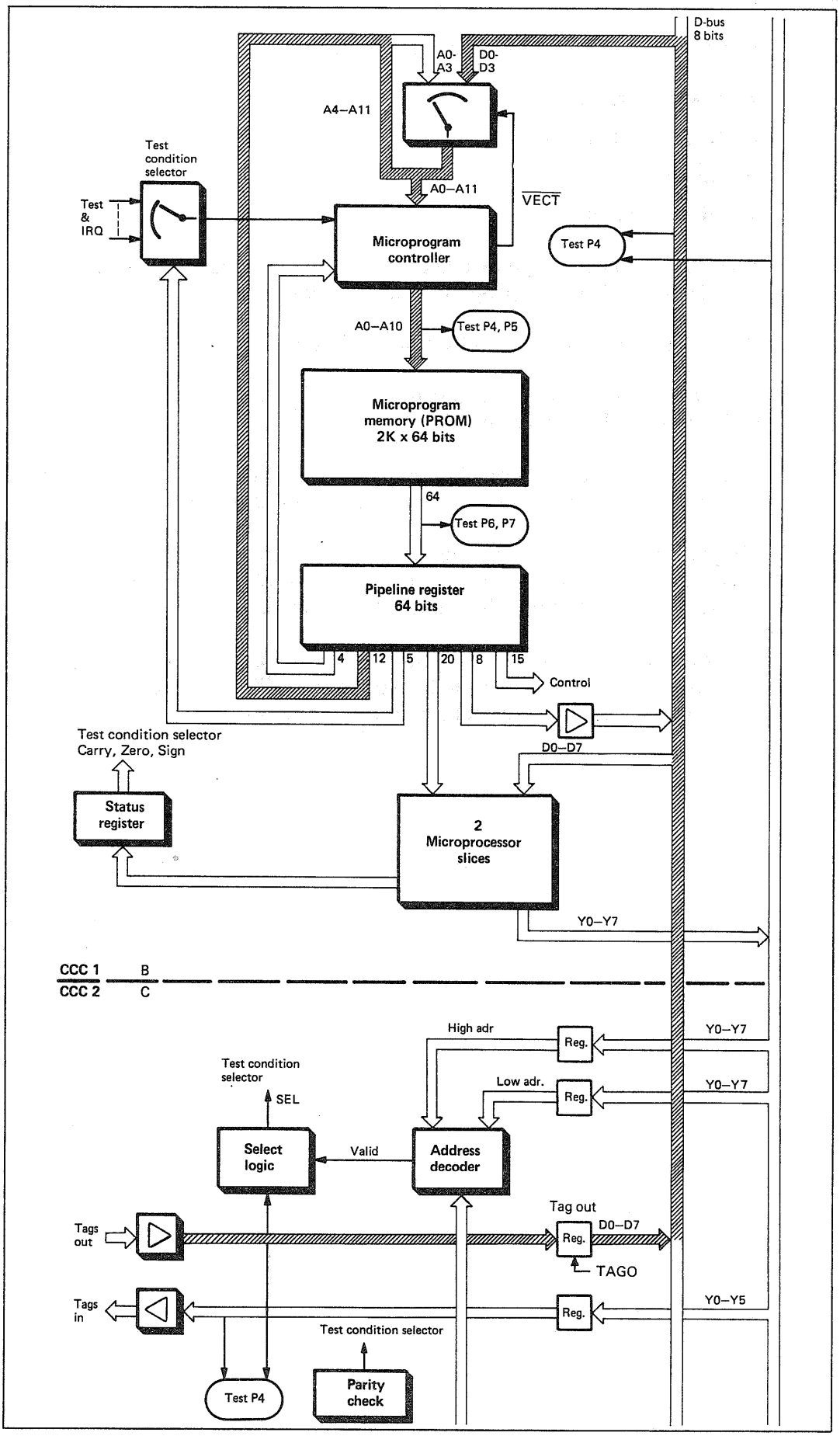


Fig. 46. Decoding of Tags in

Reset Logic

Several types of reset signals are generated:

- Power-on reset
- Reset button reset
- System reset
- Selective reset

The three first reset types are taken together, forming the general reset signal which makes the program jump to address zero.

Power-on reset is generated by a timer circuit.

The Reset button is connected to CPB. It generates the NMI signal which is actually manual reset.

System reset is caused by the channel dropping Operational out. If Suppress out is low this leads to a general reset of all units connected.

If Operational out drops with Suppress out high, the channel has generated a Selective reset. This reset, which is sensed by the program, is only causing reset in the unit which has Operational in high.

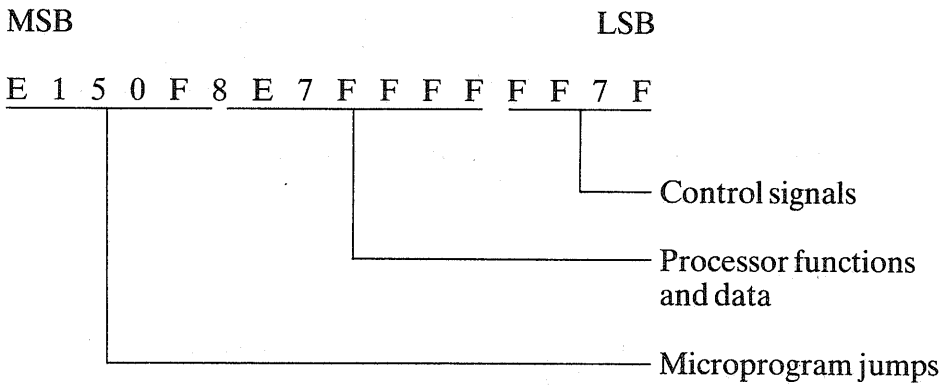
When a reset has caused the program to jump to address zero, the program route from there on depends on the type of reset. Therefore, a latch circuit stores this information so that the program can test it.

Appendix 1

Microinstructions

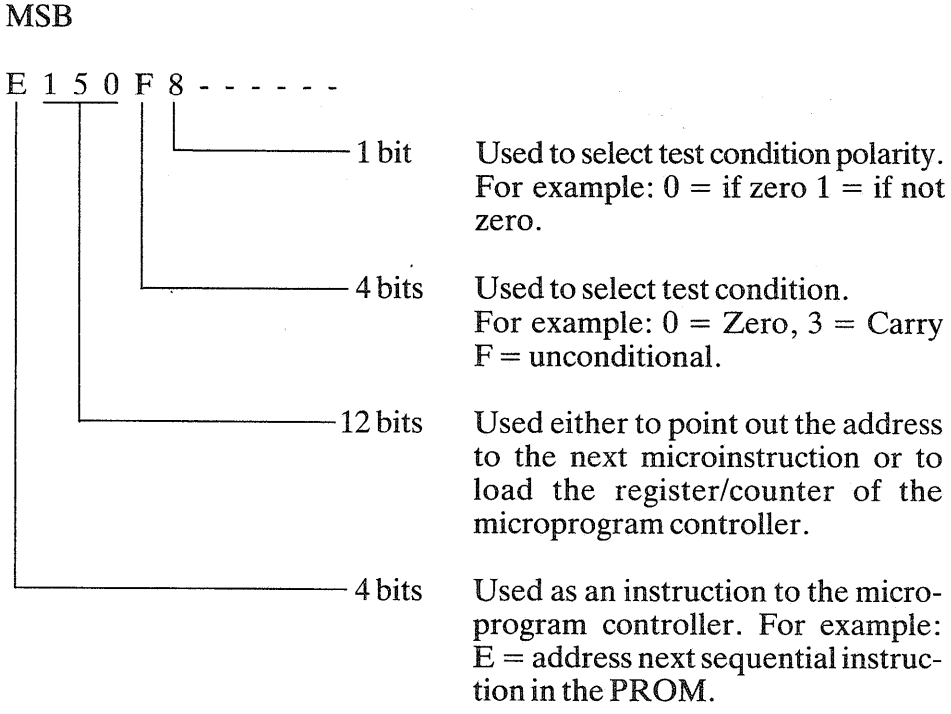
General

The microinstructions, which control the CCC unit, are presented as 64-bit words from the pipeline register. In the program list, each microinstruction is presented as a 16 character string. Each character represents the hexadecimal code of four microinstruction bits. To make it easy to understand the function of the microinstruction, a microinstruction can be divided into the following three sections:



Microprogram jumps

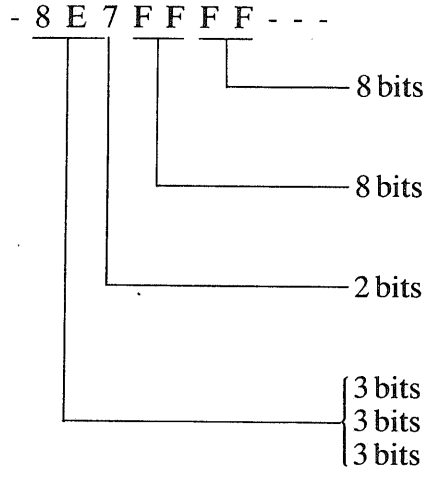
The 21 most significant bits in the microinstruction are used to control the microprogram controller and the test condition selector.



See also section entitled Microprogram controller.

Processor Function and Data

The following 27 bits are used to control the processor and to put immediate data on the D-bus from the microinstruction PROM.



Used to address the RAM (A and B address) of the processor.

Immediate data to the D-bus from the microinstruction.

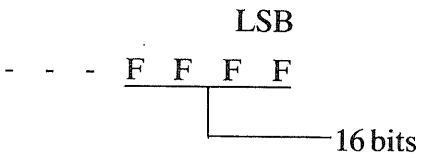
S1, S2 used to control the shift selector of the processor.

ALU source control
ALU function control
ALU destination control

See also section entitled Microprocessor slice.

Control Signals

The least significant bits in the microinstruction are used as control signals such as R/W, register latch, etc.



Used as control signals

Bit	Hex	Signal	Function
0		CHANGE	Changes the access direction of the line buffer memory by affecting the direction flip-flop.
1		Srv in	Service in to the channel
2		Es	Enable to the communication memory
3		R/W	Read/Write to the communication memory
4 5 6		These microinstruction bits are decoded in a three-to-eight decoder, giving the following result:	
	0	INC	Increments the address counter of the line buffer memory

	1	CLR	Clears the address counter
	2	IRQCCC	Interrupt request to CPB
	3	RPC	Resets the parity check flip-flop
	4	LDC	Loads the carry flip-flop in the status register
	5	Disable	Sets the enable/disable flip-flop in state disable
	6	Enable	Sets the enable/disable flip-flop in state enable
	7		Not used
7		C_D	Carry in to the processor
8		W	Write to the line buffer memory
9		R	Read to the line buffer memory
10, 11, 12		Decoded in a three-to-eight decoder	
	0	DIN	Loads data in register to the channel
	1	TAGI	Loads tag in register to the channel
	2	STAA	Loads the communication memory address register from the Y-bus
	3	MASK	Loads the mask register from the Y-bus
	4	ADR	Loads the address (CPL- and device address) into the address register from the Y-bus
	5	IND	Loads register for the indicators on the front panel from the Y-bus
	6	LDZ	Loads the zero and sign flip-flops in the status register
	7		Not used
13, 14, 15		Decoded in a three-to-eight decoder	
	0		Resets the interrupt flip-flop
	1		Not used

2		Not used
3	DATAO	Enables data out from the channel to the D-bus
4	TAG O	Enables tags out to the D-bus
5		Enables immediate data from the pipeline register
6	Sta read	Enables the buffer for data transfer from the communication memory to the D-bus
7	CONDS	Clocks the condition register (DIS, TEST and SYSTEMRESET)

Appendix 2

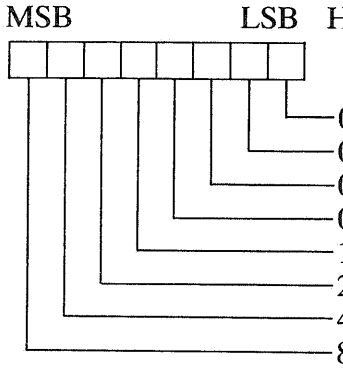
Microprocessor Register Allocations

The 16 word RAM of the microprocessor contains the following registers.

Addr	Mnemonic	Name
0 1 2 3	W1 W2 W3 W4	Working registers
4 5	Counth Countl	High Low byte count
6 7	Not used Not used	
8 9 A B C D E F	Scan Mskreg Adrreg Stareg Senreg Pgmsta Ticopy Cmdreg	Device address scanner Mask for local device number Device address register Status register Sense register Program status register Tags in copy register Channel command register

W1 to W4	Working registers
Counth/Countl	Counts the number of sent or received data bytes to or from the channel
Scan	Address counter for scanning the device status table (00-1F)
Maskreg	Contains the highest device address.
Adrreg	Contains the lowest device address.
Stareg	Register for storing the status of the device, before sending to the channel.
Senreg	Register for storing the sense byte of the device.

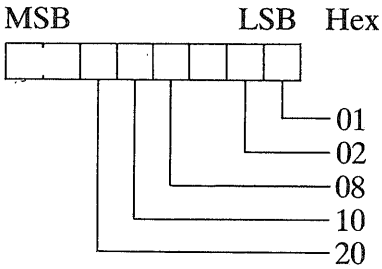
Pgmsta Program status register



Combinations:

A0	Stareq/Intbusy
98	Stareq/Intdisc/Selres
88	Stareq/Selres
18	Intdisc/Selres
19	Intdisc/Selres/Timeout
60	Cubusy/Intbusy

Ticopy Copy of tags in register, used when changing bits without intervention with other bits in tags in.



Combinations:

30	Adri/Opli
28	Opli/Stai
18	Adri/Stai
13	Adri/Reqis
03	Reqi/Reqis

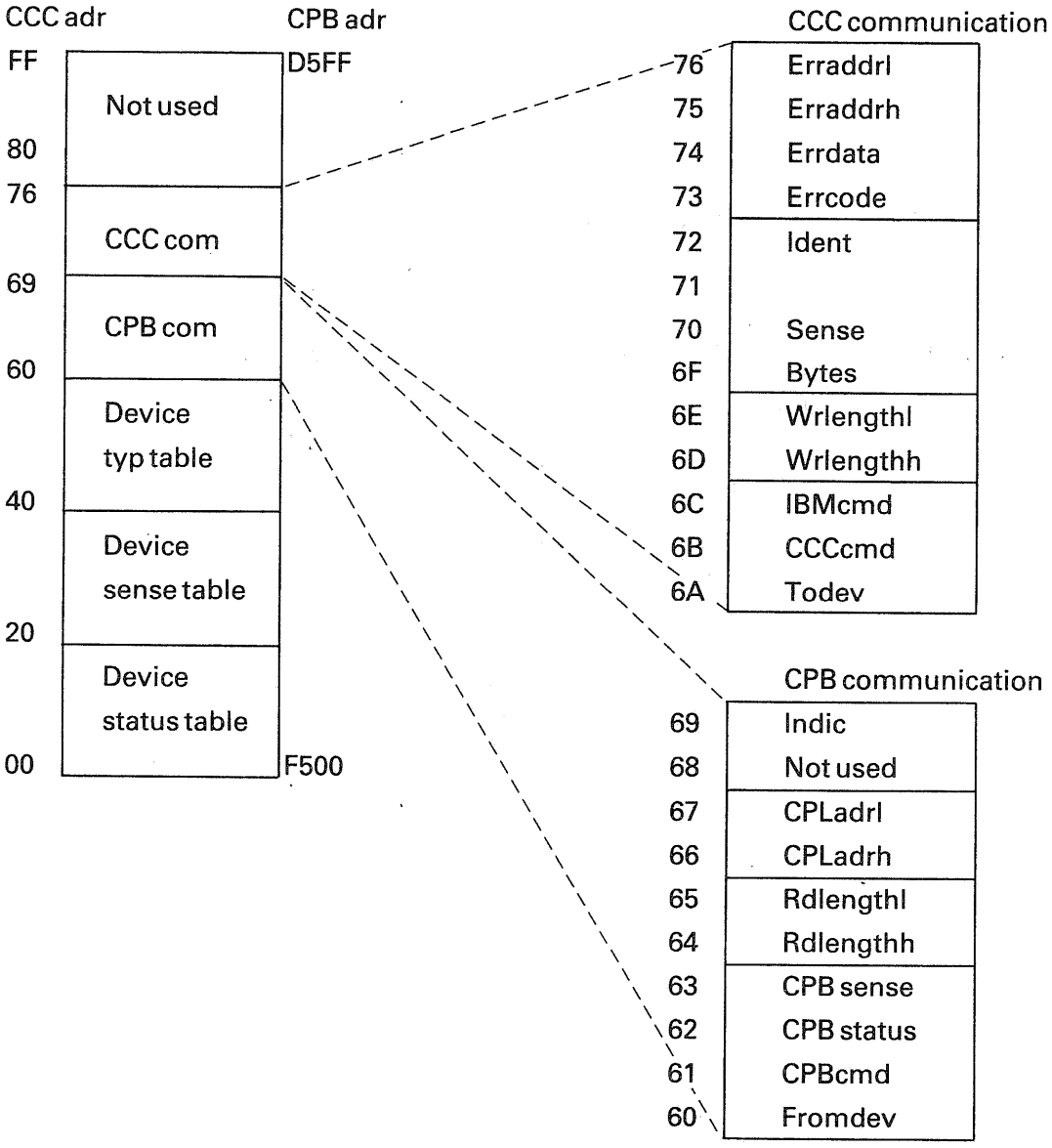
Cmdreg Commands to or from the channel are temporarily stored in this register.

Appendix 3

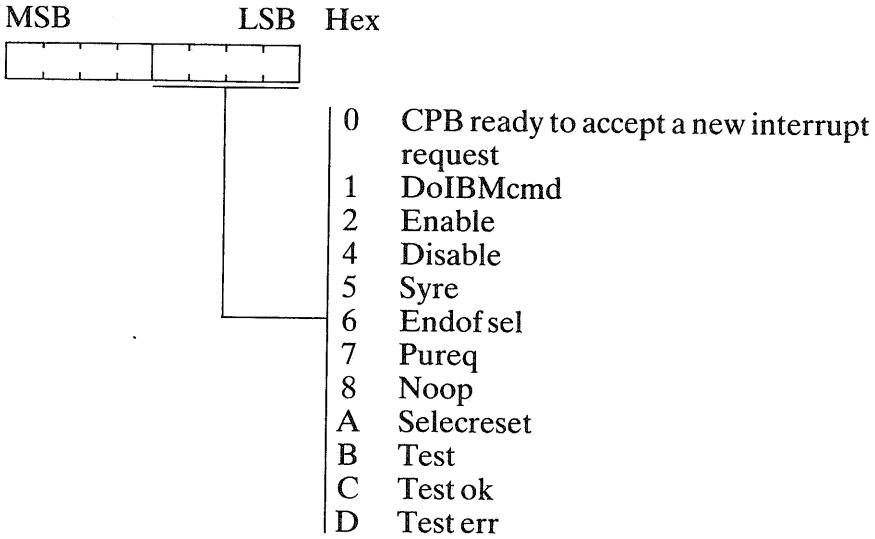
Communication Memory Allocations

General

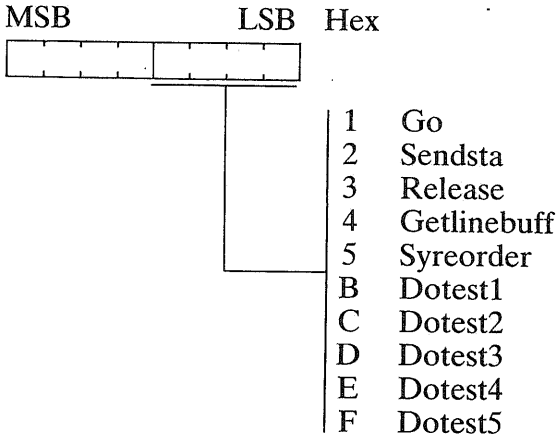
The communication memory, multiplex shared between the CCC and the CPB, contains the following areas:



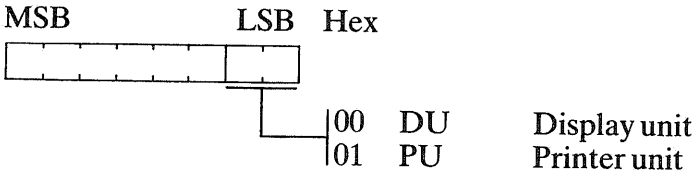
CPBcmd, Command to CPB



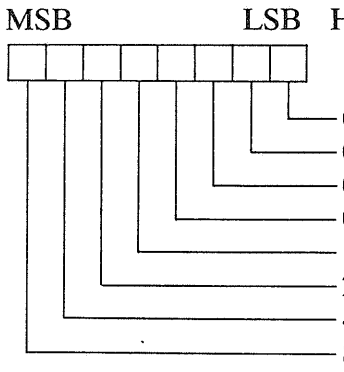
CCCcmd, Orders from CPB (Command to CCC)



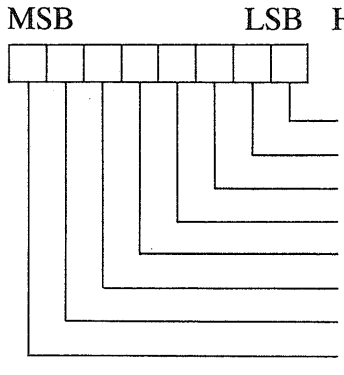
Device type word



Device sense word



Device status word



Combinations:

94	AT/DE/BU
84	AT/DE
50	SM/BU
0C	CE/DE
0E	CE/DE/UC
06	DE/UE
03	UC/UE

Appendix 4

Microprogram Function

General

This appendix will explain the function of the CCC by presenting some parts of the microprogram. To understand this information, the reader must be familiar with the channel signals and the signalling (data and commands) between the computer and the different devices. For information about registers and commands, see Appendices 2 and 3. As a detailed example of the microprogram design, the read sequence is chosen (a device sends data to the computer). The reader can then follow other sequences by means of the microprogram list.

The microprogram of the CCC has a general design, which means that all types of channels can be handled (selector-, burst multiplexer-, and byte multiplexer channels). To interface the high-speed channel as fast as possible, some functions must be executed in hardware. These special functions are described in the detailed hardware description.

Start-up routines

The start-up routines (see Fig. 47) include internal tests, communication test and initiation routines of the CCC. These routines are entered at Power on (when power to the CPL is switched on), Reset (the Reset pushbutton on the front panel is depressed) or System reset (ordered from the channel at any time).

Power on/Reset

Immediately after power to the CPL is switched on or the Reset push-button is depressed, the lamps L1-L4 on the front panel are switched on. Lamp 1 is then switched off as a sign that the program has started.

Then the program performs a test of the microprocessor. If the test is carried out correctly lamp 2 is switched off, else the program enters an error loop (see later on).

Then the program tests the line buffer, by writing and reading data (zero and one). If no errors occur lamp 3 will be switched off.

These two tests will take place very rapidly. That is to say, lamp 1-3 will only be on for a short while at power on or reset if no errors occur in the tests.

If an error occurs, only lamp 1 or lamp 1 and lamp 2 will be switched off depending on which test failed, and the program will stop. At program stop an error code will be available at the Y-bus.

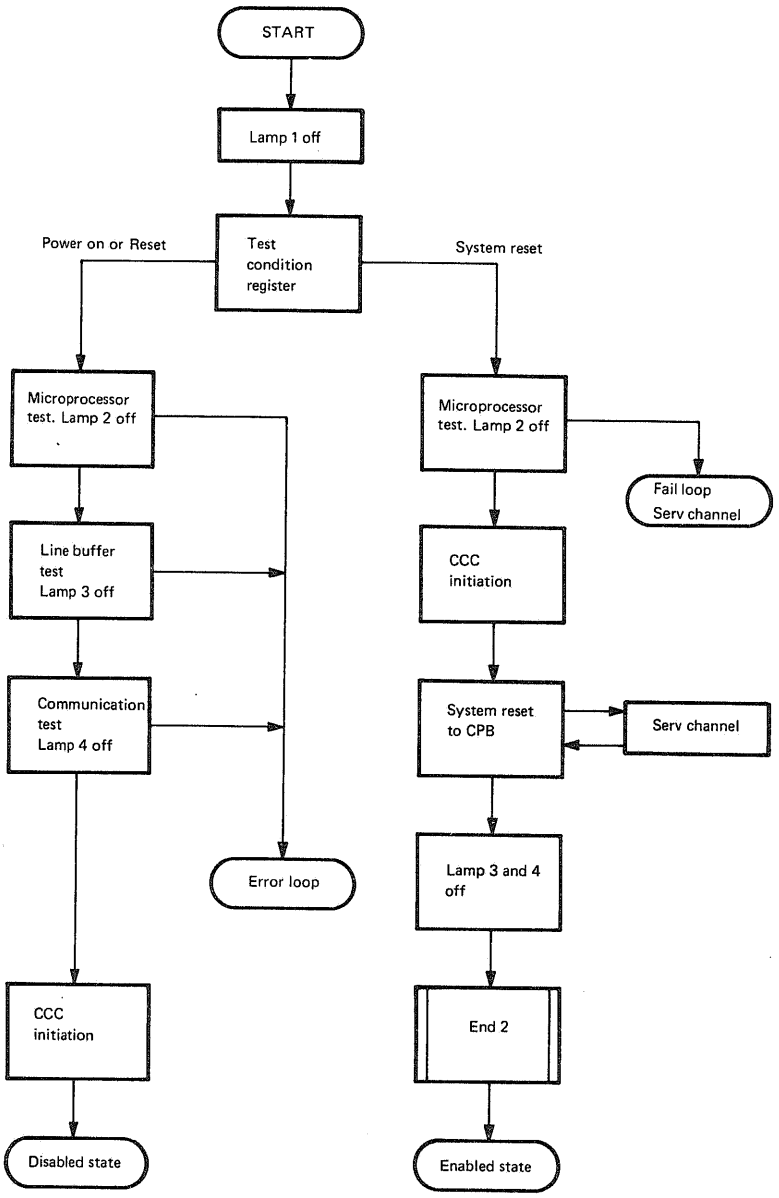


Fig. 47. Start-up routines

Lamp 4 will be on until a communication test has been correctly carried out between the CCC and the CPB. This test can not be entered until the system program has been loaded into the CPB, which means that lamp 4 might be on for approximately 15 seconds before it will be switched off. If an error occurs lamp 4 will be steadily on and the program will enter the error loop.

If all tests have been carried out correctly (lamp 1 – 4 switched off) the program enters the CCC initiation program. This means that the channel address to the CPL is set up in the CCC address decoder. The channel address is built up by one high address (highest device address) and one low address (lowest device address). Between these two addresses, the CCC detects a valid address to the CPL.

The two addresses are determined by the system program and are available in the communication memory, where the CCC fetches the addresses and stores them in two registers (high address register and low address register).

Moreover, the registers in the microprocessor are reset in the CCC initiation state. The program then enters the disabled state.

NOTE
If there has been a Power on or a Reset start, the CPL is not yet connected to the channel. (The Select out signal of the channel passes the CPL unit both with high and low priority). The Select out signal can only be caught in the enabled state (the Enable lamp on the front panel on).

System reset

A System reset can be ordered at any time from the channel, when the CPL is in the enabled state (the Enable lamp on the front panel is on). The program then enters the same point as at Power on and Reset (address $0000_{(16)}$ in the microprogram). By sensing a condition register (Enable, Disable and System reset) the program decides if it was a System reset.

The difference between Power on/Reset and System reset is, that the CCC is connected to the channel in System reset. The CCC thereby has to serve the channel during the tests.

The System reset starts with a microprocessor test, as in the Power on/ Reset state. If the test is carried out correctly lamp 2 is switched off. If the test fails, error codes are set up in the communication memory to inform the CPB, and the program enters an error loop still serving the channel.

If the microprocessor test was carried out correctly the program initiates the CCC (CPL addresses to the address decoder and reset of the microprocessor registers, see Power on/Reset).

The program then sets System reset to the CPB (Syre to CPB command in the communication memory) and sends an interrupt to the CPB. While the CPB executes the System reset command, the CCC is serving the channel. If the channel addresses the CPL in this state, the CCC immediately sends back CU busy status to the channel.

When the CPB has executed the System reset command, the CPB returns an interrupt to the CCC. The CCC acknowledges this interrupt by switching lamp 3 and 4 off.

This means that a System reset can be noticed by watching the lamps on the front panel. A system reset rapidly switches lamp 1 and 2 on and off and makes lamps 3 and 4 glow steadily for about 10 seconds.

Since System reset occurs in the enabled state, the System reset program module ends by entering a general subroutine End 2, which resets different registers in the communication memory and in the microprocessor. For information about End 2 see later on.

The program then returns to the enabled state.

Disable

In the disabled state (see Fig. 48) the CCC is not connected to the channel, that is to say the Select out signal from the channel is by-passed via the relays on the CIB board both in high and low priority. The microprogram runs in a loop and can be affected only by the switches on the front panel or by interrupt requests from the CPB (which today is considered as an error). The Enable lamp on the front panel is off.

If Test is selected on the front panel the CCC enters a background test loop and can be ordered by the CPB to perform microprocessor-, RWM-, and communication tests.

If the Disable/Enable switch on the front panel is set to position enable, the program enters the module Dis → enable, which is a program for changing state to enabled.

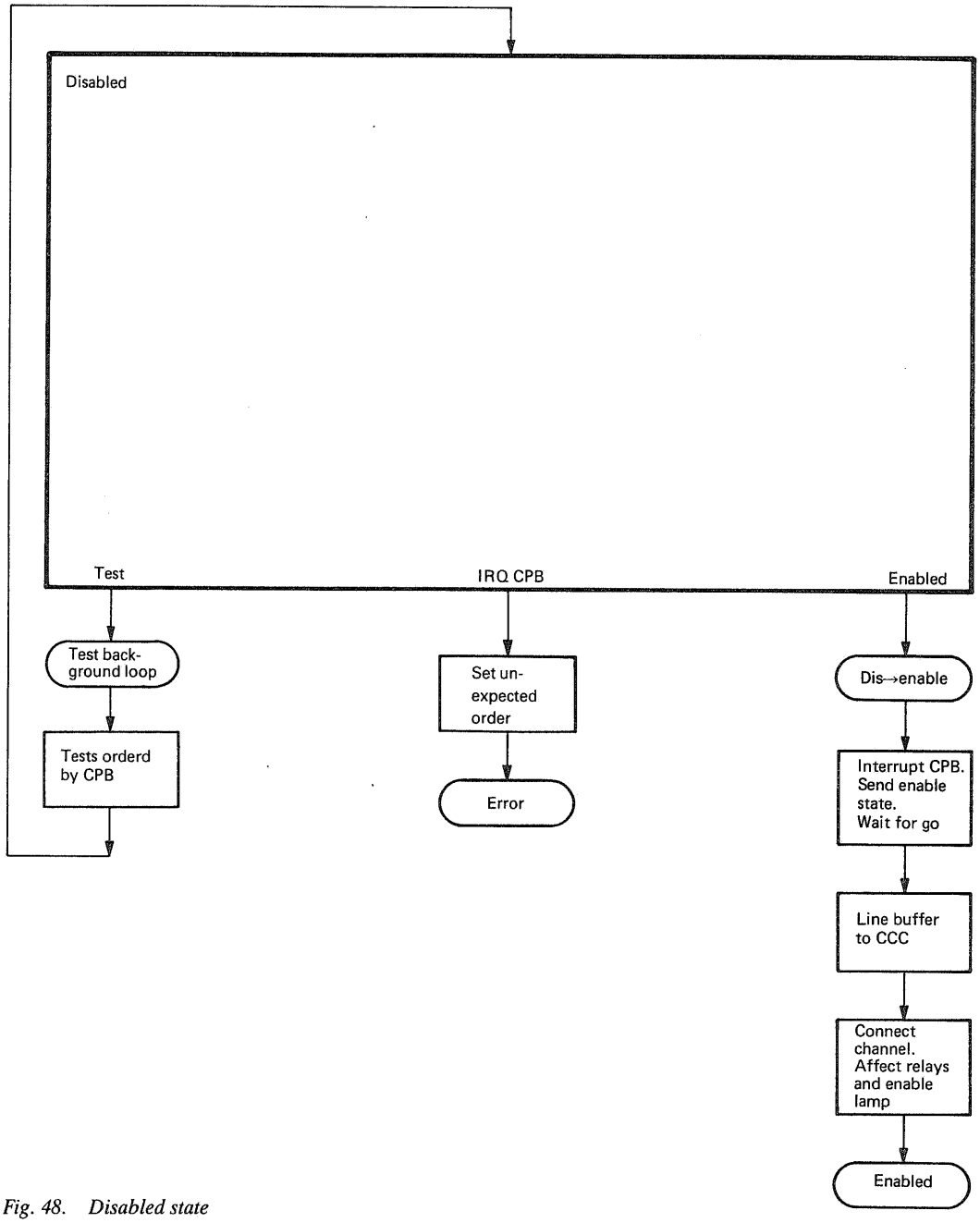


Fig. 48. Disabled state

Dis → enable

This program module, used for change over from disabled to enabled state, first sets up the status meaning enabled and then sends an interrupt to the CPB. When the CPB has performed the interrupt routine and returns to the CCC with a "go" command, the CCC changes over the line buffer to direction CCC.

Then the microprogram affects the relays in such a way that the Select out signal from the channel is fed in to the detector circuits of the CCC. The output signal, which affect the relays simultaneously lights the Enable lamp on the front panel, and the program enters the background loop of the enabled state.

NOTE
Selection of high or low priority is made by means of the switch on the CIB board (Up – high priority, Down – low priority). See circuit description.

Enable

When the microprogram enters the background loop of the enabled state (see Fig. 49), the CCC is connected to the channel and the Enable lamp on front panel is on. In the enabled state background loop the signals DIS (Disable from the switch on the front panel) and SEL (valid address and Select out from the channel) are tested. (IRQ CPB is actually also tested but not supposed to occure in current programs.) If none of the signals is present, the program checks the status word of one device (from the Device status table in the communication memory). If there was no AT (Attention) or DE (Device end) in the status, the device address scanner (Scan) in the processor will be incremented and the program returns to the start of the background loop. The looping continues until an AT/DE status is found somewhere in the device status table, SEL is detected from the channel, or the Enable/Disable switch is set to position disable.

If a SEL is detected, it means that the channel wants to enter a write or read sequence, and the program enters the module Channel sel described later on.

NOTE
If the Normal/Test switch is set to position test, the program immediately disconnects the CPL unit from the channel and stops the microprogram execution. This function is used as an emergency stop.

Enable → dis

If a DIS is detected in the background loop of the enabled state, the Enable → dis microprogram module is entered. This program releases the relays in a special sequence (necessary for the fast channel), and sends an interrupt to the CPB telling that the CCC enters the disabled state. When the CPB returns from the interrupt with a "go" command, the CCC switches over the line buffer to the CPB and enters the disabled state.

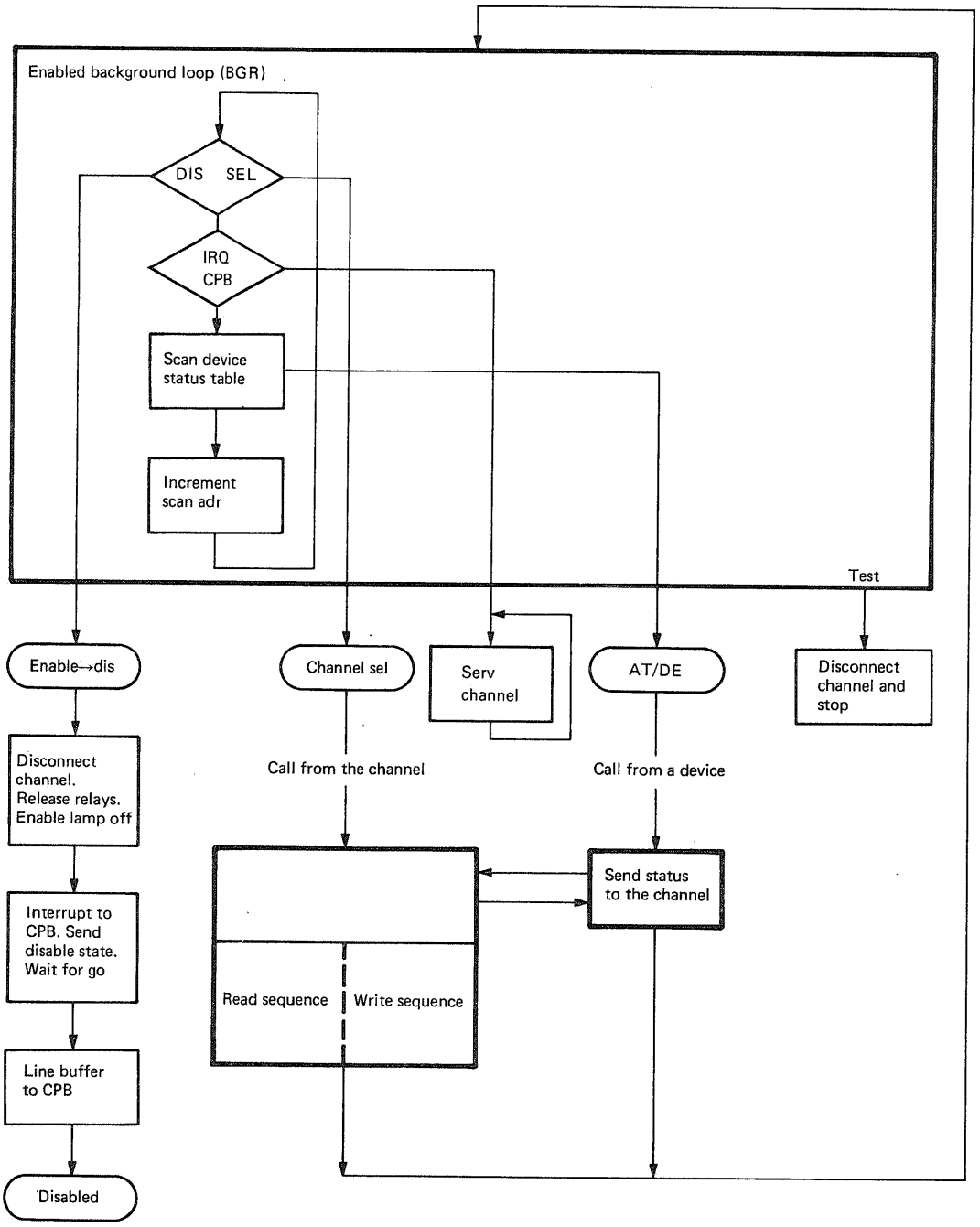


Fig. 49. Enabled state

Attention/Device end

When the status Attention or Device end is found in the device status table, the microprogram immediately enters the AT/DE module, (see Fig. 50). The program then transfers the status to the Stareg in the microprocessor and resets the Attention bit in the status register of the device status table. Then the Unit check bit (UC) in the device status is checked.

If the Unit check bit is set, there has been some error in the device. If so the status of the device is transferred to Senreg in the microprocessor for later transmission to the channel.

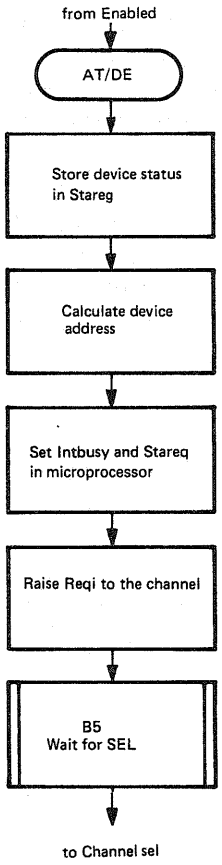


Fig. 50. AT/DE

Then the "real" device address is calculated, since the address pointer in the Scan register in the microprocessor scans only between $00_{(16)}$ and $1F_{(16)}$ in the device table. That is to say, the Scan register address is added to the low CPL address, which is available in the CPLadrl register in the CPB communication section of the communication memory. The device address is then stored in the device address register (Adrreg) in the microprocessor.

To indicate that the CCC is now busy and cannot serve a call from the channel, the Status request and Interface busy bits in the Pgmsta register of the microprocessor are set.

Now all required information about the device has been set up in the microprocessor (device address, device status and, if Unit check was set, device sense).

The program status register (Pgmsta) in the microprocessor indicates that the CCC is busy (Intbusy bit set). The CCC can now set Reqis to the channel.

The tag in signalling to the channel proceeds via the Ticopy register in the microprocessor. Reqis in the Ticopy register is set and Ticopy is clocked to the tag in register, where Req in to the channel is raised.

The program then enters a background loop (B5) waiting for a call back from the channel, that is to say the internally generated SEL signal. The SEL signal is generated when Select in is raised by the channel and the Bus in contains a valid address for the CPL unit. The SEL signal is fed to the test condition selector, where it is tested by the background loop.

Since the program is now waiting for a call from the channel, the module Channel sel (see Fig. 51) is entered when the SEL signal becomes active. In the general Channel sel module, which is decribed in detail later on, there is a test if the call is an answer to the request. If so (if the Adr out is not raised in connection with Select out signal) the program immediately enters the module Srvstareq (Serve status request).

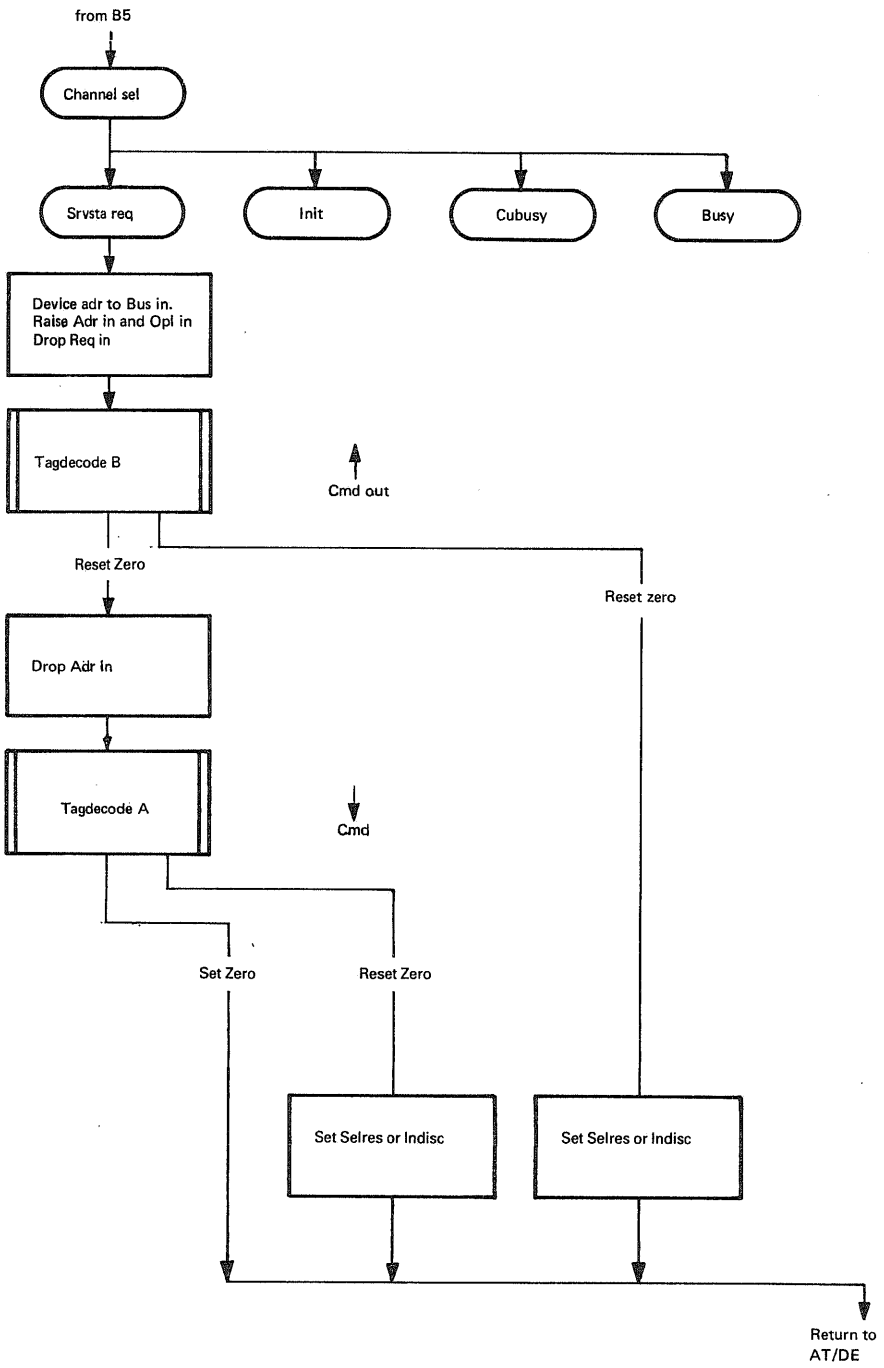


Fig. 51. Channel sel

Srvstareq

The Srvstareq program immediately answers the call from the channel by latching the device address (stored in Stareg of the microprocessor) to the Bus in, raising Opl in and dropping Adr in. Moreover Req in is dropped. The tag in operation is done via the tag in copy register, Ticopy, in the microprocessor, which always contains a copy of the present status of Tag in.

By that sequence the CCC has answered the channel call and enters the Tagdecode B table waiting for Cmd out to raise.

Tagdecode is a 16 byte table in the microprogram. When the module Tagdecode is entered the four lowest address bits to the microinstruction memory are taken from the tag out register (Adr out, Opl out, Cmd out and Srv out). The Tagdecode procedure is described in detail in under Channel interface logic.

In the Tagdecode B table three things can happen: Cmd out goes up (Zero is set), the channel orders selective reset (Selres) for the device (Selres in Pgmsta is set and Zero is reset) or the channel orders interface disconnect (Indisc in Pgmsta is set and Zero is reset).

If Selres or Indisc was ordered from the channel in the Tagdecode B table the program immediately returns to the AT/DE program module. If Cmd out was raised, the program resets Adr in, resets Zero and goes to Tagdecode A.

In Tagdecode A the program is waiting for Cmd out to drop. When that happens, Zero is set and the program returns to the AT/DE program module. In Tagdecode A the program can order Selres or Indisc as in the Tagdecode B case.

When the program then returns to the AT/DE program module the call from the channel has been answered by transmission of the device address. The channel can either have responded by saying "continue" or by ordering a Selective reset or an Interface disconnect.

What happened in the Tagdecode table is found out by testing Zero. If Zero is reset the program enters a module for serving Selective reset or Interface disconnect (ID/SR, described later on). If Zero is set the program enters the general module Sta for sending status about the device.

Sta

In the Sta subroutine (see Fig. 52) the device status, stored in the Stareg in the microprocessor, is clocked to the bus in register. Then Tag in is raised via the Ticopy register in the microprocessor and the program enters Tagdecode B waiting for Srv out or Cmd out to rise. When this happens there is a Zero test just as in Srvstareq. If Zero is reset, the Selres bit in Pgmsta is set and the program returns to AT/DE.

If Zero is set there is a test of Srv out. If Srv out is raised by the channel it means that the channel cannot receive the status just now. If so, subroutine Stacksta is entered, to stack the status until the channel is ready to receive it. Stacksta subroutine is described later on.

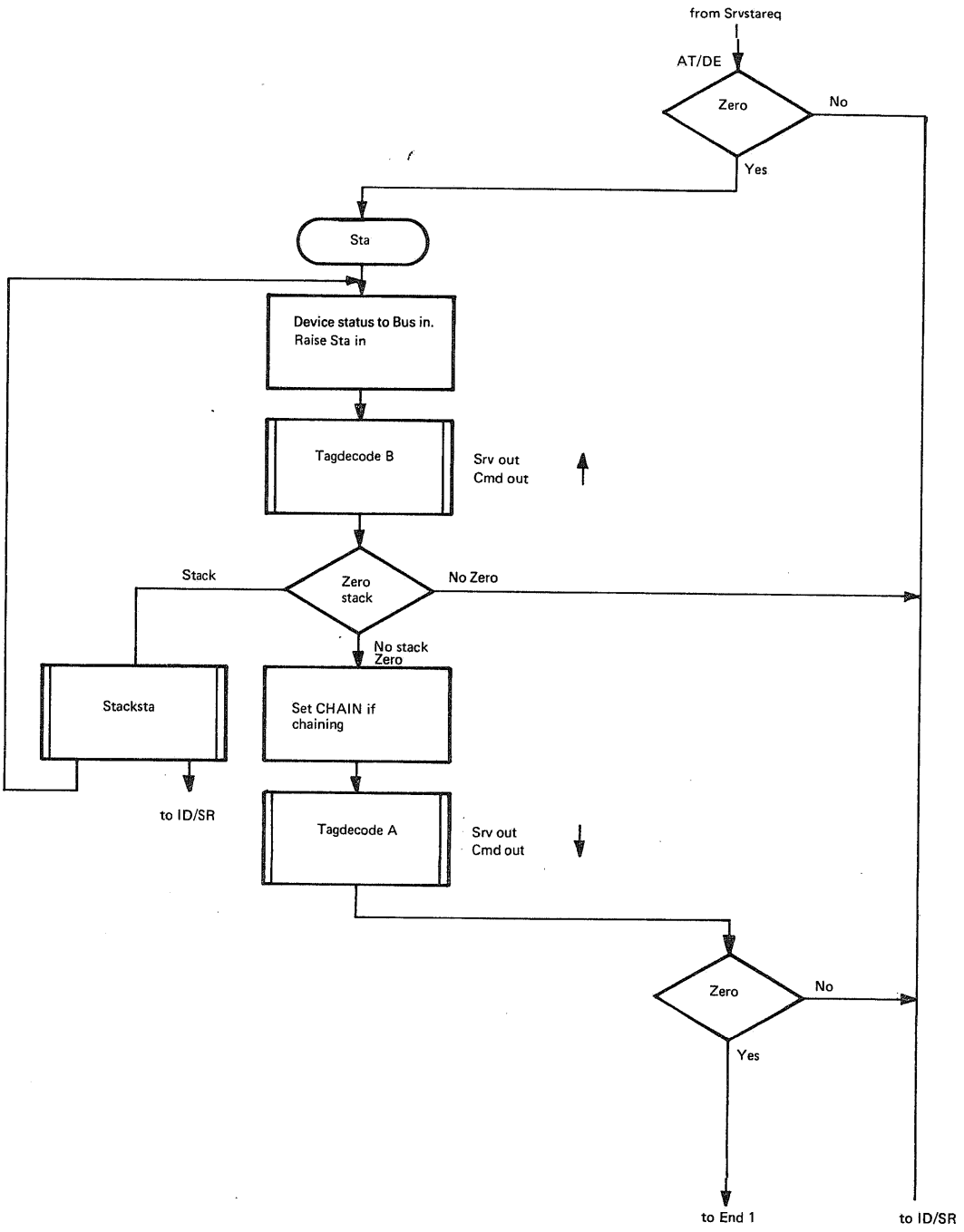


Fig. 52. Sta

If Srv out is not up a chaining flip-flop in the control circuits of the channel is sensed via the test condition selector. If the channel orders chaining (the flip-flop is set), the Chain bit in Pgmsta of the microprocessor is set. Then Zero is set and the program enters Tagdecode A waiting for Srv out to drop.

When Srv out drops the program returns to AT/DE and tests Zero. If Zero is reset, a selective reset has been ordered from the channel and the program enters the ID/SR module, for serving the reset. If Zero is set the program continues to End 1, which is a general routine for entering the background loop of the enabled state.

That means that we now have ended the sequence AT/DE, where an address and a status for a device have been sent to the computer. To return to the background loop of the enabled state, the program however must pass some general End routines.

End 1

The general End 1 routine (see Fig. 53) starts with a test of the Eos bit in Pgmsta (End of selection bit in the Program status register in the microprocessor). If DE (Device end) was sent to the channel the program continues in the End 1 module otherwise the End 2 module is entered. End 1 continues by resetting the Stareg of the microprocessor. Then the program checks Cubusy and Chaining (not described) and enters the Endsel program module.

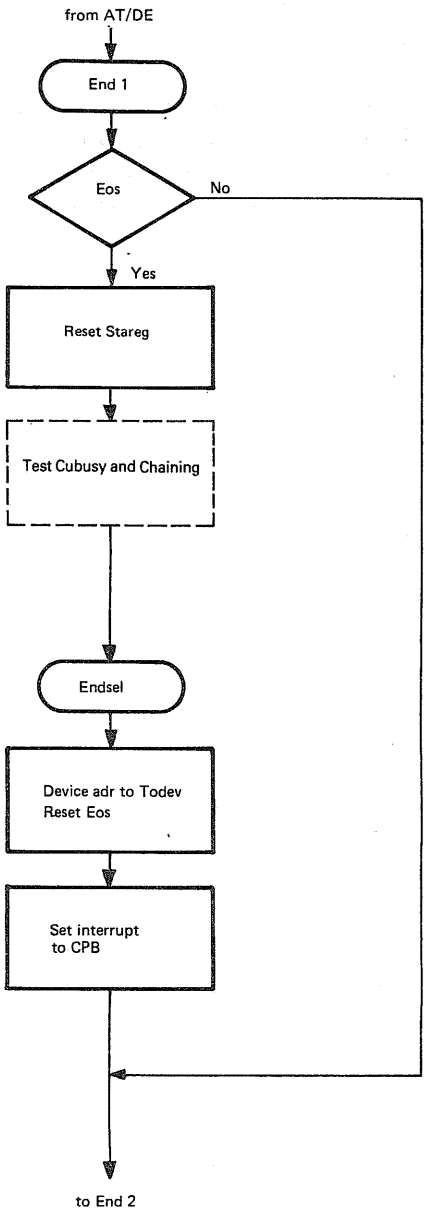


Fig. 53. End 1 and Endsel

Endsel

Endsel performs an acknowledge to the CPB that the status for a device is sent to the channel. First the device address is stored in Todev and Eos command is stored in the CCCcmd of the CCC communication memory area.

Then the Eos bit in Pgmsta is reset and an interrupt is sent to the CPB. The program then enters the End 2 module without waiting for an acknowledge of the interrupt sent.

End 2

The End 2 program module (see Fig. 54) first checks if the Cubusy bit in Pgmsta is set. If not, the Stareq and Intbusy bits in Pgmsta are reset and the line buffer is set in direction CCC. Then the Stareg (Status register) in the microprocessor is reset and the program returns to the background loop of the enabled state.

If we continue in our example, the device now has sent status Attention for a read sequence and the CPB is ready to accept a Read command from the channel. The CCC, however, has no information stored about a coming read sequence (CCC loops in the enabled state).

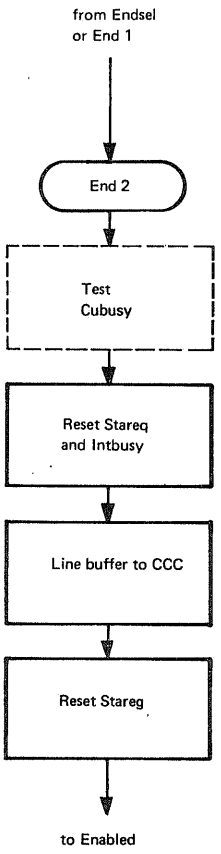


Fig. 54. End 2

Channel sel

Then, when a valid address and the Select in signal comes up, the program enters the Channel sel program module (see Fig. 55). This module was also used when the device sent status to the channel (AT/DE). This time the same tests are performed in the module.

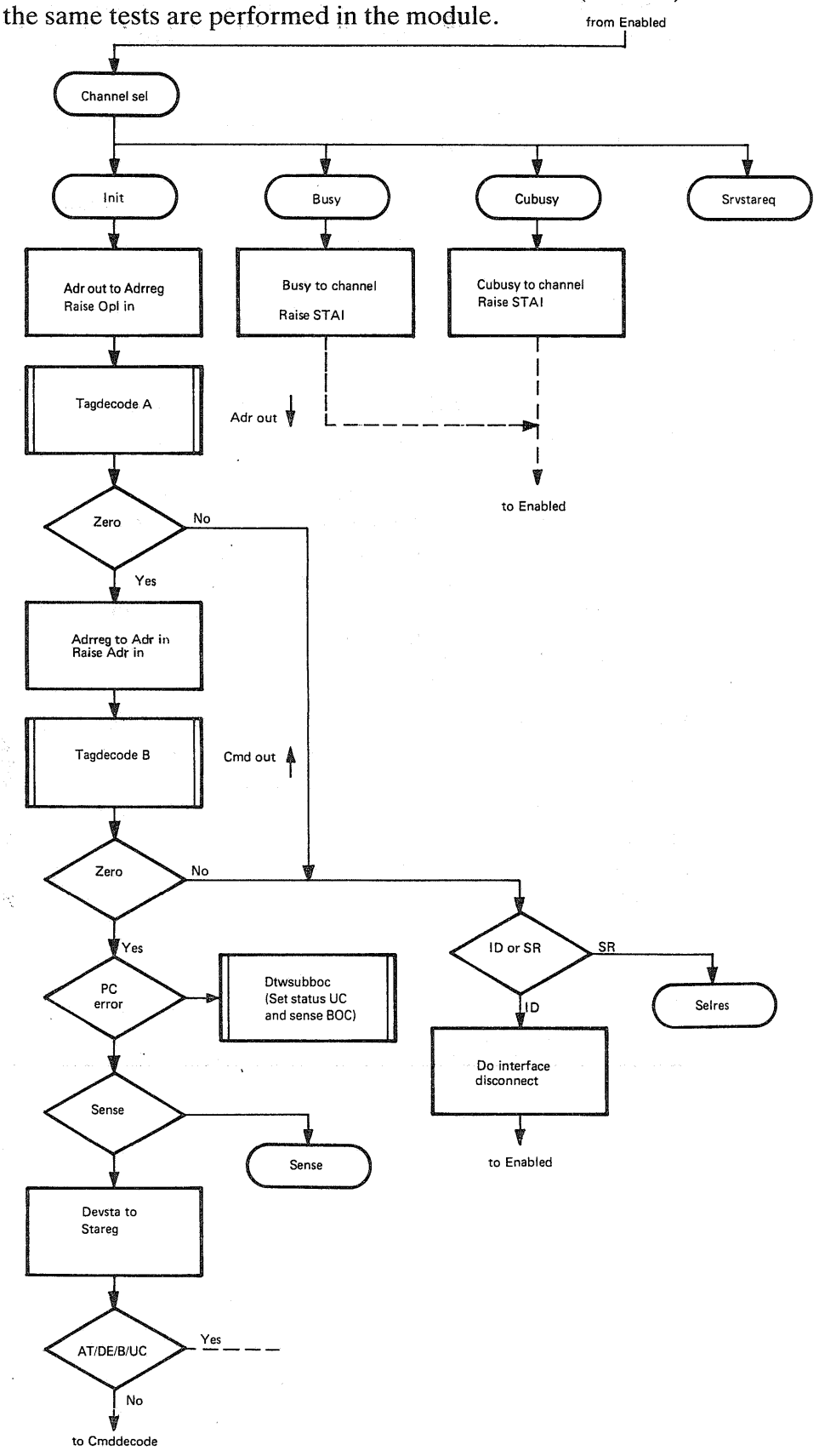


Fig. 55. Channel sel, Init

In the previous case Srvstareq was selected, thus Adr out was not raised. In this case there is a check if Intbusy is set in Pgmsta. If not the program enters the Init program module.

If Intbusy is set there can be two types of busy: Cubusy or Busy. At Cubusy (control unit busy) the CCC is busy serving another device than the one just addressed by the channel. If so, Cubusy status is sent to the channel (via BUSI and STAI raised).

At Busy, the CCC is busy serving the device that the channel addresses. Then Busy is sent to the channel.

The two types of Busy program modules also contain several other operations not described here.

Init

In our case we assume that the CCC is not busy, which makes the program enter the Init (Initial selection program module).

The Init module immediately transfers the address from the channel to Adrreg in Pgmsta of the microprocessor and raises Opl in as an acknowledge to the channel. The program then enters the Tagdecode A table after resetting Zero, waiting for Adr out to drop. If Zero is reset after passing Tagdecode A it means that the channel has ordered an Interface disconnect or a Selective reset, as in the first case (see AT/DE). If so the program enters the following Zero test.

If Zero is set, the contents of the Adrreg is sent back to the channel (via BUSI and ADRI raised). The program then enters Tagdecode B waiting for Cmd out to rise. If Zero is reset after Tagdecode B, the program enters a test of Interface disconnect (ID) or Selective reset (SR). If ID was ordered, disconnection is performed immediately and the program returns to the background loop of the enabled state. If SR was orded the Selres module is entered.

When Cmd out was raised in Tagdecode B it means that the channel has got a valid command on Bus out. If there is a parity error in the command (the PC flip-flop is set) the program enters the subroutine Dtwsubboc. If the command is a sense command (the channel wants to read the sense byte of the addressed device) the program module Sense is entered.

Then the program checks the status of the addressed device (Device status table of the communication memory). If there was no Attention (AT) Device end (DE), Busy (B) or Unit check (UC) the program enters the Cmddecod table.

In our example, the CCC has now received an address to a device, answered the channel with the same address, and checked that the device is free to receive a command.

Cmddecode

In the Cmddecode module, (see Fig. 56) there is first a check if the channel command is valid for the CCC unit. Then the command is decoded in the same way as in Tagdecode (by the address selector in position D bus).

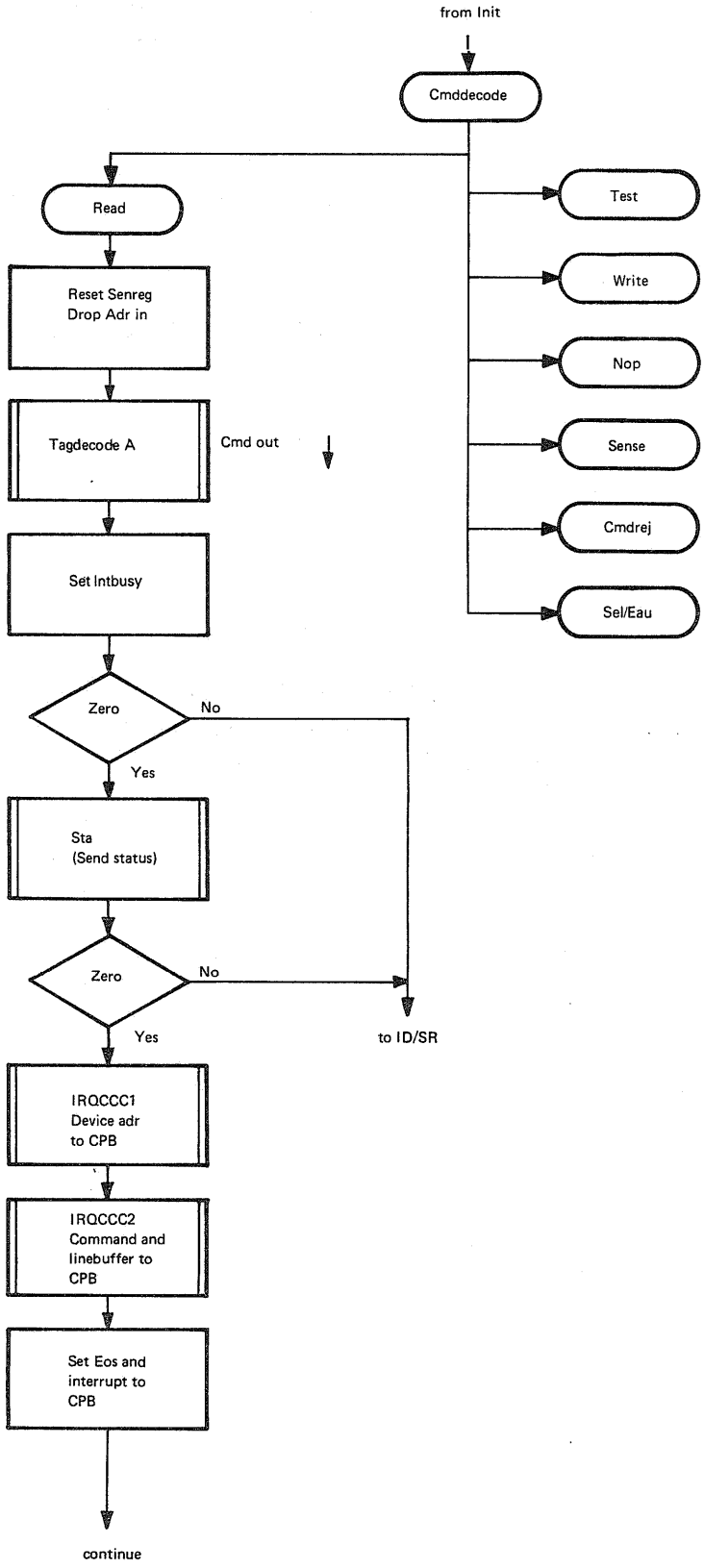


Fig. 56. Cmddecode

The following commands except Read and Write can be detected:

Sel/Eau	sent to, and carried out by, the CPB Sel = Select, Eau = Erase all unprotected
Sense ID	sent to the CPL at channel start-up. The CPL answers the command by informing the channel about the type of control unit (CPL) that is connected. That is to say, four bytes (Ident, location $6F_{(16)} - 72_{(16)}$, from the communication memory) are sent to the channel. The contents ($FF_{(16)}$, $32_{(16)}$, $74_{(16)}$, and $1D_{(16)}$ or $1B_{(16)}$ depending on the type of control unit) of Ident is determined by the system program.
Test I/O	makes the CPL send present status to the channel.
Nop	makes the CPL send different information depending on status.
Sense	makes the CPL send status for the addressed device.
Cmdrej	invalid command.

Read

In our example we suppose that the channel has ordered a read sequence. Then the Senreg in the microprocessor is reset and the Adr in is dropped, as a signal to the channel that the command is accepted. The program then resets Zero and enters Tagdecode A, waiting for Cmd out to fall. When this happens the Intbusy bit in Pgmsta is set, because the CCC is now busy.

Then Zero is tested to sense if the channel has ordered Interface disconnect or Selective reset. If so the program enters the ID/SR program module.

The CCC is going to send the device status, stored in Stareg in the microprocessor, to the channel. For this purpose the general subroutine Sta is used, as described in case AT/DE.

At return from Sta Zero is tested, because the channel can have ordered ID/SR.

The CCC has now received the device address from the channel and answered by returning the same address. A Read command from the channel has been decoded and the CCC has sent status for the addressed device. Now the CCC must inform the CPB that a read sequence is going to take place.

This is done by setting up the address to the device, in Todev in the CCC communication memory area. This task is performed by the subroutine RQCCC 1.

In IRQCCC 2 the channel command is transferred from Cmdreg in the microprocessor to IBMcmd in the CCC communication memory area and the line buffer is set in direction CPB.

Then the Eos bit is set in Pgmsta and an interrupt request is sent to CPB.

The CCC is now waiting for an interrupt from the CPB, (see Fig. 57) which means that the CPB has stored the data from the device in the line buffer. During the wait, the CCC resets the address counter of the line buffer by using the CLR signal, and enters one of two background loops (B2 or B3). Background loop B2 is entered if the channel works in multiplex mode (Opl in drops) and B3 is entered in burst mode.

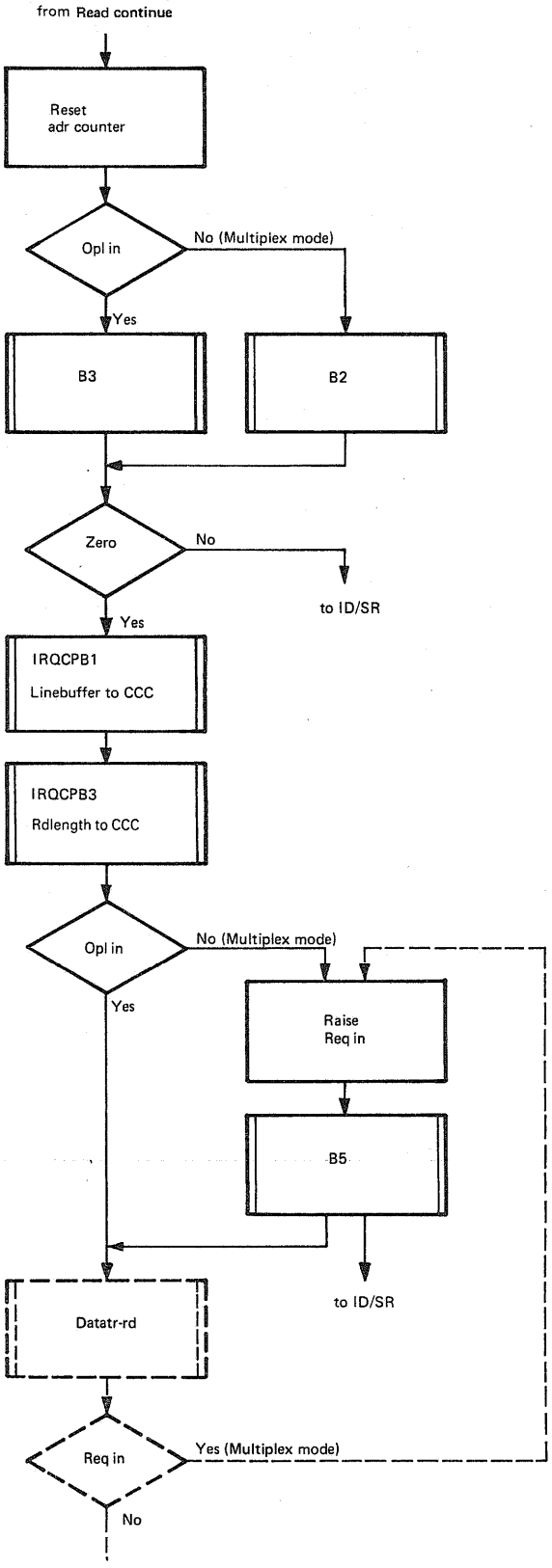


Fig. 57. Read (continued)

In the background loop the CCC is waiting for the interrupt from the CPB, but the CCC also serves the channel. If the channel calls during this sequence, the CCC immediately answers with status Cubusy (control unit busy).

When the CPB then interrupts the CCC, the program enters a Zero test to see if the channel has ordered ID/SR. If not, subroutine IRQCPB 1 is entered. Among other things IRQCPB 1 sets the line buffer in direction CCC. IRQCPB 3 then transfers the length. (Rdlengthh and Rdlengthl) from the CCC communication memory area to the working registers in the microprocessor.

Rdlength is stored in the communication memory by the CPB during the background loop and contains the length of data to be sent to the channel.

The CCC is now ready to send data from the addressed device to the channel.

If the channel works in multiplex mode (Opl in is dropped), the CCC has to raise Req in and wait, in background loop B5, for the SEL signal. Then subroutine Datatr-rd is entered to send one data byte and the program returns to raise Req in and background loop B5. This looping continues until all data bytes are sent. Then the Read-end module is entered.

If the channel works in burst mode the subroutine Datatr-rd is entered at once. In Datatr-rd all data bytes are sent (burst) and the program finally enters the Read-end module.

Datatr-rd

In the Datatr-rd subroutine, (see Fig. 58) data from the line buffer is sent to the channel via Bus in and the counters (Address counter, Counth and Countl) in the microprocessor are decremented. Then Srv in is raised, indicating that data on Bus in is valid, and the address counter to the line buffer is incremented. Then the program is waiting for Srv out and Sel out to rise during four clock periods. In burst mode Srv out and Sel out usually rise during this time and the program drops Srv in, transfers a new data byte to Bus in and decrements the counter (Count) in the microprocessor.

If Srv out and Sup out are dropped the program immediately returns to raise Srv in and increments the address counter. If Srv out and Sup out are not dropped the program enters subroutine Waitsrvdn (Wait for service down) before returning.

In burst mode this loop normally is repeated until Counth and Countl in the microprocessor are both equal to zero, then the program enters Dtr-end.

If the channel wants to interrupt the read sequence or changes over to multiplex mode, Srv out and Sel out are not raised within the four clock periods. In this case the program immediately enters Tagdecode C waiting for Srv out or Cmd out to rise. When this happens Srv in is dropped and there is a test to find out which signal was raised.

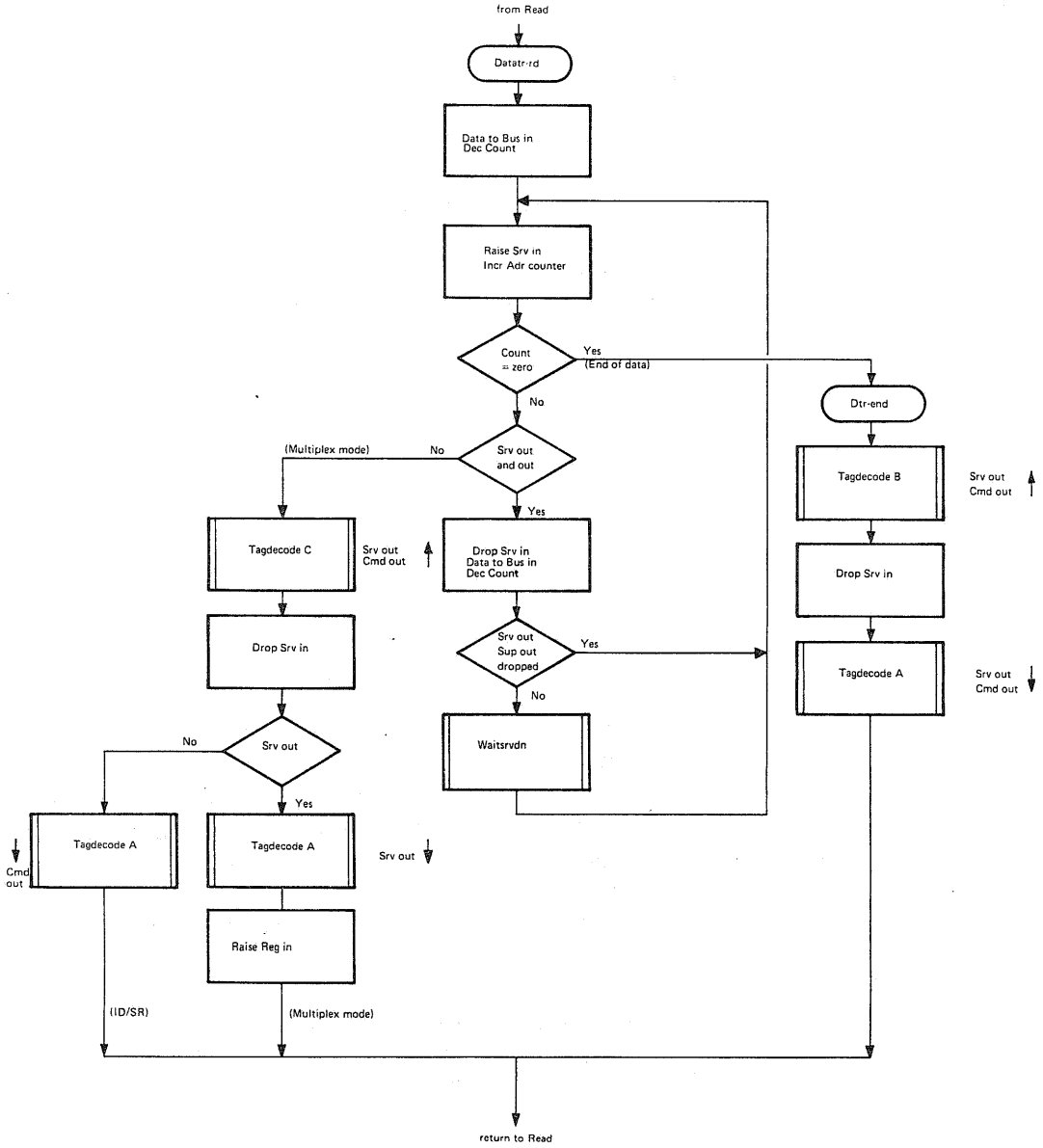


Fig. 58. Datatr-rd

If Cmd out was raised (interrupt from the channel), Tagdecode A is entered, waiting for Cmd out to drop.

If Srv out was raised (multiplex mode) Tagdecode A is entered, waiting for Srv out to drop. Then Req in to the channel is raised. When returning from Datatr-rd, Req in is tested to see if multiplex mode is selected by the channel.

When the program returns from the Datatr-rd module there is a Zero test to se if the channel has ordered ID or SR. If it has, the ID/SR module is entered. Then Req in is tested and if Req in is raised (multiplex mode) the program returns to background loop B5 (see Fig. 59).

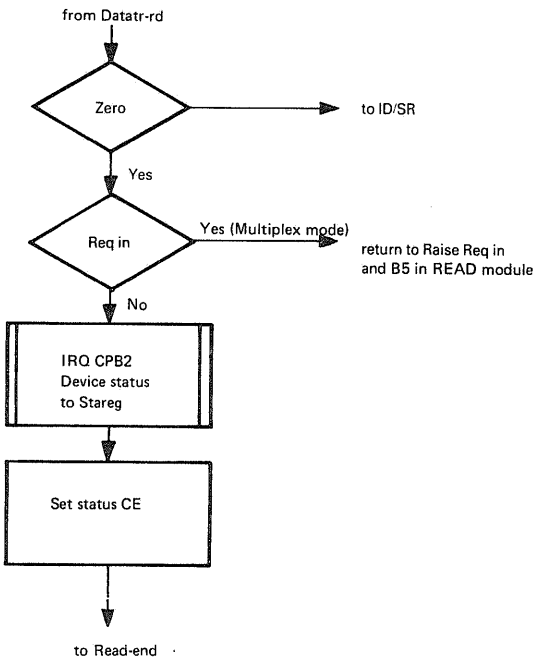


Fig. 59. Read (continued)

Finally when all data bytes have been sent, the Count registers in the microprocessor are equal to zero and the program enters the Dtr-end module. In this program module Tagdecode B is entered, waiting for Srv out or Cmd out to rise. When this happens Srv in is dropped and Tag-decode A is entered, waiting for Srv out and Cmd out to drop.

Now the program has sent all data bytes and only the status byte remains.

IRQCPB 2 transfers the status and sense bytes from the communication memory to the corresponding registers in the microprocessor (see Fig. 59). The microprocessor then adds the CE (Channel end) status to the device status in Stareg and enters the Read-end module.

Read-end

In the Read-end module, (see Fig. 60) there is first a test if the channel is operating in multiplex mode (Opl in dropped). If that is the case the Stareq bit in Pgmsta is set and Req in is raised. Then background loop B5 is entered, waiting for SEL.

If the channel is operating in burst mode the program immediately enters the subroutine Sta, where the ending device status is sent. In subroutine Sta the channel can order Interface disconnect or Selective reset. If it does the ID/SR module is entered. Otherwise the program enters End 1. Finally the program enters the background loop of the enabled state and the read sequence is ended.

Finally in this section the total microprogram design is presented in Figs 61 and 62.

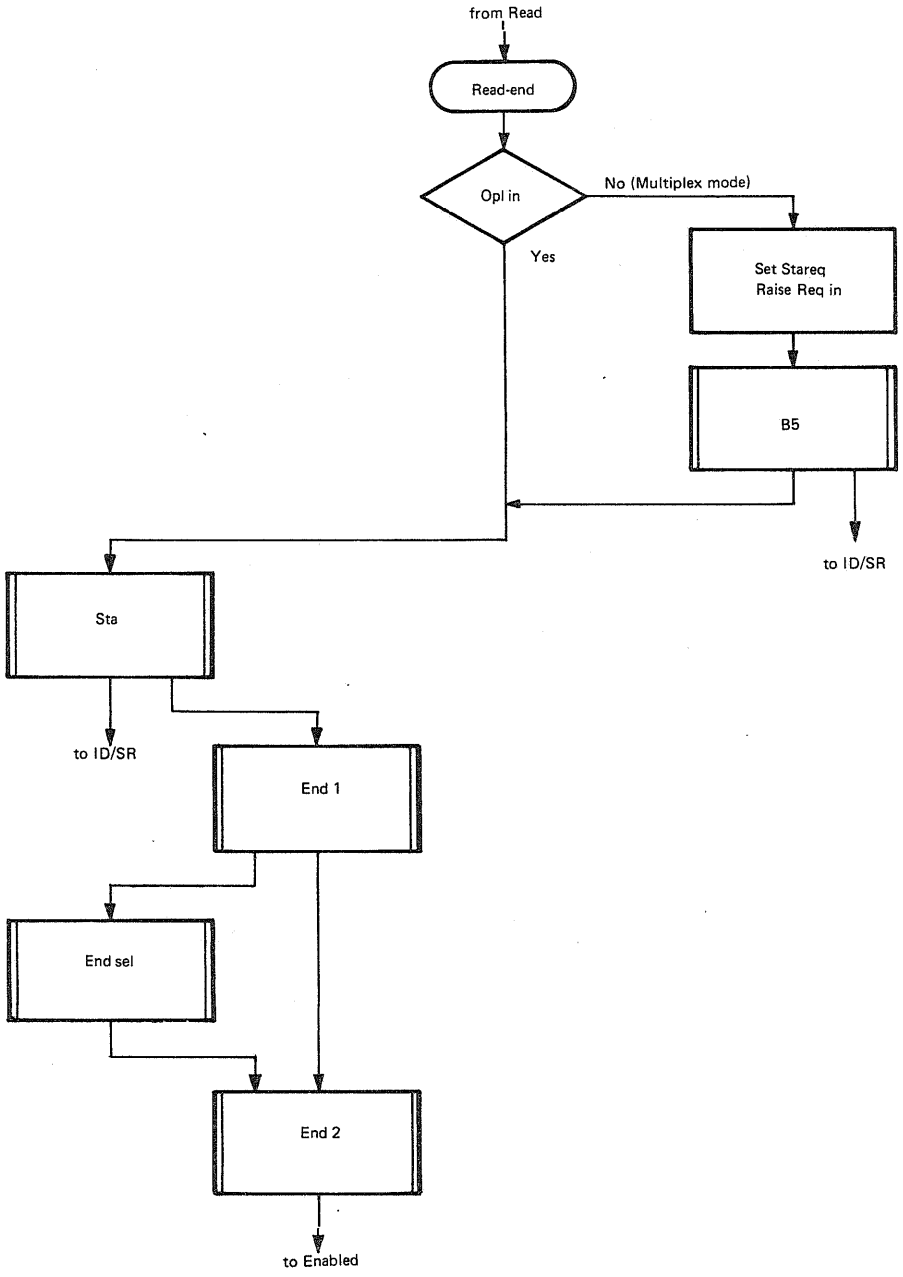


Fig. 60. Read-end

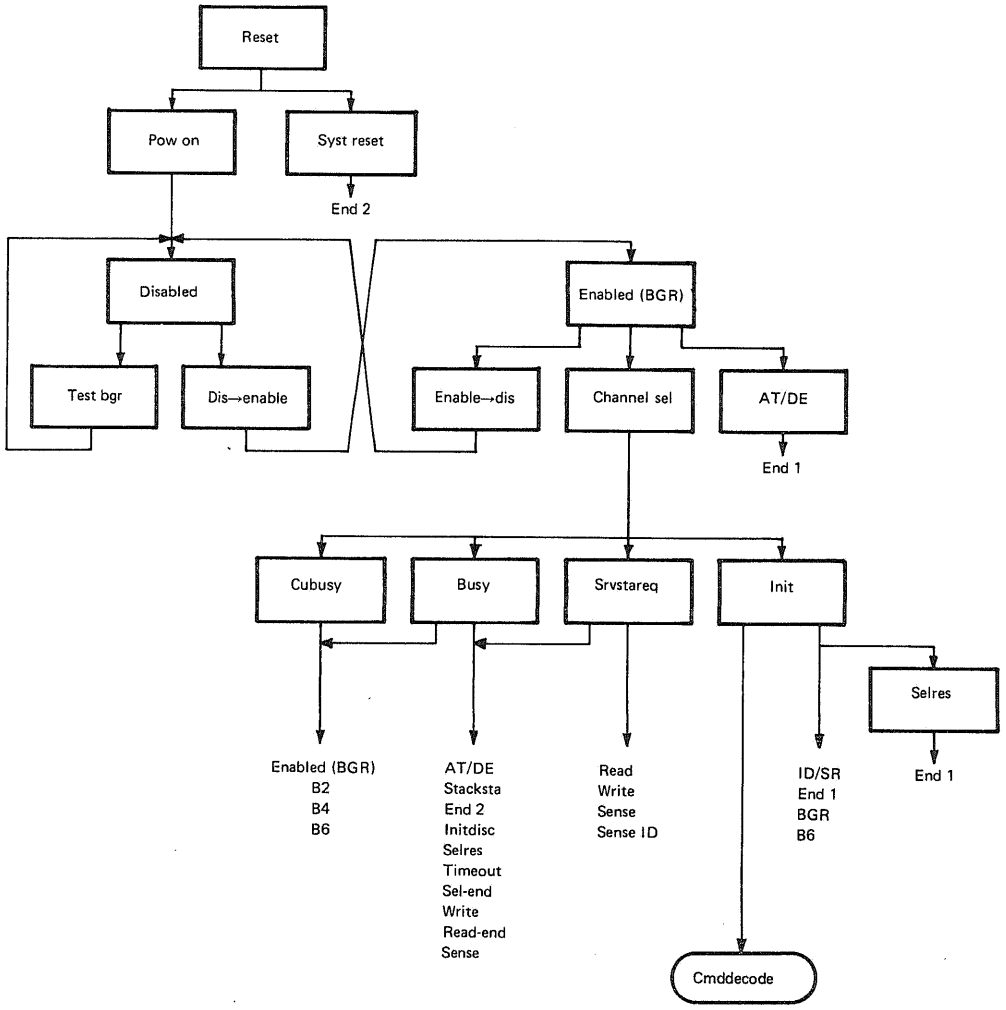


Fig. 61. Microprogram summary 1

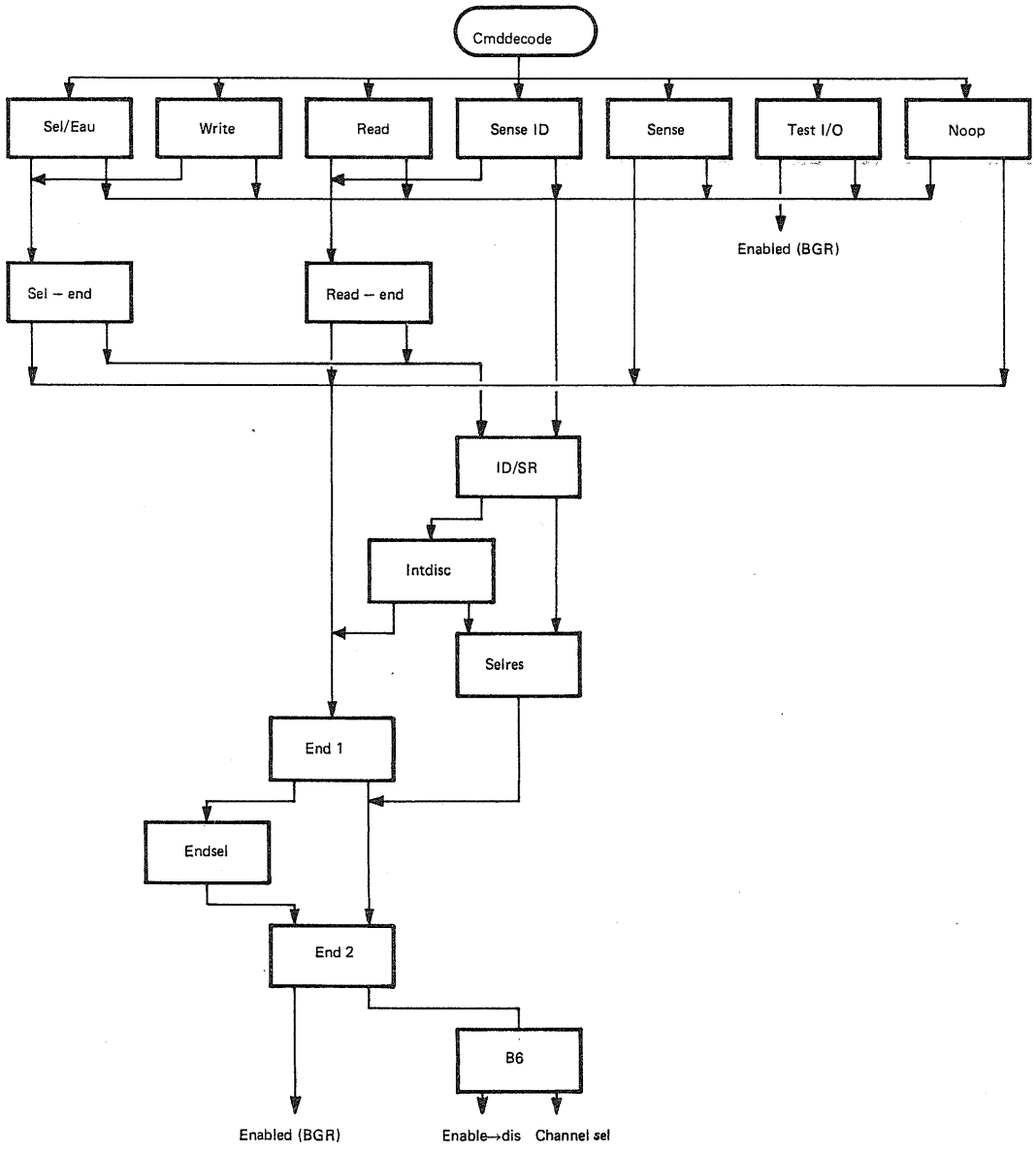


Fig. 62. Microprogram summary 2

Communication Processor, Remote, CPR 4103

Contents

General

Communication processor remote, CPR 4103, forms an interface between a host computer and a cluster of Alfaskop System 41 terminals. Both a communication processor and a flexible disk drive with necessary logic are encased in one cabinet.

The Microcomputer chapter ought to be well understood before this chapter is studied. A good orientation on the Communication chapter is also recommended; especially the Hardware part under Internal Communication via Two-wire. This chapter will in principle only treat circumstances not discussed in, or solved in other ways than said in the two chapters mentioned above.

Functions

The CPR is used when a remote connection to a host computer is wanted (e.g. via modem and telephone network) of a small cluster of up to 16 display units (and printer units) or/and flexible disk units. The CPR contains one built-in flexible disk unit.

The CPR can be used in systems where double flexible disk functions are needed, as e.g. in Alfaform, Alfaedit, or Alfabatch, only if one or more additional flexible disk units (FD 4120) are connected. Two CPRs are needed for dual host systems. The flexible disk drive can be started or stopped by the program.

The CPR:

- Carries the system software storage (on diskette).
- Handles the program loading of the connected terminals at power on.
- Regularly polls the connected terminals for status and transmission requests.
- Governs transmission to and from the host computer via modem adapters (SCA/SCC), performs editing and code transformation etc.
- Interconnects terminals in the cluster.
- Keeps trace of connected (turned on) units and which volumes (diskettes) that are inserted into the flexible disk units of the system.

Subunits

Basically the CPR can handle two or eight display units and printers. It then contains (see Fig. 1):

- CFM, CPR mechanics assembly, i.e. cabinet including front panel.
- FPS, flexible disk power supply including universal power board, UPB. See chapter Power Supplies.
- CPB, communication processor board, containing the basic micro-computer with MPU, timing logic, interrupt control logic, address decoding logic, memory access multiplexing logic (including direct

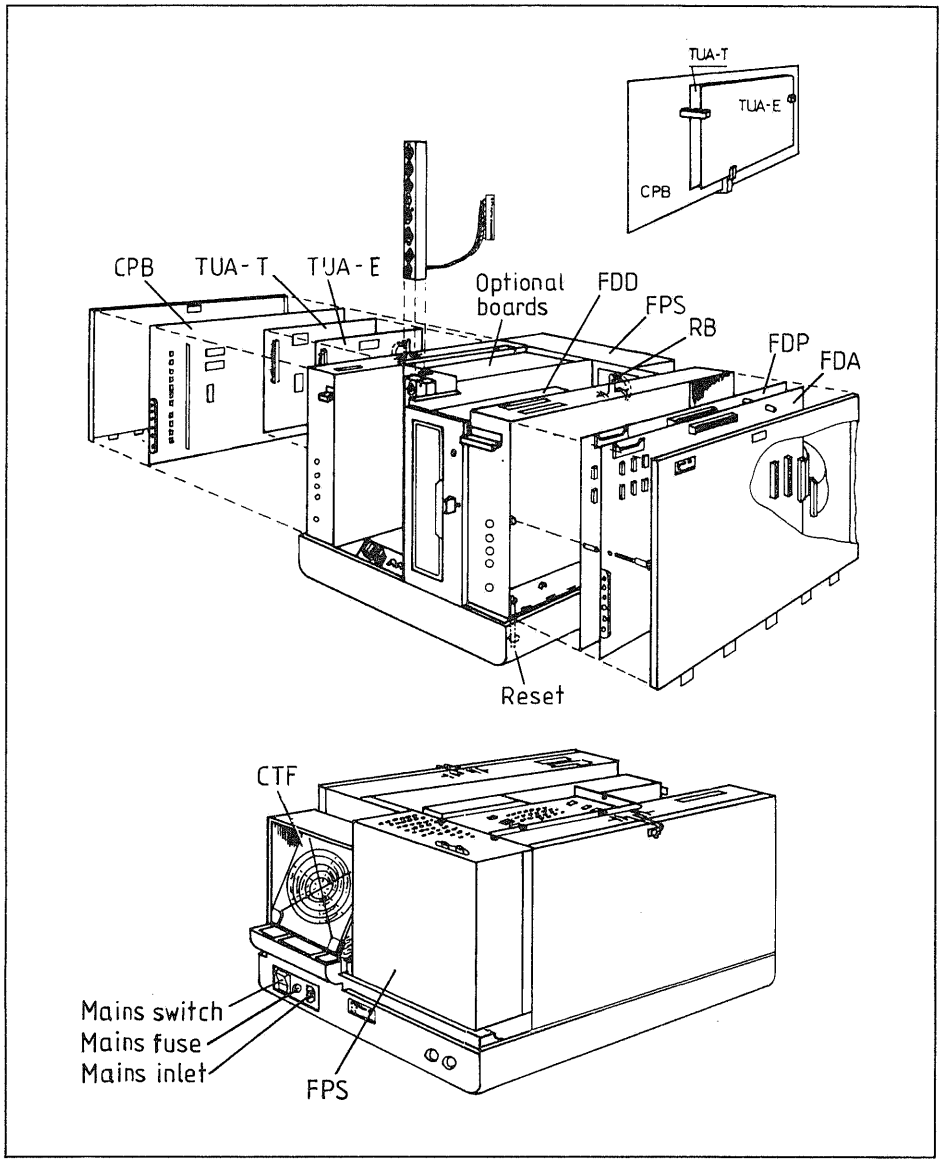


Fig. 1. CPR assembly.

memory access logic), 32 + 2 kbytes of memory, crosspoint selection logic, and (two-wire) data link controllers.

- TUA-T, terminal unit adapter T providing two or eight two-wire connections with port number 0 – 1 or 0 – 7. The flexible disk unit is internally connected to port 0. The TUA-T contains high speed modems, a crosspoint system (including line transceivers) for connection of the terminals in the cluster, and pulse transformers for the two-wires. The transfer speed is 300 kbits/s.
- FDP, flexible disk processor with MPU, timing logic, interrupt logic, address decoding logic, memory access multiplexing logic, and 32 + 4 kbytes of memory.
- FDA, flexible disk adapter with address decoding logic, memory access multiplexing logic (including direct memory access logic), data link controller, and flexible disk controller.
- FDD, flexible disk drive.
- CTF, CP and terminal fan.

Four optional boards can be inserted:

- TUA-E, terminal unit adapter E. One TUA-E can be inserted if TUA-T with eight ports is used and eight additional ports are wanted. The port numbers will be 8 – 15. The TUA-E contains the same circuits as the TUA-T except for modems (and off switches in the crosspoint system).
- MRW, MRW-A or MRW-B memory board, read/write. A memory board can be mounted if up to 29 3/4 additional kbytes of (CP) memory are wanted. See chapter Memory Board R/W.
- SCA, synchronous communication adapter which is used for communication with a host computer via modem. See chapter Synchronous Communication Adapter.
- SCC, synchronous communication controller which is used for communication with a host computer via modem. See chapter Synchronous Communication Controller.

It is not possible to use the ordinary TUA (4181) board.

The FDP board is the same as in FD 4120. The FDA board differs only in the following respects from the ordinary FDA board:

- The printed pattern has another layout.
- Only circuits, necessary for handling of one drive, are mounted.
- An additional connector for a standard flexible disk drive is mounted beside the one for the CDC flexible disk drive.
- There is a connection for a new start/stop function of the flexible disk drive.

The flexible disk part of the CPR is not described here, due to the similarity with FD 4120. Please refer to the Flexible Disk Unit chapter.

The CPR can be seen as two main functional parts. These definitions will be used in the following:

- CP. Communication processor part including CPB and TUA-T (if used also TUA-E, MRW, MRW-A, MRW-B, SCA, and SCC).
- FD. Flexible disk part including FDP, FDA and FDD. The FD will be considered as a terminal to the CP.

Mechanics

The mechanical build-up of CPR 4103 is shown in Fig. 1. The plastic cabinet is the same as for (late versions of) the CPR 4101 and the FD 4120. The main parts of the CPR, from the mechanical aspect, are:

- Box for the CPB, which has an opening for the TUA board(s).
- Rack for four optional boards. There is, however, only space for three boards when the TUA-E is used.
- Flexible disk drive, FDD.

- Box for FDP and FDA.
- Power supply.
- Fan unit.

The TUA-T board is stacked on the CPB. The TUA-E board is, if used, stacked on the TUA-T board. The optional boards are stacked on each other and connected to the CPB with a cable.

The following connectors may exist in front of the optional boards:

- 2, 8 or 16 two-wire or coaxial cable connectors.
- 2, 8 or 16 switches for connecting or disconnecting of line terminating resistors.
- Connector for V.24/V.28 (25 pins).
- Connector for X21 (15 pins).

A relay board, RB, for the start and stop function of the flexible disk drive is mounted behind the flexible disk drive.

Brief Outline

Monitoring

The CPR front panel shows eight lamps. See circles in Fig. 2. The two to the left are mounted on the CPB:

- Ready lamp is governed by the CP program. It blinks when the CP has started an internal poll in order to find the system diskette and glows steadily when the CP has loaded the operating system in its read/write memory.
- Line 1 lamp is governed by a line activity signal from the SCA (or SCC). It glows steadily if data is regularly sent to and blinks if data is only received from the host computer. (For details see chapters on Synchronous Communication Adapter or Synchronous Communication Controller.)

The five lamps to the right are mounted on the FDA board (and controlled by the FD program except for the Power on lamp):

- Ready lamp glows steadily when the flexible disk processor has finished the initial program loading. The lamp blinks if the CP does not scan the FD.
- Status 1, 2, and 3 lamps indicate the status of the flexible disk drive. The interpretation is given in the Installation and Maintenance Manual.
- Power on lamp is directly governed by the +5 V power.

The lamp on the flexible disk front indicates whether the flexible disk door can be opened or not.

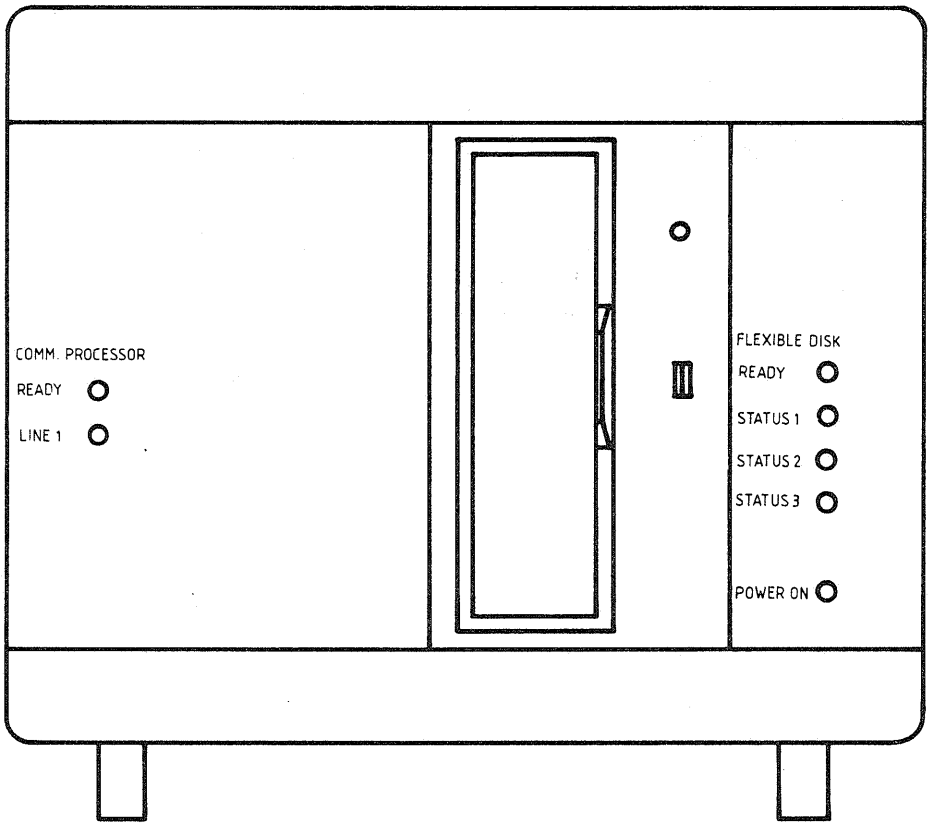


Fig. 2. Front of CPR

Memory Map

Fig. 3 presents the memory map of the CP.

The memory areas are described in the Microcomputer chapter, circumstances pertaining to more than 32 kbytes of memory in the Memory Board R/W chapter, and the interrupts under the following heading. The I/O addresses are specified in Appendix 1.

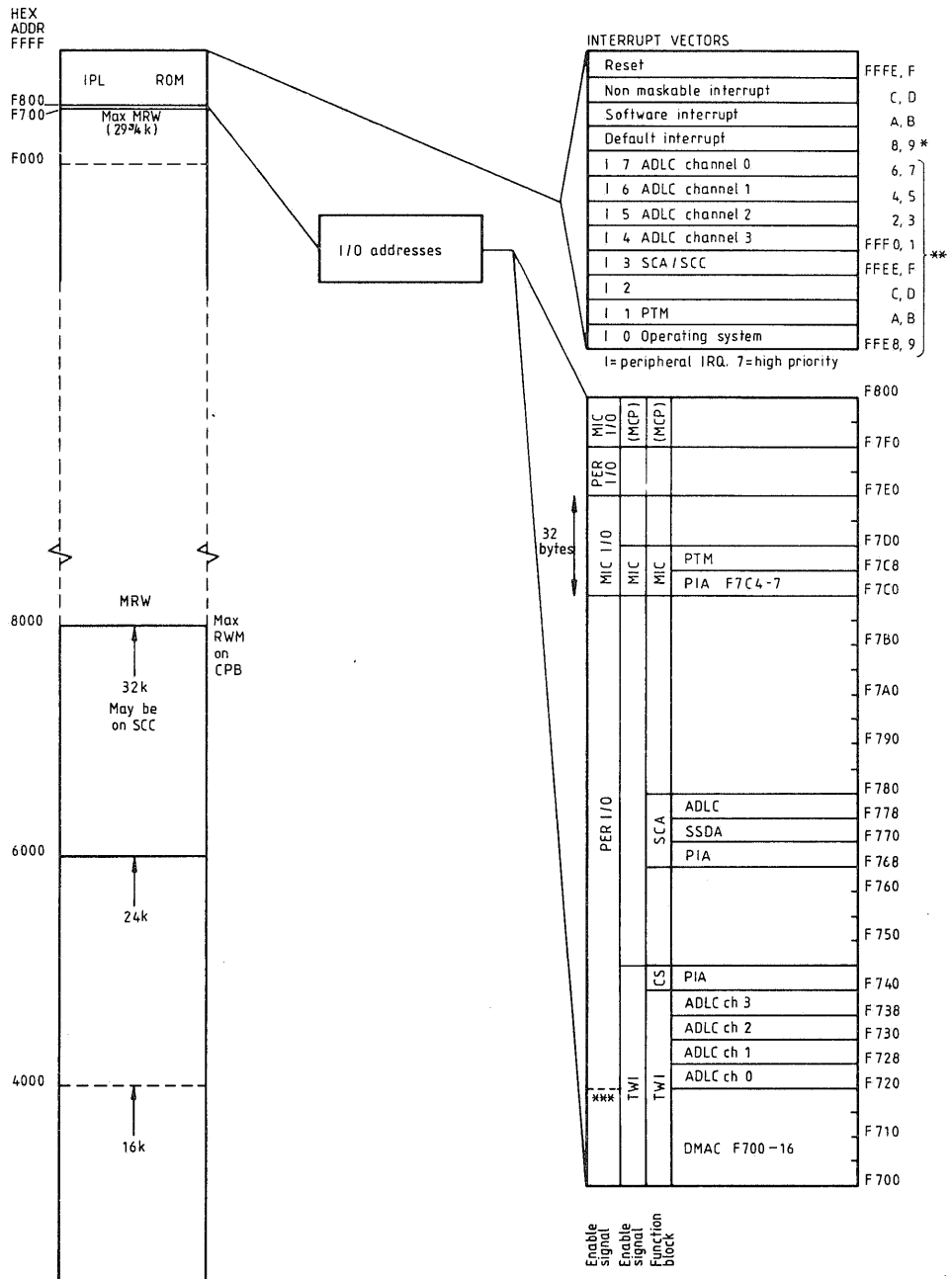
Communication with a host computer may proceed via circuits on an SCA at addresses $F768_{(16)} - F77F_{(16)}$.

If an SCC is used, the microcomputer of the CPB reads data from and writes data to the host computer in a dual access area on the SCC. The addresses $6000_{(16)} - 7FFF_{(16)}$ are then reserved for this purpose and the memory on the CPB must in this case be limited to 24 kbytes by strapping. Strapping information is found in the Installation and Maintenance Manual. The SCC handles certain message editing and the actual transfers between the dual access area and an interface circuit (SSDA or ADLC) on the SCC.

Interrupts

The principles of the interrupt system are described in the Microcomputer chapter. The locations of the interrupt vectors (i.e. the addresses to the interrupt routines) are found in the memory map.

Here follows a description of the origin, properties and use of the different CP interrupt signals.



ADLC = Advanced data link controller
CPB = Communication processor board
CS = Crosspoint selection
DMAC = Direct memory access controller
IPL = Initial program load
MCP = Microprocessor control panel
MIC (I/O) = Microcomputer, central part (input/output)
MRW = Memory board read/write
PER I/O = Peripheral input/output
PIA = Peripheral interface adapter
PTM = Programmable timer module
SCA = Synchronous communication adapter
SSDA = Synchronous serial data adapter
TWI = Two-wire interface

* MPU reading FFF8 – FFF9 will transfer the contents of two cells within FFE8 – FFF9, depending on highest priority valid interrupt, to the MPU.

** A (read or preferably) write operation into one of the cells FFE8 – FFF7 will result in a setting of a corresponding mask value. As an example, FFEE or FFEF **from the MPU** will disable interrupts of priority < 3.

*** Not present during the three DMA cycles DMA T1 – T3.

Fig. 3. CP of CPR 4103 memory map.

Interrupts out

Two interrupt signals can be generated and fed to subunits. Interrupt slave I is generated when bit 5 is set in the peripheral register A of the MIC PIA. Interrupt slave II is generated when bit 3 is set in the same register.

Reset, NMI, and SWI

The MPU is reset at power on or at depression of the Reset button. The MPU will after a reset fetch the contents of the cells $FFFE_{(16)}$ and $FFFF_{(16)}$ to the program counter and execute the reset program (see also Reset functions).

NMI, non-maskable interrupts may be generated from the test unit MCP (Microprocessor control panel) or when a parity error is detected at readouts from the read/write memory. The program can test the source of the interrupt by reading the peripheral register A of the MIC PIA and test bits 0 and 2 (see Microcomputer PIA, MIC PIA).

SWI, software interrupt is generated by an MPU instruction.

Interrupt 7 – 0 and Default Interrupt

Interrupts 7 – 0 (I7 – 0) are set in the interrupt register. The interrupts are arranged in a priority order where I7 has the highest priority. A set interrupt must be of the same as or higher priority level than denoted by the present value of the mask register in order to be valid. The interrupt vector is fetched from the cells $FFF8_{(16)}$ and $FFF9_{(16)}$ (default interrupt) if a triggering interrupt is not valid at the time of the vector fetch.

The maskable interrupts, I7 – 0, are further only serviced if the I-bit of the MPU condition code register is reset. I7 – 0 originate from the following sources:

I7 originates from the ADLC (advanced data link controller) of channel 0.

I6 originates from the ADLC of channel 1.

I5 originates from the ADLC of channel 2.

I4 originates from the ADLC of channel 3.

I3 may originate from a non-steady source (short pulse interrupt). It will be latched by the interrupt logic if bit 4 of the MIC PIA peripheral register A is set. Any latched interrupt is reset and the state of the I3 line to the interrupt register will be equal to the state of the incoming interrupt line if bit 4 is reset. The I3 line is used for instance by an SCA or an SCC.

I2 functions as I3 but is controlled by bit 6 of the MIC PIA peripheral register A.

I1 originates from the PTM.

I0 originates from the CA2 output of the MIC PIA. The program can thus make an interrupt by writing $110_{(2)}$ in bits 5 – 3 of the MIC PIA control register A.

Programmable timer module, PTM

A programmable timer module, PTM, is used to provide two or three time bases for generation of timeouts and a real time clock (calendar clock) in the CP. A detailed description of the PTM circuit is found in the Microcomputer chapter.

The PTM contain three timers with 16-bit counters. Timer 1 is typically programmed to divide the basic clock (∅2) by 21 300 and thus make an interrupt (I1) every 20:th ms (50 Hz). The output from timer 1 is fed to the clock input of timer 2 and there typically divided by 50. Timer 2 will thus make an interrupt (I1) every second. Timer 3 is a spare timer, which for instance may be used for short time measurements.

Terminal Communication

General

The CP communicates with the terminals in the cluster via four ADLC (advanced data link controller) circuits, four high speed modems, and a crosspoint system. For a description of the used communication procedure and a survey of the hardware functions please refer to the Communication chapter.

Each one of the ADLCs forms a part of one of four channels, interfacing a high speed modem and the crosspoint system on the TUA-T board. The TUA-T board comprises two or eight two-wire connections. An optional TUA-E board may be used, providing eight additional two-wire connections. See Fig. 4. Each two-wire may be chained to a display unit (and a printer unit) or/and a flexible disk unit. Note that the demands on maximum response times and number of terminal addresses in different applications introduce restrictions in the number of connected terminals.

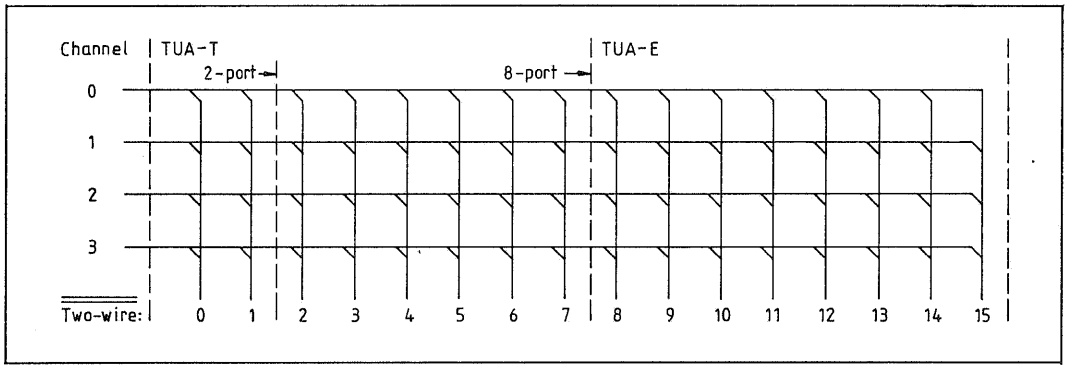


Fig. 4. Crosspoint system.

A certain channel may be connected to any one and any number of two-wires in the crosspoint system. Transfers may also take place on more than one channel simultaneously. Note, however, that a certain two-wire should not be connected to more than one channel at the same time.

The ADLCs are controlled by the MPU or by the direct memory access logic, including a DMAC (direct memory access controller) circuit. The

crosspoint system is controlled by a crosspoint selection PIA (peripheral interface adapter) and a crosspoint selection logic. See block diagram.

Direct Memory Access

The DMAC (see Microcomputer chapter) control of the four channels, means that the DMAC registers associated with a certain channel must be used both for read memory (transmit data) and write memory (receive data) access sequences. As suggested above, more than one channel may be enabled simultaneously. The direct memory access timing (including DMAC), ensures that only one channel transfers a byte at a certain time.

An ADLC may thus be accessed in two ways (see block diagram):

- Addressed by the MPU; CE ADLC provided by the two-wire interface address decoder, R/W and address lines for register selection provided by the MPU. All ADLC registers can be accessed.
- Addressed by the DMAC during DMA T2 (DMA mode); CE ADLC provided as DMAC encoded channel address (TXAKA and TXAKB) directed DMA mode signal to the proper ADLC, R/W and one address line provided by the DMAC. Receiver and transmitter FIFOs can only be accessed as the other address line (A1) is tied high.

Channel Usage

The four channels are used in the following way:

- Channel 0 is used for internal polling, i.e. to investigate if the connected terminals want to communicate with the host computer or transfer data to or from another terminal in the cluster. (Note that the FD is considered as a terminal.)
- Channel 1 is used for connections between the CP and the terminals in the cluster, e.g. for initial program loading of the CP or connecting sequences. It can also be used for terminal to terminal communication (see Channel 3 below).
- Channel 2 is used for traffic between the terminals and the host computer. As an example, say that the CP while polling has found that a number of display units want to send messages to the host computer. When the CP then is polled (or a terminal selected) by the host computer the CP connects the display units, one after the other on channel 2. The actual message is then via the CP stored in the line buffer and by the SCA (or SCC) transferred to the host computer. The SCA handler program takes care of the data in the line buffer, makes necessary code transformations, and sends the data.
- Channel 3 is used for internal traffic (terminal to terminal communication). A display unit on one two-wire may e.g. answer to a poll (on channel 0) that it wants to make a local printout on a printer unit connected to a display unit on another two-wire. In case this printer unit is not busy and no other units are queuing to use the printer unit, the CP connects the requesting display unit with the printer unit on channel 3. Note that the ADLC 3 (in the CP) will still be listening to the traffic. Thus, the CP program may know when the transfer is completed.

Turning Crosspoints On and Off

Each crosspoint consists of two SCRs (silicon controlled rectifiers or thyristors). To connect one two-wire to one channel a selection address and an On order are written in the crosspoint selection PIA. Thereby one TUA enable signal, one two-wire enable signal (Tw en) and one Channel select (Ch sel) signal are activated, making the selected pair of SCRs conducting.

The selection signals disappear when the On order is cancelled via the PIA. The SCRs require hold currents in order to stay on. These hold currents are provided as long as the Channel off (Ch off) signal is not activated. The Channel off signal will, when activated, turn off all crosspoints connected to one (selected) channel.

Thus:

- When a selection word and an On order are written into the crosspoint selection PIA, only one pair of SCRs (one crosspoint) is activated. To connect more two-wires to the channel, one selection must be made for each crosspoint.
- A certain crosspoint may not be disconnected separately – all crosspoints of one channel will be disconnected at the same time.

Crosspoint Signals

There are two physical connections (for each channel) through the crosspoint switch, see block diagram. One is used for:

- Modulated received data (transferred as current) and TUA Request to send (TUA RTS, transferred as voltage).

The other is used for:

- Modulated transmitted data (transferred as voltage).

Modulated transmitted data is provided by the high speed modem of the channel in question and may originate from:

- The ADLC (i.e. from the CP memory).
- A terminal connected to the channel, that sends data to the high speed modem on the Modulated received data line (transferred on the same line as TUA RTS trough the crosspoint switch).

TUA RTS is generated in the modem when:

- The ADLC signals RTS, set by the CP program before the CP starts transmitting to the terminal(s) connected to the channel, or:
- The high speed modem receives valid Modulated received data.

Note that the modem generates TUA RTS and Modulated transmitted data not only when the CP is transmitting but also when some terminal is sending data into the crosspoint system. The two-wire line transceiver (on the TUA) connected to the terminal sending into the CP will be in the receiving state, i.e. not affected by the TUA RTS signal occuring on the

channel – the line transceivers of all other two-wires connected to the channel will, however, be in transmitting state.

This enables (in addition to ordinary transfers):

- Transfers of messages between terminals in the cluster.
- Connections of display units listening to the traffic on the channel.
- The CP to keep track of terminal to terminal communications.

Example of Data Transfers

There are thus three communication modes as shown in Fig. 5.

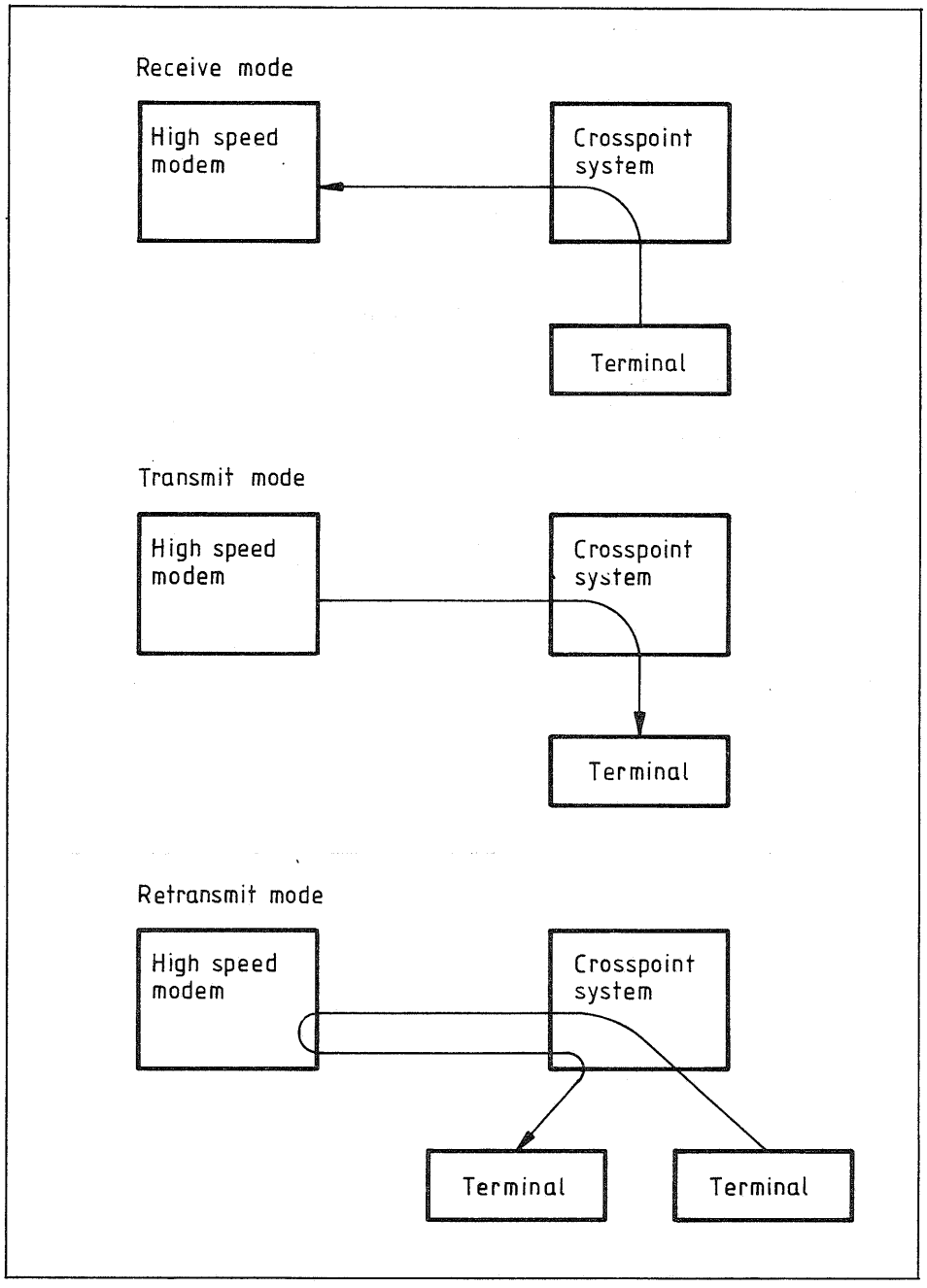


Fig. 5. Communication modes.

Fig. 6, an excerpt from the block diagram, exemplifies a data transfer situation in the crosspoint system of an 8-port TUA-T:

- One byte is transferred from the memory of the CP to the ADLC of channel 0 under the control of the DMAC circuit.
- A previously transferred byte of a poll message is shifted out from the ADLC of channel 0, modulated, and transmitted to a terminal connected to two-wire number 7.
- Data is transferred from a terminal connected to two-wire 0 to another terminal on two-wire 4. Note that:
  - Channel 3 is used.
  - Data proceeds via the high speed modem.
  - Data is accessible for the microcomputer of the CP (in ADLC 3).

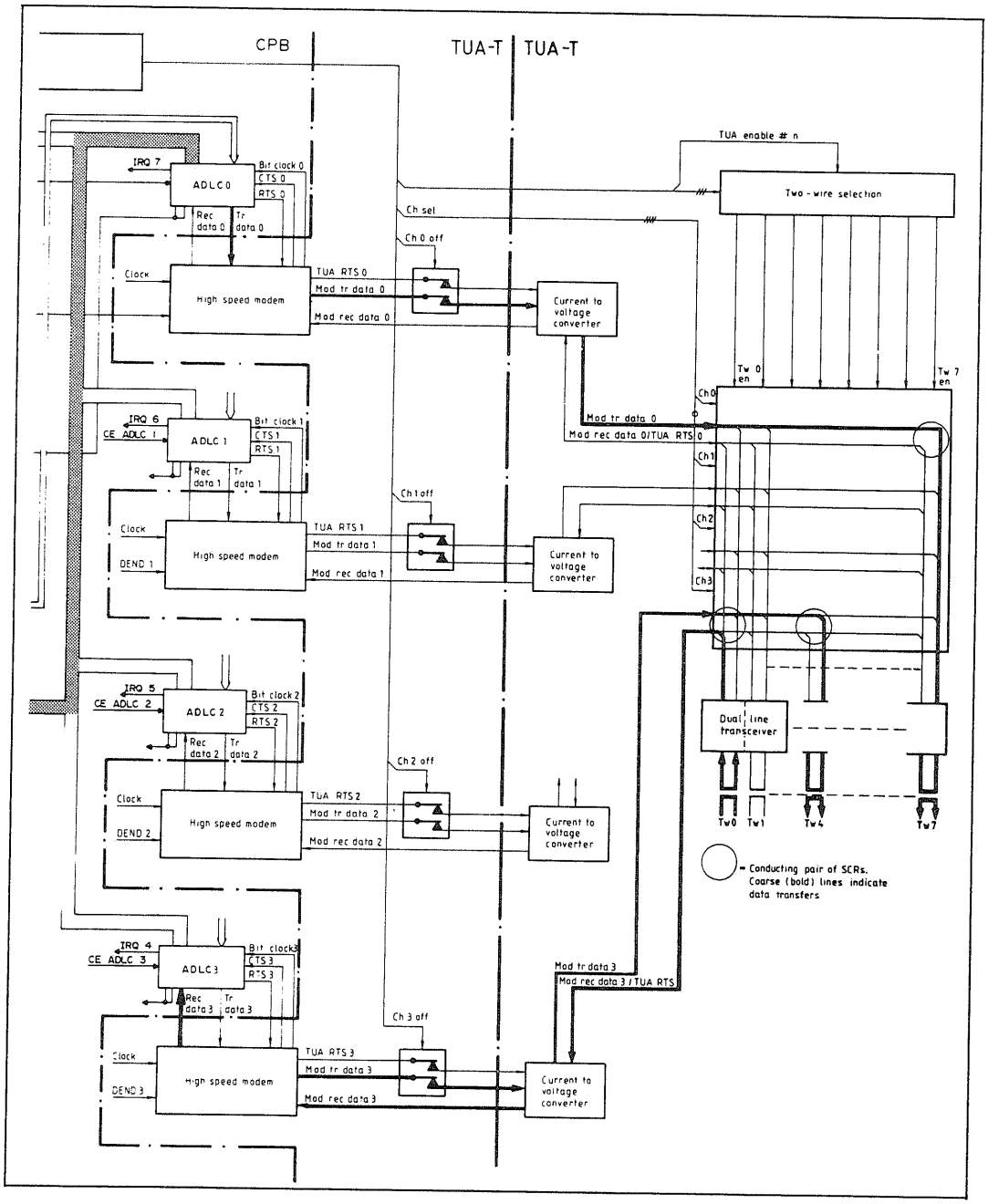


Fig. 6. Cluster data transfer example.

Detailed Description

This section will only cover more complicated circuits or special solutions.

Microcomputer PIA, MIC PIA

Input/Output Functions

The MIC PIA controls several central functions. See chapter Microcomputer; Peripheral Interface Adapter, PIA for definitions and a general survey. IRQA, CA1, IRQB, CB2, PB7, 5, and 4 are not used.

The CA2 output is used to generate interrupt 0 if CRA bit 3 is zero-set (CRA bits 5, 4 = $11_{(2)}$).

The MIC PIA interrupt flag (CRB bit 7) is set if the Reset button is depressed (as CB1 is then taken low). This must be checked by the program as IRQB is not connected.

The PA outputs/inputs are used in the following way (opposite polarity means "do not" or "not"):

PA7	Input	Hard-wired low
PA6	Output	1 = latch interrupt 2
PA5	Output	1 = interrupt slave I
PA4	Output	1 = latch interrupt 3
PA3	Output	1 = interrupt slave II
PA2	Input	0 = MCP connected and in test mode
PA1	Output	0 = lit Ready LED
PA0	Input	1 = parity error FF is set

The PB outputs are used in the following way:

PB6	Output	1 = activate general Reset. The output must be set low after power on. (The MPU and the MIC PIA are not reset by general Reset.)
PB0	Output	1 = reset and keep parity error FF reset.

PB3-1 are not used today. PB3 ought to and PB2-1 must, however, be low.

Reset Functions

The MPU is reset by a power on reset or a Reset button depression. The MIC PIA is only reset by the power on reset. The general Reset is activated at power on and stays active until MIC PIA PB6 is set low by the MPU.

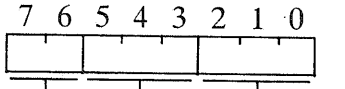
The general Reset will not be activated at a Reset button depression. The MPU, however, will check the MIC PIA interrupt flag (CRB bit 7) and find that the button has been depressed. The MPU can then make a general reset by setting MIC PIA PB6 first high and then low.

Crosspoint Selection Control

Crosspoint Control Signals

The control signals to the crosspoint system are generated in a crosspoint selection PIA (peripheral interface adapter, see Microcomputer chapter for a general description) and distributed via a decoder and a selector on the CPB (crosspoint selection logic on the block diagram) and a selector on each TUA board.

The PIA generates an On signal from CB2 and a Channel off (Ch off) signal from CA2. All of peripheral register B is used as output with the following function:



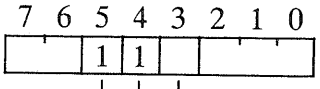
Two-wire address (Tw addr) decoded to one of eight two-wire enable (Tw en) signals when applicable TUA enable is present. $000_{(2)}$ = Tw en 0. $111_{(2)}$ = Tw en 7.

TUA select. When On is active these bits are decoded to one of five TUA enable signals. TUA enable 0 (TUA select = $000_{(2)}$) enables the decoding of Tw en 7 – 0 on the TUA-T. TUA enable 1 (TUA select = $001_{(2)}$) enables the decoding of Tw en 7 – 0 on the TUA-E. TUA enable 2 – 7 are not used.

Channel select ($00_{(2)}$ = channel 0, $11_{(2)}$ = channel 3). These bits are used to control the distribution of On (called Ch sel 3 – 0 when fed to the TUAs) and Ch off to applicable channel.

Crosspoint Selection PIA Control

In order to get the above mentioned functions the control register A (CRA) of the crosspoint selection PIA must be set as follows:

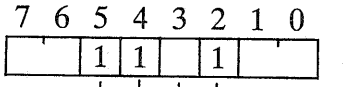


CA2 output (Ch off) follows what is written into this bit (if bits 5 – 4 are ones). 0 = selected channel is turned off. 1 = connections may be made on all channels.

CA2 follows CRA bit 3.

CA2 functions as output.

Control register B (CRB) should contain:



Peripheral register B accessible. This bit must of course be 0 at the initiation when the Data direction register B is set to all ones.

CB2 output (On) follows what is written in this bit (if bits 5 – 4 are ones).

CB2 follows CRB bit 3.

CB2 functions as output.

TUA Functions

The TUA functions are performed by means of four high speed modems and on each one of TUA-T (8-port) and TUA-E; a crosspoint system, eight line transformers, and a decoder. The crosspoint system consists of four off switches, four current to voltage converters, two 4 × 4 × 2 crosspoint switches, and four dual line transceivers. The decoder decodes the two-wire address (from the crosspoint selection PIA) when enabled by TUA enable.

The TUA-T with two ports is funcionally the same as the eight port version, but is not fully equipped. It differs from what is said above in that only one 4 × 4 × 2 crosspoint switch, one dual line transceiver and two line transformers are mounted. Neither is each one of the four current to voltage converters fully equipped.

High Speed Modem

The high speed digital modem consist of one chip, which contains a modulator, a demodulator, a frequency divide and phase correction logic, and a delay circuit for Clear to send. See Fig. 7.

Data to be transmitted (Tr data, from the ADLC) is presented in serial, not modulated, format to the modulator. The modulator converts the signal to a frequency shifted signal (Mod tr data) which then eventually may be fed to the two-wire(s).

The frequency shifted data transferred from the two-wire (Mod rec data) is fed to the demodulator and to the frequency divide and phase correction logic. An internal strobe from the latter logic shifts the Modulated received data into the demodulator and generates the Bit clock, fed to the ADLC. The demodulator converts the received data to demodulated valid data (Rec data, fed to the ADLC). Received data is (as Retransmit is hard-wired low) internally also fed to the modulator and retransmitted (with a delay of ~ 2 bit times) to the line (Mod tr data).

The modulation method, clock generation and phase correction are described in the Communication chapter.

The Request to send output (TUA RTS) follows (but inverted) the Request to send input (RTS) in the transmit mode and is a function of Modulated received data in the retransmit mode.

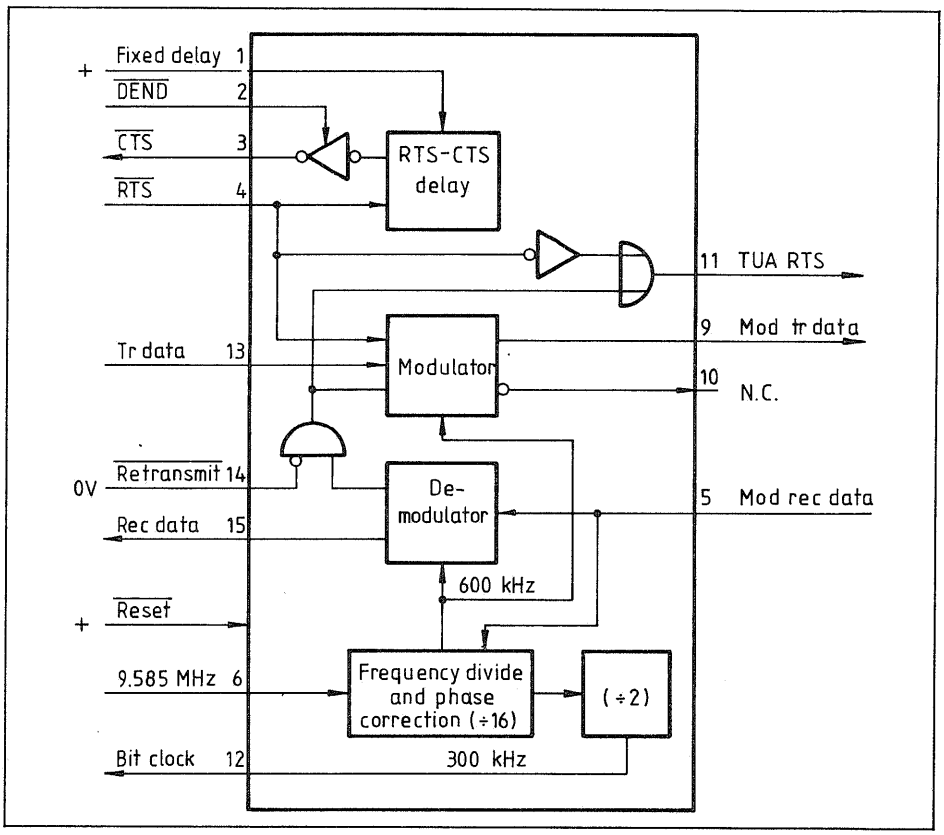
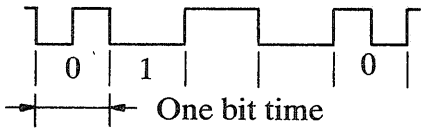


Fig. 7. High speed modem circuit.

Clear to send (CTS) follows Request to send input (RTS) with a delay of 32 (controlled by strapping) bit clock periods. This delay will enable the modem in the receiving unit (e.g. in a DU) to synchronize before the transmission is started. A frequency corresponding to the state of the Transmitted data input will be sent on Modulated transmitted data during the delay. An active DEND signal (at the end of a transmission sequence) from the DMAC (see Microcomputer chapter) will deactivate Clear to send.

Modulated received data is validated i.e. each bit must consist of an LH, HL, LL or HH and there must be a shift of polarity between the bits in order to be accepted:



Received data is always high when Modulated received data is invalid or before the modem has synchronized with Modulated received data. TUA RTS is only activated for valid data at retransmission.

High Speed Modem/Two-wire Interface

Fig. 8 shows the interface between one channel (one high speed modem) and one two-wire. It consists of an off switch, a current to voltage converter, an SCR pair in the crosspoint switch, a line transceiver, and a line transformer.

The connection through the crosspoint switch will be established when the Two-wire enable (Tw en) and Channel select (Ch sel) signals are activated, i.e. when the encircled double collector transistor in Fig. 8 is turned on (if Channel off [Ch off] is inactive). The SCRs will then be kept on by the holding current between the ~ +10 V to RC1 and RC2 in the current to voltage converter and the −5 V of the current sources in the line transceivers. The SCRs are turned off when the bases of the transistors T1 and T2 in the current to voltage converter are sunk to ~ −5 V by an activated Channel off signal. (No one of Modulated transmitted data, TUA RTS, and Modulated received data can then be transferred through the crosspoint system.)

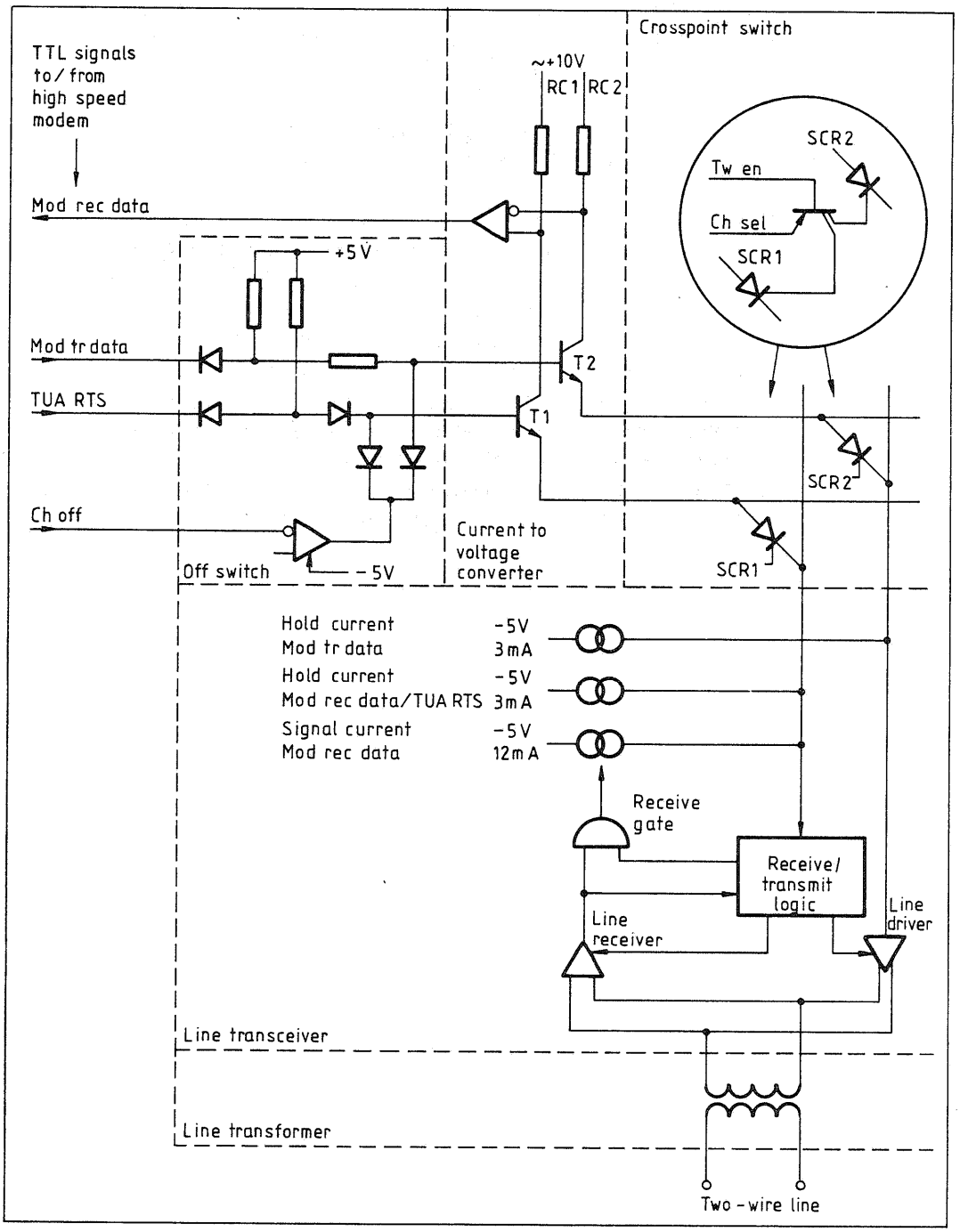


Fig. 8. Principles of high speed modem/two-wire interface.

When data is received (receive mode) from the line, the signal is converted to a current signal in the line transceiver. The output from a receive gate turns an additional current source (12 mA) on and off. The current signal (3 mA or 15 mA) flows through the SCR1, T1, and RC1. The voltage level at the line transceiver pin is not affected by these signal currents and is always ~ 1.5 V below the high speed modem output TUA RTS. The signal voltage created across RC1 is converted to a TTL level by a comparator and fed to the high speed modem as Modulated received data.

When sending data to the line (transmit mode), the high speed modem raises the TUA RTS output to a high level (+4 V; the signal is then dropped ~ 1.5 V by T1 and SCR1). TUA RTS high will enable the line driver and disable the line receiver in the line transceiver (unless the transceiver is receiving). Modulated transmitted data from the high speed modem flows through T2 and SCR2 (and is also dropped ~ 1.5 V). The currents through the SCR1, SCR2, T1, T2, RC1, and RC2 are not affected by the different voltage outputs from T1 and T2 because the two inputs of the line transceiver are of high impedance type.

The high speed modem will always refresh and retransmit the received signal to the crosspoint switch at the high speed modem output Modulated transmitted data. The high speed modem output TUA RTS will also be active and the two output signals are sent to all connected line transceivers, even the line transceiver which receives the original signal from the line, but the line driver in this circuit is disabled by the incoming signal from the two-wire. Note that at least one more than the receiving crosspoint must be connected in order to obtain an actual retransmission (in retransmit mode). Note also that Modulated transmitted data and TUA RTS are transferred through the crosspoint system at the same time as Modulated received data is transferred through it in the opposite direction.

Appendix 1

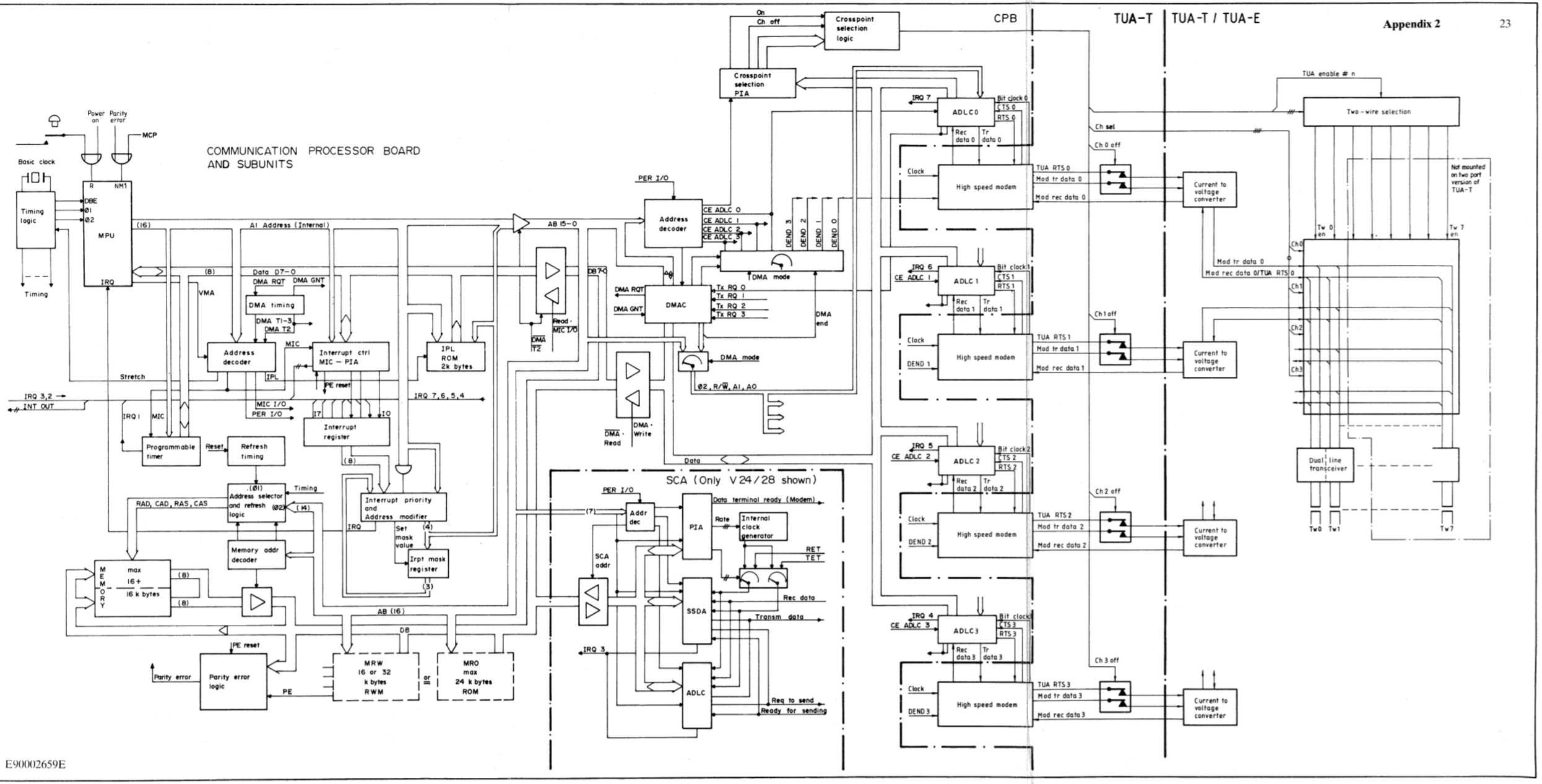
I/O Addresses, $F7FF_{(16)} - F700_{(16)}$

(The first two figures, $F7_{(16)}$, are omitted in the address column below. R/W = W means write, R/W = R means read and R/W = R/W means read or write as seen from the MPU.)

Address (Hex)	R/W	Other condition	Addressed unit and register
			MCP
FF	W		= F7
FE	W		= F6
FD	W		= F5
FC	W		= F4
FB	R		= F3
FA	R		= F2
F9	R		= F1
F8	R		= F0
F7	W		STORE (bit 7), CCR, ACCB, ACCA, XHI, XLO, PCHI and PCLO switches
F6	W		Data 7–0 switches
F5	W		Address 15–8 switches
F4	W		Address 7–0 switches
F3	R		A, B, C, D, NMI, IRQ, VMA and R/W LEDs
F2	R		Data 7–0 LEDs
F1	R		Address 15–8 LEDs
F0	R		Address 7–0 LEDs
EF–D0			**Not used**
			PTM
CF	W		Timer 3 LSB latches
CF	R		LSB buffer (timer 3)
CE	W		MSB buffer (timer 3)
CE	R		Timer 3 counter MSB
CD	W		Timer 2 LSB latches
CD	R		LSB buffer (timer 2)
CC	W		MSB buffer (timer 2)
CC	R		Timer 2 counter MSB
CB	W		Timer 1 LSB latches
CB	R		LSB buffer (timer 1)
CA	W		MSB buffer (timer 1)
CA	R		Timer 1 counter MSB
C9	W		Control register 2 (CR2)
C9	R		Status register
C8	W	CR2:0 = 0	Control register 3
C8	W	CR2:0 = 1	Control register 1
			MIC PIA
C7	R/W		Control register B (CRB)
C6	R/W	CRB2 = 0	Data direction register B
C6	R/W	CRB2 = 1	Peripheral register B
C5	R/W		Control register A (CRA)
C4	R/W	CRA2 = 0	Data direction register A
C4	R/W	CRA2 = 1	Peripheral register A
C3–80			**Not used**

Address (Hex)	R/W	Other condition	Addressed unit and register
			SCA ADLC
7F	→	→	= 7B
7E	→		= 7A
7D	→	→	= 79
7C	→		= 78
7B	W	CR1:0 = 0	Transmitter FIFO (terminate)
7B	W	CR1:0 = 1	Control register 4
7B	R		Receiver FIFO
7A	W		Transmitter FIFO (continue)
7A	R		Receiver FIFO
79	W	CR1:0 = 0	Control register 2
79	W	CR1:0 = 1	Control register 3
79	R		Status register 2
78	W		Control register 1 (CR1)
78	R		Status register 1
			SCA SSDA
77	→	→	= 71
76	→		= 70
75	→	→	= 71
74	→		= 70
73	→	→	= 71
72	→		= 70
71	W	CR1:7,6=00	Control register 2
71	W	CR1:7,6=01	Control register 3
71	W	CR1:7,6=10	Synchronization code register
71	W	CR1:7,6=11	Transmitted data FIFO
71	R		Received data FIFO
70	W		Control register 1 (CR1)
70	R		Status register
			SCA PIA
6F	R/W		= 6B
6E	R/W	→	= 6A
6D	R/W		= 69
6C	R/W	→	= 68
6B	R/W		Control register B (CRB)
6A	R/W	CRB2 = 0	Data direction register B
6A	R/W	CRB2 = 1	Peripheral register B
69	R/W		Control register A (CRA)
68	R/W	CRA2 = 0	Data direction register A
68	R/W	CRA2 = 1	Peripheral register A
67–48			**Not used**
			CS PIA
47	R/W		= 43
46	R/W	→	= 42
45	R/W		= 41
44	R/W	→	= 40
43	R/W		Control register B (CRB)
42	R/W	CRB2 = 0	Data direction register B
42	R/W	CRB2 = 1	Peripheral register B
41	R/W		Control register A (CRA)
40	R/W	CRA2 = 0	Data direction register A
40	R/W	CRA2 = 1	Peripheral register A

Address (Hex)	R/W	Other condition	Addressed unit and register
			TWI ADLC channel 3 – 0 (The ADLCs can also be addressed by the DMAC at DMA cycle 2 but the R/W line is then inverted and address bit 1 is hard-wired 1.)
			ADLC channel 3
3F – 38	→	→	Correspond to channel 0 (Addresses 27 – 20)
			ADLC channel 2
37 – 30	→	→	Correspond to channel 0 (Addresses 27 – 20)
			ADLC channel 1
2F – 28	→	→	Correspond to channel 0 (Addresses 27 – 20)
			ADLC channel 0
27	→	→	= 23
26	→		= 22
25	→	→	= 21
24	→		= 20
23	W	CR1:0 = 0	Transmitter FIFO (terminate)
23	W	CR1:0 = 1	Control register 4
23	R		Receiver FIFO
22	W		Transmitter FIFO (continue)
22	R		Receiver FIFO
21	W	CR1:0 = 0	Control register 2
21	W	CR1:0 = 1	Control register 3
21	R		Status register 2
20	W		Control register 1 (CR1)
20	R		Status register 1
1F – 17			**Not used**
			DMAC
16	R/W		Data chain
15	R/W		Interrupt control
14	R/W		Priority control
13	R/W		Channel 3 control
12	R/W		Channel 2 control
11	R/W		Channel 1 control
10	R/W		Channel 0 control
0F	R/W		Byte count, channel 3 (LSB)
0E	R/W		Byte count, channel 3 (MSB)
0D	R/W		Address, channel 3 (LSB)
0C	R/W		Address, channel 3 (MSB)
0B	R/W		Byte count, channel 2 (LSB)
0A	R/W		Byte count, channel 2 (MSB)
09	R/W		Address, channel 2 (LSB)
08	R/W		Address, channel 2 (MSB)
07	R/W		Byte count, channel 1 (LSB)
06	R/W		Byte count, channel 1 (MSB)
05	R/W		Address, channel 1 (LSB)
04	R/W		Address, channel 1 (MSB)
03	R/W		Byte count, channel 0 (LSB)
02	R/W		Byte count, channel 0 (MSB)
01	R/W		Address, channel 0 (LSB)
00	R/W		Address, channel 0 (MSB)


COMMUNICATION PROCESSOR BOARD AND SUBUNITS
CPB
TUA-T
TUA-T / TUA-E
Power on
Parity error
MCP
Basic clock
Timing logic
Timing
MPU
R
NM1
DBE
Ø1
Ø2
IRQ
(16)
AI Address (Internal)
(8)
Data D7-0
VMA
DMA RQT
DMA GNT
DMA timing
DMA T1-3
DMA T2
Address decoder
MIC
Interrupt ctrl MIC – PIA
IPL
PE reset
IPL ROM 2k bytes
Stretch
IRQ 3,2
INT OUT
IRQ 1
MIC
MIC I/O
PER I/O
I7
I0
IRQ 7,6,5,4
Interrupt register
Programmable timer
Reset
Refresh timing
(8)
(Ø1)
Address selector and refresh logic
(Ø2)
(4)
Timing
RAD, CAD, RAS, CAS
Interrupt priority and Address modifier
IRQ
Set mask value
Irpt mask register
(3)
Memory addr decoder
MEMORY max 16+ 16 k bytes
(8)
(8)
AB (16)
DB
PE reset
Parity error
Parity error logic
PE
MRW 16 or 32 k bytes RWM
or
MRO max 24 k bytes ROM
AB 15-0
DB7-0
Read · MIC I/O
DMA T2
PER I/O
Address decoder
CE ADLC 0
CE ADLC 1
CE ADLC 2
CE ADLC 3
DEND 3
DEND 2
DEND 1
DEND 0
DMA mode
DMA RQT
DMA GNT
DMAC
Tx RQ 0
Tx RQ 1
Tx RQ 2
Tx RQ 3
DMA end
DMA mode
Ø2, R/W, A1, A0
DMA · Read
DMA · Write
Data
On
Ch off
Crosspoint selection logic
Crosspoint selection PIA
SCA (Only V 24/28 shown)
PER I/O
(7)
Addr dec
SCA addr
PIA
Data terminal ready (Modem)
Rate
Internal clock generator
RET
TET
Rec data
Transm data
SSDA
IRQ 3
ADLC
Req to send
Ready for sending
IRQ 7
ADLC 0
Bit clock 0
CTS 0
RTS 0
Rec data 0
Tr data 0
Clock
High speed modem
TUA RTS 0
Mod tr data 0
Mod rec data 0
IRQ 6
CE ADLC 1
ADLC 1
Bit clock 1
CTS 1
RTS 1
Rec data 1
Tr data 1
Clock
DEND 1
High speed modem
TUA RTS 1
Mod tr data 1
Mod rec data 1
IRQ 5
CE ADLC 2
ADLC 2
Bit clock 2
CTS 2
RTS 2
Rec data 2
Tr data 2
Clock
DEND 2
High speed modem
TUA RTS 2
Mod tr data 2
Mod rec data 2
IRQ 4
CE ADLC 3
ADLC 3
Bit clock
CTS 3
RTS 3
Rec data 3
Tr data 3
Clock
DEND 3
High speed modem
TUA RTS 3
Mod tr data 3
Mod rec data 3
Ch sel
Ch 0 off
Ch 1 off
Ch 2 off
Ch 3 off
Current to voltage converter
Current to voltage converter
Current to voltage converter
Current to voltage converter
TUA enable # n
Two - wire selection
Not mounted on two port version of TUA-T
Tw 0 en
Tw 7 en
Ch0
Ch1
Ch2
Ch3
Mod tr data 0
Mod rec data 0/TUA RTS 0
Dual line transceiver
Tw0
Tw1
Tw7

E90002659E

DATASAAB

Flexible Disk Unit

Contents

Figures

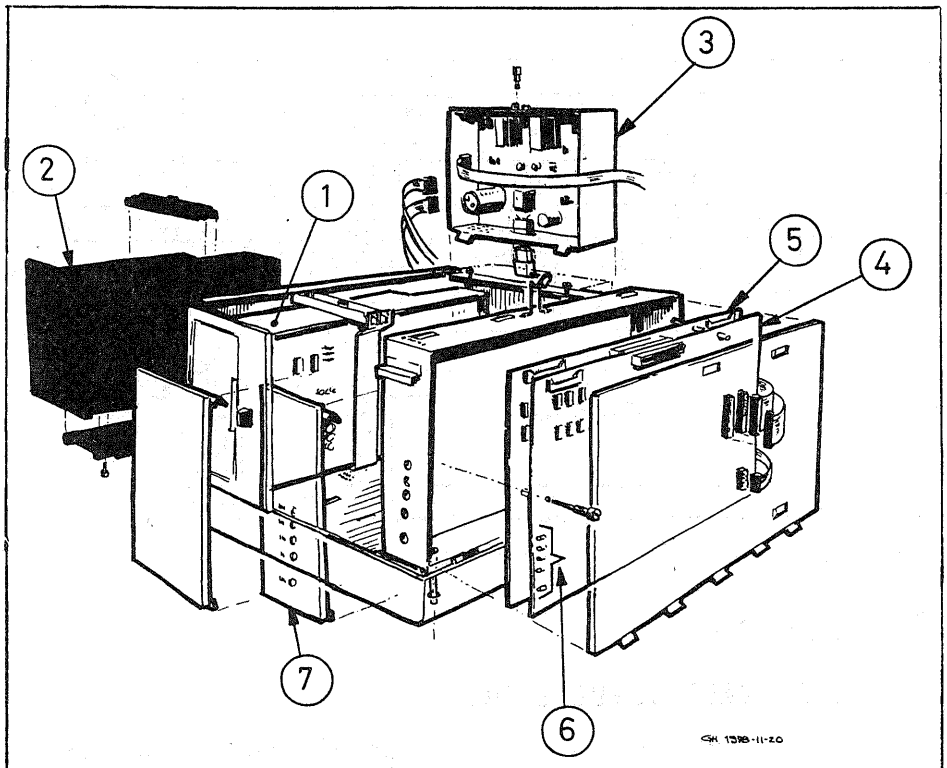
EE369-810
1979-08-15

GENERAL

The main task of the FD4120 terminal is to serve as a storage for

- the system programs to be loaded into the different terminals on start of the system
- the application programs created by means of Alfaform and Interactive Cobol
- data when the Transaction Collection Package is used.

MECHANICS



1. Flexible Disk Drive (FDD)
2. Optional FDD
3. Flexible Disk Power Supply (FPS)
4. Flexible Disk Adapter (FDA)
5. Flexible Disk Processor (FDP)
6. LED display
7. LED panel

Fig. 1. FD 4120 mechanical layout

BLOCK STRUCTURE

The electronic circuits for control of the Flexible Disk Drive are placed on two printed circuit boards:

- the Flexible Disk Processor (FDP), and
- the Flexible Disk Adapter (FDA).

The major function blocks and the flow of signals and data are shown in the diagram. It is followed by a brief presentation of the individual blocks.

Block Diagram

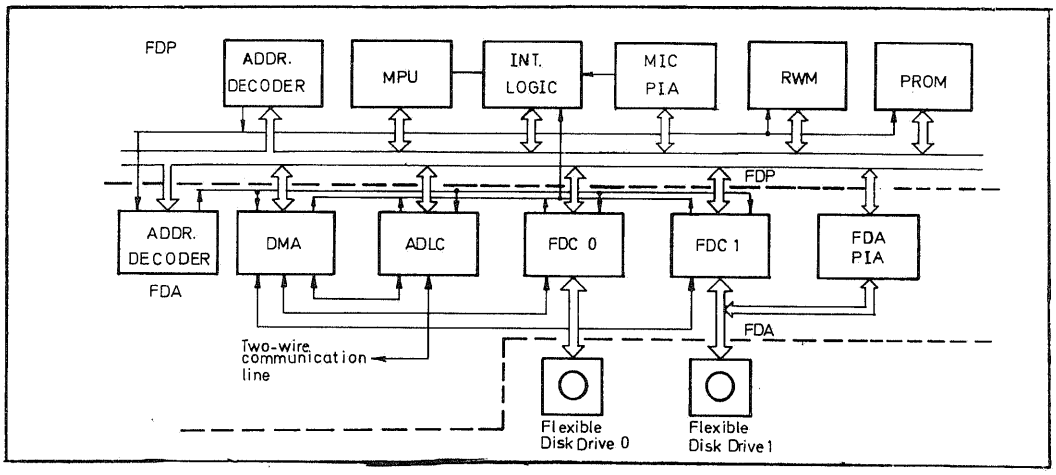


Fig. 2. Coarse block diagram

Block Description

Microprocessing Unit, MPU

The MPU consists of a Motorola 6800 chip with buffers and control circuitry. It works according to the program in IPL Memory and RWM.

It also permits a TSC mode for DMA, as well as hardware and software interrupts.

Initial Program Load-Memory, IPL-Memory

The IPL-Memory is a PROM area with a capacity of 4 kbytes. It contains all software for bootstrap loading of

- the flexible disk input/output supervisor,
- the communications handler, and
- the flexible disk handler

into the RWM-section.

RWM

The RWM has a capacity of max. 32 kbytes.

Interrupt Logic

The Interrupt Logic supports interrupts from the various system circuits such as the DMAC, ADLC, FDC and generates an interrupt request to the MPU.

Address Decoder

The Address Decoders block consists of fuseable link PROM:s, type MM6300 and 6330. They decode addresses from the MPU or DMAC blocks and generate a chip select signal to the addressed circuitry.

Direct Memory Access Controller, DMAC

The DMAC consists of a Motorola 6844 chip with buffers and control circuitry for four DMA-channels. It is working in the TSC mode (Three-state-control Steal) involving that it "steals" cycle time from the MPU.

Advanced Data Link Controller, ADLC

ADLC transmits or receives serial data on the two wire line according to specifications provided by the communication handler. The transfer rate is 300 kbits per second.

MIC PIA

The MIC PIA controls two interrupt functions (IRQ SOFT and IRQ TIMER).

FDA PIA

The FDA PIA is used as an input/output port for control signals from and to the FDD unit.

Flexible Disk Controllers, FDCØ, FDC1

The terminal has two FDCs type Western Digital 1771 or National Semiconductors 1771 with buffers and control circuitry.

Each FDC controls a flexible disk drive. The FDCs support the following features

- Reading/Writing IBM 3740 compatible diskettes
- Variable sector length
- CRC calculation

Flexible Disk Drive, FDD

The FDD can be of either a single-head or a double-head type. It contains control logic for drive mechanism, head positioning mechanism and logic for the read/write functions.

MICROPROCESSING UNIT, MPU

All circuit and signal names refer to the block diagram on app. 1. The page can be folded out for easy reference. Functional descriptions of the MPU, PIA, ADLC, and DMAC are provided in the "Microcomputer" chapter (publication No. EE365-810) and in the manufacturer's data sheets. The description below will cover the application of these circuits in the particular FD4120 configuration. It will follow a block pattern conveniently arranged to suit the necessity of referring to other documents.

An introductory statement of the basic communication routes will be made:

- The Microprocessing Unit (MPU) communicates through memory read/write operations with
  - PROM
  - RWM
  - the control registers in the MIC PIA, FDA PIA, FDCØ/1, ADLC and DMAC circuits.
- The ADLC and FDCØ/1 blocks communicate with Memory through DMA controlled by DMAC.

The MPU works according to the program stored in the PROM and that loaded into RWM through "power ON".

The memory operations by which the MPU controls the other circuits are carried out as follows:

The memory address and the opening signal VMA from the MPU are decoded by the CS-PROMs which issue a chip select signal (CS ROM, CS ADLC, etc) to the respective circuit. The selected circuit decodes the address on the A-bus, and with the R/W signal the MPU decides whether data shall be read from or written into the memory cell or register of the circuit.

Address Decoders

The Address Decoders are: CS-PROM1, CS-PROM2 and CS-PROM3.

- CS-PROM1: decodes the addresses from MPU or DMAC, when the VMA signal is present. It generates chip select signals for the memories (RWM, ROM) or an I/O signal to CS-PROM2/3. See table below

Address from MPU or DMAC A15-A8	A7-A0	Chip Select signal (CS)
00 to 3F	XX	RWM1
40 to 7F	XX	RWM2
E8 to EF	XX	ROM
F7	XX	I/O
F8 to FF	XX	ROM

XX = don't care (decoded by the selected circuit)

- CS-PROM2/3: decode the memory address from the MPU when the I/O signal is present. They generate, according to the table below, chip select signal for the MIC PIA, FDA PIA, FDCØ/1, ADLC or DMAC.

A15-A8	Address from MPU A7	A6	A5	A4	A3	A2	A1	A0	Chip select signal (CS)
F7	0	0	0	0	0	X	X	X	DMAC
F7	0	0	0	0	1	X	X	X	DMAC
F7	0	0	0	1	0	X	X	X	DMAC
F7	0	0	0	1	1	X	X	X	DMAC
F7	0	0	1	0	0	X	X	X	ADLC
F7	0	0	1	1	0	X	X	X	FDCØ
F7	0	0	1	1	1	X	X	X	FDC1
F7	0	1	0	0	0	X	X	X	FDA PIA
F7	1	1	0	0	0	X	X	X	MIC PIA
F7	1	1	1	1	1	X	X	X	TEST

X = don't care (decoded by the selected circuit)

Memory

R/W Memory

The RWM is built up of 2 x 16 kbytes dynamic RWM.

The chip select signal CS RAM1 selects the first 16 kbytes located on the addresses 0000-3FFF. The other 16 kbytes located on the addresses 4000-7FFF is selected by the signal CS RAM2.

The RWM is addressed by

- MPU
- DMAC
- Refresh counter

The refresh and memory access multiplexing logic together with the timing logic provide the memory with

- Row address and row address strobe (RAS)
- Column address and column address strobe (CAS)
- Write strobe (WG)

The R/W signal from the MPU or DMAC sets the data bus buffer to the memory in the output or input state. When present as Write (W) it also opens for the Write strobe.

PROM

The IPL memory is built up of 2 x 2 kbytes PROM. The CS PROM signal is generated when the memory is addressed (read) by MPU or DMAC.

The A12 Bit (0/1) in the address selects the first 2 kbytes located on the addresses E800-EFFF or the second 2 kbytes on the addresses F800-FFFF.

Organization

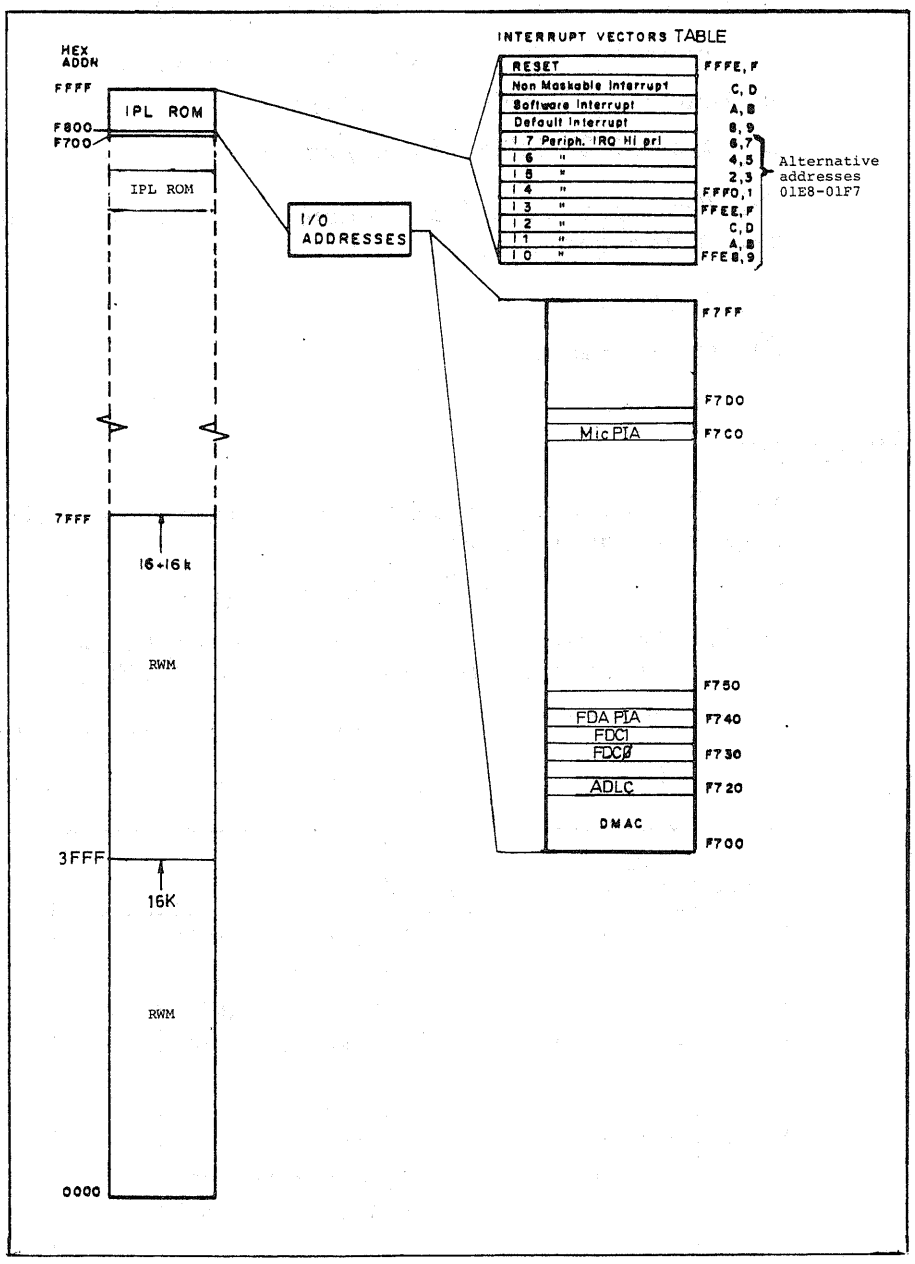


Fig. 3. Memory map

The Interrupt vectors table, contains the start addresses (vectors) for the Interrupt routines. The I/O address table shows the address fields used by the different blocks.

DIRECT MEMORY ACCESS CONTROLLER, DMAC

The DMAC controls four DMA channels.

- Channel 0 ADLC (transmit = read)
- Channel 1 ADLC (receive = write)
- Channel 2 FDCØ (read/write)
- Channel 3 FDC1 (read/write)

The DMA operation for a channel can be divided into two operations.

1) Initiation
2) Data Transfer

Initiation

The MPU selects the DMAC and initiates its different registers as follows:

- The Address and Byte Count registers obtain
  1. the address of the first byte to be read or written in memory by ADLC or FDCØ/1
  2. the number of memory bytes to be read or written.

This information is entered according to the table below.

Address from MPU A15–A8	A7	A6	A5	A4	A3	A2	A1	A0	Data Bus D7–D0	
F7	0	0	0	0	0	0	0	0	Address (MSB)	0
	0	0	0	0	0	0	0	1	Address (LSB)	
	0	0	0	0	0	0	1	0	Byte Count (MSB)	
	0	0	0	0	0	0	1	1	Byte Count (LSB)	
F7	0	0	0	0	0	1	0	0	Address (MSB)	1
	0	0	0	0	0	1	0	1	Address (LSB)	
	0	0	0	0	0	1	1	0	Byte Count (MSB)	
	0	0	0	0	0	1	1	1	Byte Count (LSB)	
F7	0	0	0	0	1	0	0	0	Address (MSB)	2
	0	0	0	0	1	0	0	1	Address (LSB)	
	0	0	0	0	1	0	1	0	Byte Count (MSB)	
	0	0	0	0	1	0	1	1	Byte Count (LSB)	
F7	0	0	0	0	1	1	0	0	Address (MSB)	3
	0	0	0	0	1	1	0	1	Address (LSB)	
	0	0	0	0	1	1	1	0	Byte Count (MSB)	
	0	0	0	0	1	1	1	1	Byte Count (LSB)	

MSB = 8 most significant bits

LSB = 8 least significant bits

- The Channel control register is initiated according to the table below with commands for the DMAC to

  1. produce read (DØ=1) or write (DØ=Ø) state on the R/W line when addressing the Memory.

  2. operate in TSC steal mode (D2=1, D1=Ø),

  3. count the memory address up (D3=Ø)/down (D3=1).

Channel	Address from MPU								Data Bus							
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	F7	0	0	0	1	0	0	0	0	X	X	X	X	0/1	0	0/1	
1	F7	0	0	0	1	0	0	0	1	X	X	X	X	0/1	0	0/1	
2	F7	0	0	0	1	0	0	1	0	X	X	X	X	0/1	1	0	0/1
3	F7	0	0	0	1	0	0	1	1	X	X	X	X	0/1	1	0	0/1

X = don't care

- The Priority register is initiated according to table below with information

  1. for the DMAC to refuse (DØ-D3=Ø) or accept (DØ-D3=1) a transfer request (TXRQ) from a channel,

  2. that specifies the priority chain for the channels to 0, 1, 2 and 3 (D7=Ø).

Channel	Address from MPU								Data Bus							
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0-3	F7	0	0	0	1	0	1	0	0	0	X	X	X	0/1	0/1	0/1	0/1

X = don't care

- The Interrupt Control register is initiated with an order for the DMAC to generate an Interrupt Request (IRQ) when the DMA for the respective channel is ended (DEND).

  The following programming applies:
  (Channel 0 = D0, Channel 1 = D1 etc).
  ("1"=Enables interrupt, "0"=no interrupt)

Channel	Address from MPU								Data Bus							
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0-3	F7	0	0	0	1	0	1	0	1	X	X	X	X	0/1	0/1	0/1	0/1

X = don't care

- The Data Chain register is initiated with an order for DMAC to operate in four channel mode (D0 = 0, D3 = 1).

Channel	Address from MPU								Data Bus							
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0-3	F7	0	0	0	1	0	1	1	0	X	X	X	X	1	0	0	0

X = don't care

Transfer

After initiation, DMAC is prepared to receive the transfer request (TXRQ) from ADLC or FDCØ/1. The arrival of TXRQ will trip off the following sequence of events:

- A DMA request (DRQT) is sent to the "stretch" block. This block returns two signals, the Isolate MPU and the GRANT.

- The Isolate MPU sets the MPU in the TSC Steal mode by"stretching"the clock signal Ø1 and Ø2 to the MPU. It also sets the address and data buses from the MPU in a high impedance state to enable the DMA.

- The GRANT signal goes to the DMAC, giving it control of the address bus and the data direction line (R/W C BUS).

- DMAC then produces the signals TxAKA, TxAKB and TxSTB. A combination of the signals TxAKA and TxAKB generates a CS signal to the block that issued TxRQ. This is done according to the table below.

TxRQ on channel	$\overline{CS}$/TxAKB	TxAKA	CS to
0=ADLC (trans)	0	0	ADLC
1=ADLC (receive)	0	1	ADLC
2=FDCØ	1	0	FDCØ
3=FDC1	1	1	FDC1

The TxSTB serves as a strobe signal for the following purposes:

1. to open for the CS signal to ADLC or FDCØ/1.
2. to set the selector connected with the ADLC in DMA mode. The ADLC (if selected) will then read or write in Memory under the control of DMAC.
3. to open for the RE or WE signals to the FDCØ/1. The FDCØ/1 (if selected) will then read or write in Memory under control of DMAC.
4. to generate, together with Isolate MPU, a VMA signal for the CS PROM1.

- From CS PROM1 comes an enable signal to the memory area addressed from the DMAC.
- A data byte is then transferred between ADLC or FDCØ/1 and memory.
- After each transfer of a data byte DMAC investigates if the initiated number of bytes (in byte count reg.) have been transferred between the memory and the unit which requested DMA. If not, the DMAC waits for a new TxRQ from the unit.
- When the transfer of the whole message is completed a DATA END (DEND) signal will be issued from the DMAC.
- This signal and the absence of the GRANT signal will cause an interrupt request (IRQ DMA) to the MPU and the DMA is ended.

TWO-WIRE INTERFACE

Advanced Data Link Controller, ADLC

The ADLC is the link between the memory of the FD unit and the 2-wire line. It works with direct access to Memory under control of DMAC. Hence DMAC must be initiated before the ADLC can request a DMA.

The operation of ADLC comprises three phases:

- Initiation
- Reception
- Transmission

Initiation

ADLC contains four Control registers. When power is switched on, Control Register 1 is initiated by the MPU as follows:

Address from MPU									Data Bus							
A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
F7	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1

Bit	Significance
D0=1	Provides the internal Address Control (AC) signal which will be used in conjunction with A0, A1 to address control registers 3 and 4.
D6=1	Keeps the receiver section of the ADLC in the Reset state.
D7=1	Keeps the transmitter section of the ADLC in the Reset state.

- Control register 3

  Control register 3 is initiated as follows:

Hex Address from MPU									AC	Data Bus							
A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	from reg 1	D7	D6	D5	D4	D3	D2	D1	D0
F7	0	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0

Bit	Significance
D0=0	The word length within the Information Field in a frame is defined in Control Register 4.
D1=0	The Control field of the frame is assumed to be 8 bits wide.
D2=0	All eight bits in the Address field of the frame will be used for addressing the ADLC.
D3=0	The ADLC will generate consecutive 1's in the Transmission IDLE state or give an interrupt after reception of 15 or more consecutive 1's.
D4=0	No indication when the Flag in the frame is received.
D5=0	The ADLC operates in the point-to-point communication mode (Non-Loop-mode).
D7=1	The Data Terminal Ready (DTR) output indicates that the terminal is ready.

- Control register 4 is initiated as follows:

Address from MPU									AC	Data Bus							
A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	from reg 1	D7	D6	D5	D4	D3	D2	D1	D0
F7	0	0	1	0	0	0	1	1	1	0	0	0	1	1	1	1	1

Bit	Significance
D0=1	Separate opening and closing flags will be transmitted successively.
D1 and D2=1	The word length within the Information field is 8 bits when ADLC is transmitting.
D3 and D4=1	The word length of the Information field in a frame received by the ADLC shall be 8 bits.
D5 and D6=0	No abort conditions used.
D7=0	The transmit/receive data format is NRZ.

Reception

After initiation the MPU will set the ADLC in reception mode through programming Control Register 1 as follows:

Address from MPU								Data Bus							
A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
F7	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0

Bit	Significance
D0=0	No internal Address Control signal (AC).
D1=1	Enable Interrupt Request (IRQ) from the receiver section. Permits error indication during reception.
D3=1	Sets the ADLC in Receiver Data Service Request (RDSR) mode.
D6=0	Removes the reset condition from the receiver section.
D7=1	Keeps the transmitter section reset.

- ADLC is now prepared to receive data on the RxD input.
- When the first data byte of a frame has been received and is available in the receiver data register (RxData FIFO), ADLC generates the signal RDSR as a transfer request (TxRQ) to the DMAC.
- The DMAC then takes over the control of ADLC (see DMAC, chapter "Transfer").
- The DMAC generates a Read signal and an address (RS=1 and RSØ=0) which select the receiver data register.
- The Read signal also sets the data output buffer into the Out state.
- The data byte is then transferred from ADLC to the memory position pointed out by DMAC. ADLC removes the RDSR signal.
- The next data byte in the frame will cause a new transfer request.

- The loop described above will continue until the ADLC receives the End Flag.
- The DMAC will then generate an IRQ (see DMAC section Transfer) to the MPU, and the DMA transfer is ended.
- Alternatively, the DMA transfer can be terminated by an IRQ from ADLC.
- An IRQ from ADLC would be caused by an error in the reception (framing error, IDLE, receiver overrun etc).
- After a frame has been received, the MPU will decode its contents. During this operation both receiver and transmitter parts of ADLC are kept in the reset state by the MPU.

Transmission

The MPU sets the ADLC in transmission mode through programming the control registers as follows:

- Control register 1

Address from MPU								Data Bus							
A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
F7	0	0	1	0	0	0	0	0	0	1	0	1	0	1	0	0

Bit	Significance
D0=0	No internal Address Control signal (AC).
D2=0	Enables Interrupt Request (IRQ) from the transmitter section.
D4=1	Sets the ADLC in Transmitter Data Service Request (TDSR) mode.
D6=1	Keeps the receiver section reset.
D7=0	Removes the reset condition from the transmitter section.

- Control register 2

Address from MPU								Data Bus							
A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
F7	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0

Bit	Significance
D1=0	Single byte data transfer.
D2=0	Bit by bit Mark Idle (1's) for the IDLE state in transmission.
D3=0	TDRA status.
D7=1	Activates the Request to Send (RTS) output from ADLC.

- The DELAY block responds to the send request (RTS) with "Clear to Send" (CTS).
- CTS together with the condition TDSR (in status register 1) will cause the signal TDSR (TxRQ from ADLC to DMAC).
- The DMAC responds to the transfer request (see DMAC section "Transfer") by generating a Write signal and a "Frame Continue" address (RS1=1 and RSØ=Ø) which selects the transmitter data register in ADLC.
- The Write signal also sets ADLC input buffer in the In state.
- The first data byte is then transferred from Memory (addressed by DMAC) to the transmitter data register in ADLC, and the TDSR signal is removed.
- When the ADLC is prepared to receive another data byte for the frame it again generates a TDSR signal as a transfer request to DMAC.
- The loop described above will continue until the DMAC indicates that all bytes have been transferred (see DMAC, section "Transfer").
- When ADLC receives the "Frame terminate" address (RS1=1 and RSØ=1) from DMAC it knows that the data byte received is the last to be transmitted. After the End Flag has been transmitted the frame is terminated and the ADLC goes into the Mark IDLE state (sending consecutive 1's).
- DMAC sends an IRQ to the MPU and the DMA transfer is ended.
- If an error (underflow) occurs during transmission, the ADLC sends an IRQ to the MPU, and the frame is terminated.

Modulator/Demodulator

- Modulation (see description of Display Unit EE360-810).
- Demodulation (see the same description).

FLEXIBLE DISK INTERFACE

FDA PIA

The FDA PIA is programmed by the MPU during system initialization as follows:

- Control registers (CR)

Control register	Address from MPU A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	Data Bus D7	D6	D5	D4	D3	D2	D1	D0
A	F7	0	1	0	0	0	0	0	1	X	X	X	X	X	0	X	X
B	F7	0	1	0	0	0	0	1	1	X	X	X	X	X	0	X	X

X = don't care

Bit	Significance
D2=0	Selects the Data Direction Registers when the proper register select signals are applied to RSØ and RS1.

- Data Direction Registers (DDR)

Data Direction Register	Address from MPU A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	CR bit 2	Data Bus D7	D6	D5	D4	D3	D2	D1	D0
A	F7	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
B	F7	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1

In the table each D-bit on the "A" row represents a data line on the PA7-PA0 bus. The bit value defines the direction of the information flow; 0 for input and 1 for output.

Similarly, bit values on the "B" row control direction of flow on the PB7-PB0 bus.

After initiating the DD registers the MPU will set bit D2 in the Control registers to "1". This enables the MPU to address the Peripheral Interface Registers in FDA PIA.

- Peripheral Interface Register A

This register serves as an interface for the functions described in the function table below and is addressed as follows:

A15-A8	Address from MPU A7 A6 A5 A4 A3 A2 A1 A0	CR bit 2	Data Bus D7 D6 D5 D4 D3 D2 D1 D0
F7	0 1 0 0 0 0 0 0	1	See PA7-PA0 in the table below (D7=PA7, D6=PA6 etc)

Note: The R/W-signal controls the direction (IN =1) (OUT=0)

PA	Signal name	Direction (see DDR)	Function if = 1	Function if = 0
0	Strap Cond Ø	IN	Initial loading of the FD-unit from the two-wire (Not strapped).	Initial loading of the FD-unit from the diskette (strapped).
1	Strap Cond 1	IN	Not used.	Not used.
2	Ready	OUT	The Ready LED on the FD 4120 front lights.	The Ready LED extinguishes.
3	Door Unlock (Drive Ø)	OUT	The door unlock magnet is not active (the LED on the FDD is not lit).	The door unlock magnet is active (the LED on the FDD is lit).
4	Write Fault Reset (Drive Ø)	OUT	Inactive.	Resets the write fault flip flop in the FDD.
5	Head Select (Drive Ø)	OUT	Selects head 1 (if two heads).	Selects head 0 (also for a drive with one head).
6	Drive Connected (Drive Ø)	IN	Drive not connected.	Drive connected.
7	Disk Type (Drive Ø)	IN	Double sided.	Single sided.

- Peripheral Interface Register B

  This register serves as an interface for the function described in the function table below and is addressed as follows:

Address from MPU A15-A8 A7	A6 A5 A4 A3 A2 A1 A0	CR bit 2	Data Bus D7 D6 D5 D4 D3 D2 D1 D0
F7 0	1 0 0 0 0 1 0	1	See PB7-PB0 in the table below (PB7=D7, PB6=D6 etc.)

Function Table

PA	Signal name	Direction (see DDR)	Function if = 1	Function if = 0
0	Bit 0 *)	OUT	The ERROR 1 LED on the FD4120 front lights.	The ERROR 1 LED extinguishes.
1	Bit 1 *)	OUT	The ERROR 2 LED on the FD4120 front lights.	The ERROR 2 LED extinguishes.
2	Bit 2 *)	OUT	The ERROR 3 LED on the FD4120 fron lights.	The ERROR 3 LED extinguishes.
3	Door Unlock (Drive 1)	OUT	The door unlock magnet is not active. (The LED on the FDD is not lit).	The door unlock magnet is active. (The LED on the FDD is lit).
4	Write Fault Reset (Drive 1)	OUT	Inactive.	Resets the Write Fault flip flop in the FDD.
5	Head Select (Drive 1)	OUT	Selects head 1 (if two heads).	Selects head 0 (also for a drive with one head).
6	Drive Connected (Drive 1)	IN	Drive not connected	Drive connected
7	Disk Type (Drive 1)	IN	Double sided	Single sided

*) The different combination of bits 0-3 and their meaning is described in chapter FDCØ/1 section "Error Messages".

Flexible Disk Controller, FDCØ/1

The tables below describe the Input and Output signals covered by the designation "CONTROL" in the Block diagram.

Input

Signal Name	Abbr.	Function
External Data Separator	$\overline{XTDS}$	Tied to 0 V. The composite serial data from the FD drive is separated externally.
Write Protect	$\overline{WPRT}$	When active indicates that the diskette is write protected. It terminates a Write command.
Write Fault	$\overline{WF}$	When active indicates that logic in the FD drive has discovered a fault in the writing process (see manual for the FD-drive). The Write command is terminated.
Index Pulse	$\overline{IP}$	Becomes active whenever an index mark is encountered (once per revolution) on the diskette.
Track 00	$\overline{TR00}$	Becomes active when the Ready/ Write head is positioned over track 00 of the diskette.
Ready	Ready	When active indicates that the FD drive is ready for a Read or Write operation. When active the Read or Write operation cannot be performed and an IRQ is generated.
Three-Phase Motor Select	$\overline{3PM}$	Tied to +5 V. The step direction motor control interface in the FDCØ/1 is used to control the FD drives. (Outputs $\overline{PHT}$/STEP and $\overline{PH2}$/DIRC).
Disk-Initialization	$\overline{DINT}$	Tied to +5 V. No function.
Test	$\overline{TEST}$	Tied to +5 V. No function.

Output

Signal Name	Abbr.	Function
Write Data	WD	Composite data (both clock and data bits) to the FD drive.
Write Gate	WG	Active when the diskette is to be written.
Track Greater Than 43	TG43	When active during a Read or Write operation, informs the FD drive that the Read/Write head is positioned between tracks 44 and 76 (see FD drive manual).
Three Phase Motors/Step Control Line	$\overline{PH1}$/ STEP	Used as STEP (see INPUT $\overline{3PM}$) line for control of step motor in the FD drive.
Three Phase Motors/Direction Control Line	$\overline{PH2}$/ DIRC	Used as DIRC (see INPUT $\overline{3PM}$) line for telling the step motor the direction of a STEP (high = step in, low = step out).

Registers

The MPU controls (reads/writes) the Status, Command, Track, Sector, and Data registers of the FDCØ/1. When a DMA is initiated the DMAC controls the Data register. The address, contents and function of each register are described below.

- Command register (CR)

  This 8-bit register holds the command presently being executed. The addressing and the commands to the register are described in the tables that follow:

  - Addressing

FDC	Address from MPU								WE	Data Bus							
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	*)	D7	D6	D5	D4	D3	D2	D1	D0
0	F7	0	0	1	1	0	0	0	0	1	Command (see table below)							
1	F7	0	0	1	1	1	0	0	0	1	Command (see table below)							

*) The WE=1 is generated by the MPU through setting the R/W in the Write state.

- Commands

The commands accepted and executed by the FDCØ/1 are of four types (I-IV). When ever a command is being executed the Busy status bit is set in the Status register. When a command is completed or an error condition accurs during execution, an interrupt request (IRQ) to the MPU is generated and the Busy status bit reset.

Type 1

Each of the Type 1 commands contains bits with a certain function. Those bits are marked with a letter and explained after the table.

Command	Data Bus bit 7 6 5 4 3 2 1 0	Function
Restore (Seek Track 0)	0 0 0 0 h v r1 r0	Upon receipt of this command, the Track 00 ($\overline{TR00}$) input is sampled. If $\overline{TR00}$ is active (indicating the Read/Write head is positioned over track 0), the Track Register is loaded with zeros and an interrupt is generated. If $\overline{TR00}$ is not active, stepping pulses at a rate specified by the r1 r0 field (bits 0 and 1) are issued until the $\overline{TR00}$ input is active. At this time, the Track Register is loaded with zeros and an interrupt is generated. If the $\overline{TR00}$ input does not turn low after 255 stepping pulses, the FDCØ/1 gives up and interrupts with the Seek Error status bit set.
Seek	0 0 0 1 h v r1 r0	This command assumes that the Track Register contains the track number of the current position of the Read/Write head and that the Data Register contains the desired track number. The FDCØ/1 will update the Track Register and issue stepping pulses in the appropriate direction until the contents of the Track Register are equal to the contents of the Data Register. An interrupt is generated at the completion of the command.
Step IN	0 1 0 u h v r1 r0	Upon receipt of this command, the FDCØ/1 issues one stepping pulse in the direction towards track 76. An interrupt is generated at the completion of the command.
Step OUT	0 1 1 u n v r1 r0	Upon receipt of this command, the FDCØ/1 issues one stepping pulse in the direction towards track 0. An interrupt is generated at the completion of the command.
Step *)	0 0 1 u h v r1 r0	Upon receipt of this command, the FDCØ/1 issues one stepping pulse 0 the floppy disk drive. The stepping motor direction is the same as in the previous step command. An interrupt is generated at the completion of the command.

*) Not used in the FD handler program.

- h = head load (D3)

  h = [1], the head is to be loaded at the beginning of the command.

  h = 0, HLD signal generated (see Head Load Timer).

  [ ] = used in the FD handler program.

- v = verify (D2)

  v = [1], verify the last track. The track address of the ID field is compared with the Track register. If there is a match and a valid ID CRC the verification is complete and an interrupt generated. If there is not a match the Seek Error is set in the Status Register and an interrupt generated.

  v = 0 No verify.

  [ ] = used in the FD handler program.

- r1, r0 = stepping motor rate (D1, D0)

  Programmable depending on FD drive type.

r1	r0	Rate
0	0	6 ms
0	1	6 ms
1	0	10 ms
1	1	20 ms

- u = up date (D4)

  u = 1, Track Register is updated by one for each step.

  u = [0], Track Register is not updated.

  [ ] = used in the FD handler program.

Type II

Each of the Type II Commands contains bits with a certain function depending on their value. Those bits are marked with a letter and explained after the table.

Command	Data bus bits 7 6 5 4 3 2 1 0	Function
Read Command	1 0 0 m b E 0 0	Upon receipt of this command, the Read/Write head is loaded. Then, when an ID field is encountered that has the correct track number, correct sector number, and correct CRC, the data field is inputted to the memory. The Data Address Mark of the data field must be found within 28 bytes of the correct ID field. If not, the Record Not Found status bit is set and the operation is terminated. When the first character or byte of the data field has been shifted through the Data Shift Register, it is transferred to the Data Register and a Data Request (DRQ) output is generated. When the next byte is loaded into the Data Shift Register, it is transferred to the Data Register and another DRQ output is generated, provided that the DMAC has previously read the Data Register. If one or more character are lost, the Lost Data status bit is set. This sequence continues until the data field has been inputted to the memory. If there is a CRC error in the data field, the CRC Error status bit is set, and the command is terminated (even if it is a multiple record command). At the end of the operation, the type of Data Address Mark encountered in the data field is recorded in the Status Register (bits 5 and 6).
Write Command	1 0 1 m b E a1 a0	Upon receipt of this command, the Read/Write head is loaded (HLD active). When an ID field is encountered that has the correct track number, correct sector number, and correct CRC, a DRQ output is generated. The FDCØ/1 counts off 11 bytes from tne CRC field and the Write Gate (WG) output is made active if the DRQ is serviced (i.e., the Data Register has been loaded by the memory. If DRQ has not been serviced, the command is terminated and the Lost Data status bit is set. If the DRQ has been serviced, the WG is made active and six bytes of all Zero leve levels are then written on the diskette. At this time, the Data Address Mark is then written on the diskette, as determined by the a1a0 field (bits 0 and 1). The FDCØ/1 then writes the data field by generating DRQ outputs to the computer. If the DRQ is not serviced in time, the Lost Data status bit is set and a byte of zeros is written on the diskette. The command is not terminated. After the last data byte has been written on the diskete, the two-byte CRC is computed internally and written on the diskette followed by one byte of all One levels. WG is then made inactive.

- m = Multiplex Record (D4)

  m = 0, a single sector is read or written and an interrupt is generated at the completion of the command.

  m = [1], multiple records are read or written with the Sector Register internally updated so that an address verification can occur on the next record. The FDCØ/1 continues to read or write multiple records and update the Sector Register until the Sector Register exceeds the number of sectors on the track or until the Force Interrupt command is loaded into the Command Register. When either of these occurs, the command is terminated and an interrupt is generated.

  [ ]= used in the FD handler program.

- b = Block Length (D3)

  b = 0, No IBM format. The value in the Sector Length Field on the diskette multiplied by 16 determines the number of bytes in the sector or data field as shown below:

Sector Length Field (hex)	Number of Bytes in Sector (decimal)
01	16
02	32
03	48
04	64
•	•
•	•
•	•
FF	4080
00	4096

  b = [1], IBM format. The value in the Sector Length Field on the diskette determines the length (number of characters) of the sector as shown below:

  [ ]= used in the FD handler program.

Sector Length Field (hex)	Number of Bytes in Sector (decimal)
00	128
01	256
02	512
03	1024

- E = Enable HLD & 10 ms delay (D2)

  E = 0 , the head is assumed engaged when the command is given.

  E = [1], HLD signal is made active and HLT (signal) input is sampled after an internal delay (see Head Load Timer).

  [ ] = used in the FD handler program.

- a1 a0 = Data Address Mark (DAM) (D1, D0)

  a1a0 determines the DAM code to be written on the diskette as follows:

a1	a0	Data Mark (hex)	Clock Mark (hex)
[0]	[0]	FB	C7
0	1	FA	C7
1	0	F9	C7
1	1	F8	C7

[ ] = used in the FD handler program.

Type III

The Read Track command in this group contains a bit with a certain function depending on the value. This bit is marked with a letter and explained after the table.

Command	Data Bus bits 7 6 5 4 3 2 1 0	Function
Read Address	1 1 0 0 0 1 0 0	Upon receipt of this command, the head is loaded. The next encountered ID field is then read in off the diskette, and the six data bytes of the ID field are assembled and transferred to the Data Register, and a DRQ output is generated for each byte. Although the CRC characters are inputted to the memory, the FCDØ/1 checks for validity and the CRC Error status bit is set if there is a CRC error. The Sector Address of the IC field is written into the Sector Register. At the end of the operation, an interrupt is generated.
Read Track	1 1 1 0 0 1 0 $\overline{S}$	Upon receipt of this command, the head is loaded. Reading starts with the leading edge of the first encountered index mark and continues until the next index pulse. As each byte is assembled, it is transferred to the Data Register and the Data Request (DRQ) output is generated for each byte. No CRC checking is performed. Gaps are included in the input data stream. Upon completion of the command, the interrupt is activated.
Write Track	1 1 1 1 0 1 0 0	Upon receipt of this command, the head is loaded. Writing starts with the leading edge of the first encountered index pulse and continues until the next index pulse, at which time the interrupt is activated. The Data Request output is activated immediately upon receiving the command and writing does not start until after the first byte has been loaded into the Data Register. If the Data Register has not been loaded by the second index pulse, the operation is terminated. This sets the Not Busy and Lost Data status bits, and activated the interrupt. If a byte is not present in the Data Register when needed, a byte of zeros is substituted. Address Marks and CRC characters are written on the diskette by detecting certain data byte patterns in the outgoing data stream as shown above. The CRC generator is initialized to all Ones when any data byte from F8 to FE is about to be transferred from the Data Register to the Data Shift Register.

S = Synchronize (D0)

$\overline{S}$ = 0 the accumulation of bytes is synchronized to each Address Mark encountered.

$\overline{S}$ = 1 Do not synchronize to AM.

☐ = used in the FD handler program.

Type IV

This command contains bits with a certain function depending on the value. Those bits are marked with a letter and explained after the table.

Command	Data Bus bits 7 6 5 4 3 2 1 0	Function
Force Interrupt	1 1 0 1 I_3 I_2 I_1 I_0	This command can be loaded into the Command Register at any time. If there is a current command under execution, the command is terminated and an interrupt is generated when the condition specified in the I_0 through I_3 field (bits 0 through 3) is detected. Mote than one condition may be specified.

- I_n = Interrupt condition (D3-D0).

  I_3-I_0= 0 No interrupt request is generated; but the current command is terminated.

  I_0 =[1] Interrupt when: Not Ready to Ready Transition.

  I_1 =[1] Interrupt when: Ready to Not Ready Transition.

  I_2 = 1 Interrupt when: Index pulse detected.

  I_3 = 1 Interrupt every 10 ms.

  [ ]= used in the FD handler program.

- Status Register

  This 8-bit register holds device Status information which is a function of the contents of the Command Register. The meaning of each status bit and the addressing of the register are described in the tables below.

  - Addressing

FDC	Hex address from MPU A15-A8	A7 A6 A5 A4 A3 A2 A1 A0	RE *	Data Bus D7 D6 D5 D4 D3 D2 D1 D0
ø	F7	0 0 1 1 0 0 0 0	1	S7-S1 see table below (S7=D7, S6=D6 etc.)
1	F7	0 0 1 1 1 0 0 0	1	S7-S1 see table below (S7=D7, S6=D6 etc.)

*) The RE="1" is generated by the MPU through setting the R/W CBUS in the Read state.

Table Status Register Summary

Bit	Commands					
	All Type I Commands	Read Address	Read	Read Track	Write	Write Track
S7	Not Ready	Not Ready	Not Ready	Not Ready	Not Ready	Not Ready
S6	Write Protect	0	Record Type	0	Write Protect	Write Protect
S5	Head Engaged	0	Record Type	0	Write Fault	Write Fault
S4	Seek Error	ID Not Found	Record Not Found	0	Record Not Found	0
S3	CRC Error	CRC Error	CRC Error	0	CRC Error	0
S2	Track 0	Lost Data	Lost Data	Lost Data	Lost Data	Lost Data
S1	Index	DRQ	DRQ	DRQ	DRQ	DRQ
S0	Busy	Busy	Busy	Busy	Busy	Busy

STATUS FOR TYPE I COMMANDS

BIT NAME	MEANING
S7 NOT READY	This bit when set indicates the drive is not ready. When reset it indicates that the drive is ready. This bit is an inverted copy of the READY input and logically 'ored' with MR.
S6 PROTECTED	When set, indicates Write Protect is activated. This bit is an inverted copy of $\overline{\text{WRPT}}$ input.
S5 HEAD LOADED	When set, it indicates the head is loaded and engaged. This bit is a logical "and" of HLD and HLT signals.
S4 SEEK ERROR	When set, the desired track was not verified. This bit is reset to 0 when updated.
S3 CRC ERROR	When set, there was one or more CRC errors encountered on an unsuccessful track verification operation. This bit is reset to 0 when updated.
S2 Track 00	When set, indicates Read Write head is positioned to Track 0. This bit is an inverted copy of the $\overline{\text{TROO}}$ input.
S1 INDEX	When set, indicates index mark detected from drive. This bit is an inverted copy of the $\overline{\text{IP}}$ input.
S0 BUSY	When set command is in progress. When reset no command is in progress.

STATUS BITS FOR TYPE II AND III COMMANDS

BIT NAME	MEANING
S7 NOT READY	This bit when set indicates the drive is not ready. When reset, it indicates that the drive is ready. This bit is an inverted copy of the READY input and 'ored' with MR. The TYPE II and III Commands will not execute unless the drive is ready.
S6 RECORD TYPE/ WRITE PROTECT	On read Record: In indicates the MSB of record-type code from data field address mark. On Read Track: Not Used. On any Write Track: It indicates a Write Protect. This bit is reset when updated.
S5 RECORD TYPE/ WRITE FAULT	On Read Record: It indicates the LSB of record-type code from data field address mark. On Read Track: Not Used. On any Write Track: It indicates a Write Fault. This bit is reset when updated.
S4 RECORD NOT FOUND	When set, it indicates that the desired track and sector were not found. This bit is reset when updated.
S3 CRC ERROR	If S4 is set, an error is found in one or more ID fields; otherwise it indicates error in data field. This bit is reset when updated.
S2 LOST DATA	When set, it indicates the computer did not respond to DRQ in one byte time. This bit is reset to zero when updated.
S1 DATA REQUEST	This bit is a copy of the DRQ output. When set, it indicates the DR is full on a Read operation or the DR is empty on a Write operation. This bit is reset to zero when updated.
S0 BUSY	When set, command is under execution. When reset, no command is under execution.

- Track register (TR)

  This 8-bit register holds the track number of the current Read/Write head position. The contents of the register are compared with the recorded track number in the ID field during disk Read, Write and Verify operations. The Track register is read or written in by the MPU.

FDC	Address from MPU								RW	WE	Data bus
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	(*)	(*)	D7-D0
Ø	F7	0	0	1	1	0	0	0	1	0	1	X
1	F7	0	0	1	1	1	0	0	1	0	1	X
Ø	F7	0	0	1	1	0	0	0	1	1	0	X
1	F7	0	0	1	1	1	0	0	1	1	0	X

(*) = The RE/WE signals are generated by the MPU through setting the R/W in the Read or Write state.

X = Track number (ex 10 decimal = 0A, hex).

- Sector register (SR)

  This 8-bit register holds the address of the desired sector position. The contents of the register are compared with the recorded sector number in the ID field during disk Read or Write operations. The Sector Register is read or written in by the MPU.

FDC	Address from MPU								RE	WE	Data bus
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	(*)	(*)	D7-D0
Ø	F7	0	0	1	1	0	0	1	0	0	1	X
1	F7	0	0	1	1	1	0	1	0	0	1	X
Ø	F7	0	0	1	1	0	0	1	0	1	0	X
1	F7	0	0	1	1	1	0	1	0	1	0	X

(*) = See above

X = Sector number (ex 15 decimal = 0F hex)

- Data register

  This 8-bit register holds the data during disk Read or Write operations. It can be addressed both from the MPU and the DMAC (during DMA operations).

FDC	Address from MPU						*	*	RE	WE	Data Bus
	A15-A8	A7	A6	A5	A4	A3	A2	A1	A0			D7-D0
Ø	F7	0	0	1	1	0	0	1	1	0	1	X
1	F7	0	0	1	1	1	0	1	1	0	1	X
Ø	F7	0	0	1	1	0	0	1	1	1	0	X
1	F7	0	0	1	1	1	0	1	1	1	0	X

* = The Isolate MPU signal will replace A1 and A0 during DMA.
X = Data

Data Transfer

On receipt of the interrupt from the Seek command sent to the FDCØ/1 the MPU checks the Status register to see that no error has occurred. If no error exists, the head is placed on the right track and the MPU can send a Read or Write command. The buffer start address and the number of bytes to be read or written etc. must be initiated in DMAC (see DMAC section "Initiation"). When DMAC receives the DMA request (DRQ) on the Write or Read command (see commands) from the MPU to the FDCØ/1, it takes control of the data transfer, (see DMAC section "Transfer"). On receipt of the IRQ from DMAC (DMA ended), the MPU terminates the Write command in FDCØ/1 by sending the Force Interrupt command.

Error Messages

An error during start up or Initial Program Load (IPL) is indicated on the ERROR-LEDs as follows:

ERROR 3	ERROR 2	ERROR 1	ERROR Message
0	0	0	No Error
0	0	1	CRC Error PROM1
0	1	0	CRC Error PROM2
0	1	1	Data Error RAM
1	0	0	Time out (FDC-chip)
1	0	1	Not Ready (FD-drive)
1	1	0	Read Error (Address, Vol. Label, VTOC
1	1	1	No sysbot read sysbot

Data/Clock Separator

A synchronization field is used to determine which output is to be the data output and which is to be the clock output when reading the sector that follows. The data characters of the sync field are all zeros, hence pulses obtained when reading the field are known to be clock pulses. Using this indication IC1771 allocates data/clock status to the outputs 53:3 and 53:6.

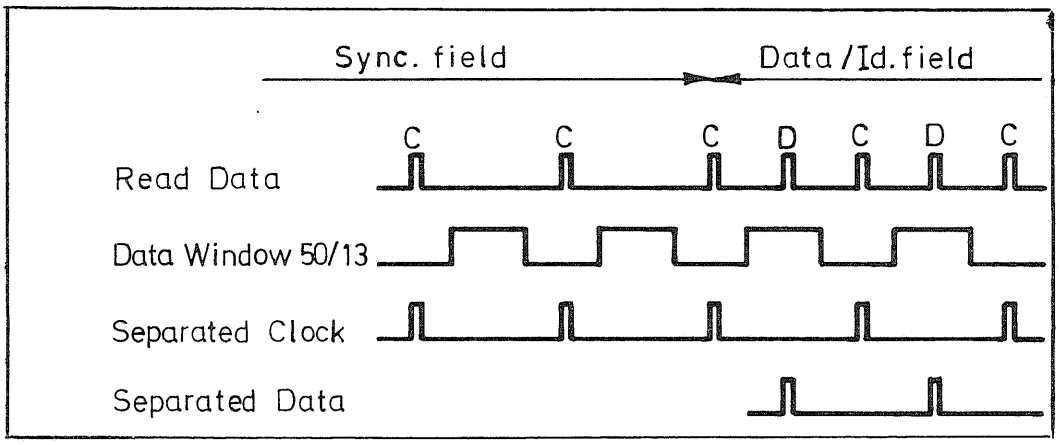


Fig. 4. Read data separation into data and clock

The Data Window defines the space of time in which the consecutive data pulse is to appear. The data window is generated by counters 49 and 50 which are loaded by each arriving Read Data pulse.

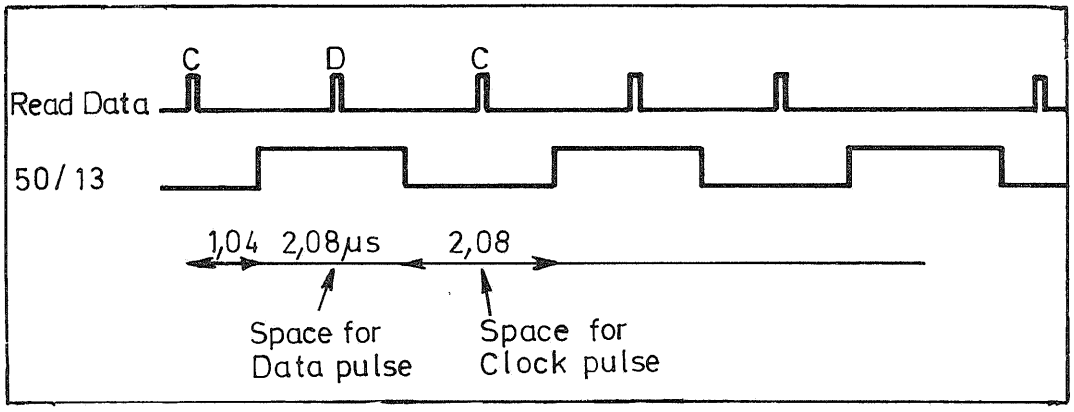


Fig. 5. Data window

Head Load Timer

The Head load timer block consists of a programmable timer (type MC14536), and the delay time is programmed (as 2^n) in the strapping field connected to the inputs A-D as follows:

Inpupt D C B A	Delay n
0 0 0 0	9
0 0 0 1	10
0 0 1 0	11
0 0 1 1	12
0 1 0 0	13
0 1 0 1	14
0 1 1 0	15
0 1 1 1	16
1 0 0 0	17
1 0 0 1	18
1 0 1 0	19
1 0 1 1	20
1 1 0 0	21
1 1 0 1	22
1 1 1 0	23
1 1 1 1	24

The clock frequency to the chip is 0.5 MHz (2 µs). This results in the programmable head load time of

$$2 \cdot 10^{-6} \cdot 2^{n-1} \text{ s}$$

When the timer receives the HLD signal it starts counting. It will count up to the programmed value in the strapping field and then generate a signal at the "Decode out" output. This signal stops the timer and combined with the signal HLD generates the signal HLT.

When HLT is active, the Read/Write head is assumed to be engaged against the diskette. The HLT input is sampled in the FDCØ/1 after each 10 millisecond internal delay.

INTERRUPT HANDLING

See "Microcomputer" chapter (Reg. No. EE356-810).

Interrupts

- Reset — An active low Reset pulse is generated at power on. This Reset signal is used to reset circuits throughout the system and to initialize the MPU. The MPU will fetch contents of cells FFFE and FFFF in the Interrupt Vectors Table (see Memory) to the program counter when Reset is deactivated.

- NMI — Non-maskable interrupt may be generated either from the test unit MCP (Microprocessor control panel) or from the NMI Switch located on the FDA board. The switch can be reached from outside (bottom right corner of the FD-unit).

  The MPU will fetch the contents of the cells FFFC and FFFD in the Interrupt Vectors Table to the program counter.

  The program may test the source of NMI by reading the MSB of the Peripheral Interface Register A in the MIC PIA.

- SWI — Software interrupt is an MPU instruction (see MPU description) and the MPU will fetch the contents of cells FFFA and FFFB in the Interrupt Vectors Table.

- IRQ — The eight interrupts that use the IRQ line are described below.

Interrupt Register

The interrupt register latches the logic value of the IRQ:s listed below and generates the I7-I0 signals to the FPLA.

IRQ	I
SOFT *)	0
TIMER *)	1
Not used	2
Not used	3
ADLC	4
FDC1	5
FDCØ	6
DMAC	7

*) See MIC PIA

FPLA

The FPLA generates an IRQ (exception see Mask Reg) to the MPU on reception of an I7-I0 signal.

The MPU responds by sending the vectors (addresses) FFF8 and FFF9.

These are modified by the FPLA into two addresses ("pair") directed towards the area FFE8-FFF7 of the Interrupt Vectors Table. The value of these addresses vary as determined by the Interrupt level.

With the IPL signal the same pair of addresses would select the area 01E8-01F7 of the alternative Interrupt vectors table in RWM.

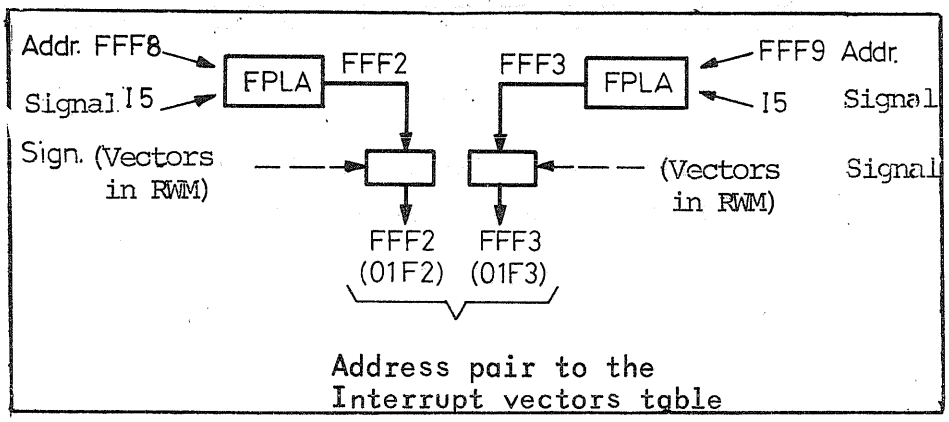


Fig. 6. Alternative interrupt vector addresses

The signals used in these procedures are shown and described in the tables on the following pages.

DATASAAB

Table. Interrupt pri and addr. mod FPLA-Truth table.

	INPUT																OUTPUT							
PIN / FUNCTION	A4	A3	A2	A1	FFEX, FFFX	From Mask Latch P3	P2	P1	I7	I6	I5	I4	I3	I2	I1	I0	P4	P3	P2	P1	INT LATCH	IRQ, SWI	MASK	IRQ *
	20	21	22	23	24	25	26	27	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	–	–	–	–	–	L	L	L	–	–	–	–	–	–	–	L								A
1	–	–	–	–	–	L	L	–	–	–	–	–	–	–	L	–								A
2	–	–	–	–	–	L	L	–	–	–	–	–	–	L	–	–								A
3	–	–	–	–	–	L	H	L	–	–	–	–	–	L	–	–								A
4	–	–	–	–	–	L	–	–	–	–	–	–	L	–	–	–								A
5	–	–	–	–	–	L	–	–	–	–	–	L	–	–	–	–								A
6	–	–	–	–	–	H	L	L	–	–	–	L	–	–	–	–								A
7	–	–	–	–	–	L	–	–	–	–	L	–	–	–	–	–								A
8	–	–	–	–	–	H	L	–	–	–	L	–	–	–	–	–								A
9	–	–	–	–	–	L	–	–	–	L	–	–	–	–	–	–								A
10	–	–	–	–	–	H	L	–	–	L	–	–	–	–	–	–								A
11	–	–	–	–	–	H	H	L	–	L	–	–	–	–	–	–								A
12	–	–	–	–	–	–	–	–	L	–	–	–	–	–	–	–								A
13	H	H	L	L	L	L	L	L	H	H	H	H	H	H	H	L		A						
14	H	H	L	L	L	L	L	–	H	H	H	H	H	H	L	–		A		A				
15	H	H	L	L	L	L	L	–	H	H	H	H	H	L	–	–		A	A					
16	H	H	L	L	L	L	H	L	H	H	H	H	H	L	–	–		A	A					
17	H	H	L	L	L	L	–	–	H	H	H	H	L	–	–	–		A	A	A				
18	H	H	L	L	L	L	–	–	H	H	H	H	L	–	–	–	A							
19	H	H	L	L	L	H	L	L	H	H	H	L	–	–	–	–	A							
20	H	H	L	L	L	L	–	–	H	H	L	–	–	–	–	–	A			A				
21	H	H	L	L	L	H	L	–	H	H	L	–	–	–	–	–	A			A				
22	H	H	L	L	L	L	–	–	H	L	–	–	–	–	–	–	A		A					
23	H	H	L	L	L	H	L	–	H	L	–	–	–	–	–	–	A		A					
24	H	H	L	L	L	H	H	L	H	L	–	–	–	–	–	–	A		A					
25	H	H	L	L	L	–	–	–	L	–	–	–	–	–	–	–	A		A	A				
26	H	H	L	L	L	L	L	L	H	H	H	H	H	H	H	H	A	A						
27	H	H	L	L	L	L	L	H	H	H	H	H	H	H	H	–	A	A						
28	H	H	L	L	L	L	H	L	H	H	H	H	H	H	–	–	A	A						
29	H	H	L	L	L	L	H	H	H	H	H	H	H	–	–	–	A	A						
30	H	H	L	L	L	H	L	L	H	H	H	H	–	–	–	–	A	A						
31	H	H	L	L	L	H	L	H	H	H	H	–	–	–	–	–	A	A						
32	H	H	L	L	L	H	H	L	H	H	–	–	–	–	–	–	A	A						
33	H	H	L	L	L	H	H	H	H	–	–	–	–	–	–	–	A	A						
34	H	H	L	–	L	–	–	–	–	–	–	–	–	–	–	–						A		
35	H	L	–	–	L	–	–	–	–	–	–	–	–	–	–	–	A						A	
36	L	H	–	–	L	–	–	–	–	–	–	–	–	–	–	–		A					A	
37	H	H	H	–	L	–	–	–	–	–	–	–	–	–	–	–	A	A						
38	H	H	–	H	L	–	–	–	–	–	–	–	–	–	–	–	A	A						
39	H	–	–	–	H	–	–	–	–	–	–	–	–	–	–	–	A							
40	–	H	–	–	H	–	–	–	–	–	–	–	–	–	–	–		A						
41	–	–	H	–	–	–	–	–	–	–	–	–	–	–	–	–			A					
42	–	–	–	H	–	–	–	–	–	–	–	–	–	–	–	–				A				
43																								
44																								
45																								
46																								
47																								

A = Active output level high ("1").

*) The IRQ output is active when Low ("0").

Table. Explanations of FPLA functions.

Operation No.	
0-12	IRQ generation if permitted by the Mask reg. value.
13-25	Address modification with regard to interrupt number and Mask value.
26-33	If the I7-I0 signal is removed when the MPU sends the vector addresses FFF8 and FFF9 these addresses are directly used as addresses to the cells for Default Interrupt in the interrupt vectors table.
34	The signal IRQ, SWI is generated on reception of the IRQ vector (FFF8, 9) or SWI vector (FFFA,B) from the MPU. It disables latching into the INT.REG because the modified addr. lines must be stable when the MPU fetches the interrupt vectors. In combination with the Vectors in RWM signal sets the IRQ, SWI signal up the addresses to the alternative interrupt vectors table.
35-36	The signal Set mask is generated on reception of the addresses FFE8 to FFF7 from the MPU. In combination with the functions 41-42 it sets the "MASK REG" (see Mask Reg.).
37-38	By pass for the interrupt vectors (addresses) FFFC,D (NMI) and FFFE,F (Reset) from the MPU.
39-42	By pass for the signals A4-A1.

Mask Register

When the MPU generates one of the addresses FFE8-FFF7 for the interrupts I7-I0, the Mask register is set to a specific value. This value specifies the interrupt levels that are allowed to activate the IRQ Line from the FPLA to the MPU (see table for FPLA Function No. 35-36).

MIC PIA

The MIC PIA is programmed by the MPU during system initialization as follows:

- Control Register (CR)

Control Register	Address from MPU A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	Data Bus D7	D6	D5	D4	D3	D2	D1	D0
A	F7	1	1	0	0	0	0	0	1	X	X	1	1	1	0	1	0

(Register B is not used)

X = don't care

Bit	Significance
D0=0	In combination with D1 and CA1 controls the IRQA output. Initiated as "0" which sets the interrupt output ($\overline{IRQA}$) to "1".
D1=1	Indicates that the interrupt input CA1 is active on low to high transition.
D2=0	Selects the Data Direction Register when the proper register select signals are applied to RSØ and RS1.
D3=1	Controls the interrupt output CA2. Initiated as "1" which sets the output CA2="1".
D5-D4=1	The condition to make CA2 an output.

- Data Direction Register (DDR)

Data Direction Register	Address from MPU A15-A8	A7	A6	A5	A4	A3	A2	A1	A0	CR bit 2	Data Bus D7	D6	D5	D4	D3	D2	D1	D0
A	F7	1	1	0	0	0	0	0	0	0	0	X	X	X	X	X	X	1

In the table each D-bit represents a data line on the PA7-PAØ bus. The bit value defines the direction of the information flow; 0 for input and 1 for output (X = not used).

After initiating the DD register the MPU will set bit D2 in the control register to "1". This enables the MPU to control the Peripheral Interface Register A and the Interrupt Lines ($\overline{IRQA}$ and CA2).

- Peripheral Interface Register A

R/W from MPU	Address from MPU A15-A8	A7 A6 A5 A4 A3 A2 A1 A0	CR bit 2	Data Bus D7 D6 D5 D4 D3 D2 D1 D0
IN=1 OUT=0	F7	1 1 0 0 0 0 0 0	1	See PA7-PAØ in the table below (D7=PA7), D6=PA6 etc.)

PA	Signal name	Direction (see DDR)	Function if = 1	Function if = 0
0	Vectors in RAM	OUT	Used for the addressing of the alternative Interrupt Vector Table.	The ordinary Interrupt Vector Table is addresses.
1-6	-	-	Not used	Not used
7	Button/MCP NMI	IN	NMI received by MPU from the NMI switch.	NMI received by MPU from MCP.

- Interrupt Lines

If the MPU wants an IRQ from the Timer the MPU sets the D0 bit to "1" in the Control register.

On arrival of the interrupt signal from the Timer to the CA1 input, the $\overline{\text{IRQA}}$ output is activated (low).

To enable a new interrupt request from the Timer the MPU must read the Data register A.

Through setting bit D3 to 0 the MPU program can generate a "SOFT" interrupt (level 0).

TIMING

The crystal controlled Clock Generator produces an 8 MHz basic clock signal, which generates four clock signals in the Clock Counter.

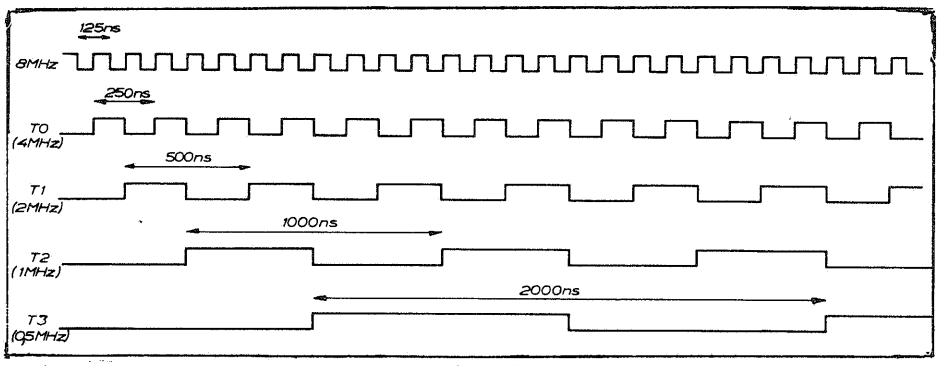


Fig. 7. Basic timing signals

Combinations of the T0-T2 signals are decoded in the Timing Logic by two PROMs (the EVEN and ODD PROMs).

- EVEN PROM

  This PROM generates the DBE, Ø1, $\overline{RAS}$, $\overline{CAS}$, $\overline{WG}$ and REFRESH. The table on page 40 shows the state of the timing signals after each half period of the T0 signal (125 ns).

  $\overline{ISOLATE\ MPU}$ indicates whether the state refers to DMA or not ("0" = DMA, "1" = no DMA). The pulse diagrams on page 41 show the non-DMA state (ISO.MPU = "0") and those on page 42 the DMA-state (ISO.MPU = "1").

- ODD PROM

  This PROM generates the timing signals DBE ROW ENABLE, $\overline{WE1771}$, $\overline{RGLATCH}$ and $\overline{RE1771}$. The table on page 40 shows the state of the timing signals after each half period of the T0 signal (125 ns).

  $\overline{ISOLATE\ MPU}$ indicates whether the state refers to DMA or not ("0" = DMA, "1" = no DMA). The pulse diagrams on page 41 show the non-DMA state (ISO.MPU = "0") and those on page 42 the DMA-state (ISO.MPU = "1").

The output from the Timing Logic is a combination of the timing signals from the EVEN and ODD latches (see pulse diagram pages 41 and 42).

The combined signals control the different operations in the system (see pulse diagram on pages 43 and 44).

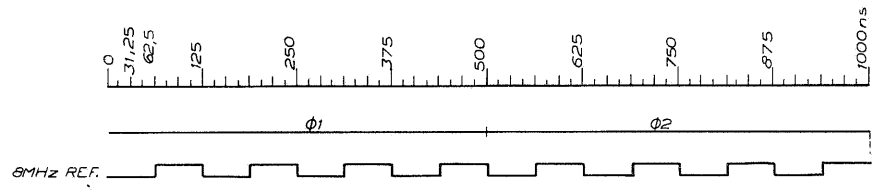
"EVEN PROM"

INPUT LINES				OUTPUT LINES								
0 V	$\overline{\text{ISOLATE MPU}}$	From Counter		DBE	Ø1	Not used	$\overline{\text{RAS}}$	$\overline{\text{CAS}}$	$\overline{\text{WG}}$	Not used	REFR	
4	3	2 1 0	Hex Value	7	6	5	4	3	2	1	0	Hex Value
0	0	0 0 0	0	0	1	0	1	1	1	0	0	5C
0	0	0 0 1	1	1	1	0	1	1	1	0	0	DC
0	0	0 1 0	2	1	1	0	0	1	1	0	0	CC
0	0	0 1 1	3	1	1	0	0	0	1	0	0	C4
0	0	1 0 0	4	1	0	0	0	0	1	0	0	84
0	0	1 0 1	5	1	0	0	0	0	1	0	0	84
0	0	1 1 0	6	1	0	0	0	0	1	0	0	84
0	0	1 1 1	7	1	0	0	0	0	0	0	0	80
0	1	0 0 0	8	0	1	0	1	1	1	0	0	5C
0	1	0 0 1	9	1	1	0	1	1	1	0	0	DC
0	1	0 1 0	A	1	1	0	0	1	1	0	0	CC
0	1	0 1 1	B	1	1	0	0	0	1	0	0	C4
0	1	1 0 0	C	1	0	0	0	0	0	0	0	80
0	1	1 0 1	D	1	0	0	1	1	1	0	1	9D
0	1	1 1 0	E	1	0	0	0	1	1	0	1	8D
0	1	1 1 1	F	1	0	0	0	1	1	0	1	8D

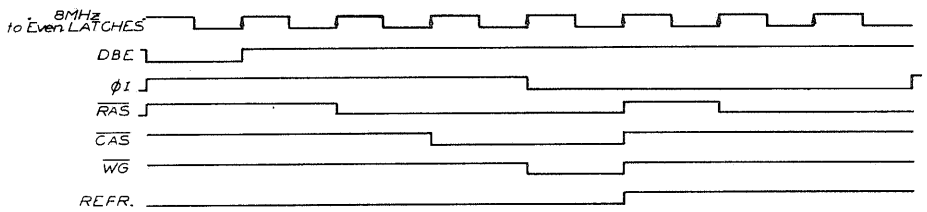
"ODD PROM"

INPUT LINES				OUTPUT LINES								
0 V	$\overline{\text{ISOLATE MPU}}$	from Counter		DBE	ROW ENABLE	$\overline{\text{WE 177T}}$	$\overline{\text{RG LATCH}}$	$\overline{\text{RE 1771}}$	Not used	Not used	Not used	
4	3	2 1 0	Hex Value	7	6	5	4	3	2	1	0	Hex Value
0	0	0 0 0	0	0	0	1	1	0	0	0	0	38
0	0	0 0 1	1	1	1	1	1	1	0	0	0	F8
0	0	0 1 0	2	1	0	1	1	1	0	0	0	B8
0	0	0 1 1	3	1	0	0	1	0	0	0	0	90
0	0	1 0 0	4	1	0	0	1	0	0	0	0	90
0	0	1 0 1	5	1	0	0	1	0	0	0	0	90
0	0	1 1 0	6	1	0	1	1	0	0	0	0	B0
0	0	1 1 1	7	1	0	1	1	0	0	0	0	B0
0	1	0 0 0	8	0	0	1	1	1	0	0	0	38
0	1	0 0 1	9	1	1	1	1	1	0	0	0	F8
0	1	0 1 0	A	1	0	1	1	1	0	0	0	B8
0	1	0 1 1	B	1	0	0	1	0	0	0	0	90
0	1	1 0 0	C	1	0	0	0	0	0	0	0	80
0	1	1 0 1	D	1	0	0	0	0	0	0	0	80
0	1	1 1 0	E	1	0	1	0	0	0	0	0	A0
0	1	1 1 1	F	1	0	1	0	0	0	0	0	A0

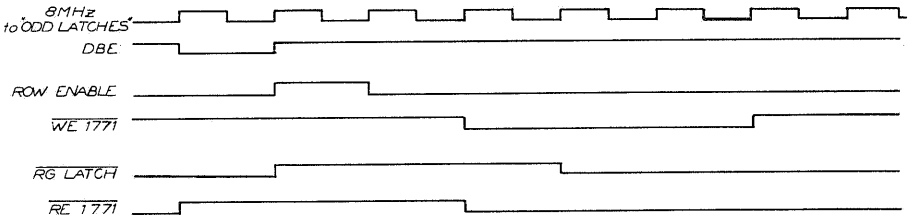
Timing pulses when ISO.MPU = "1"



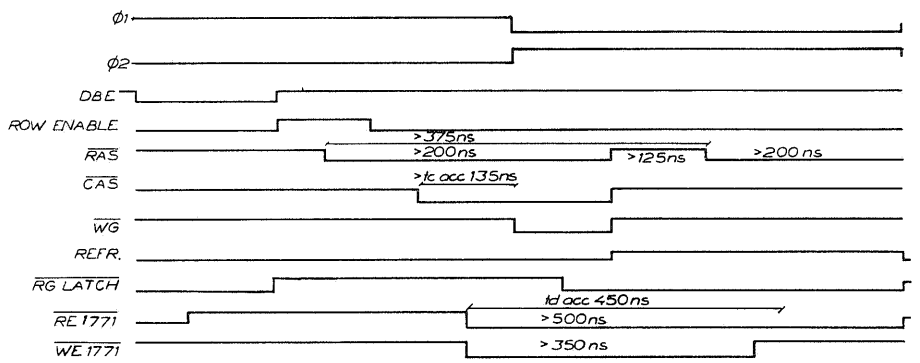
- Output EVEN Latches



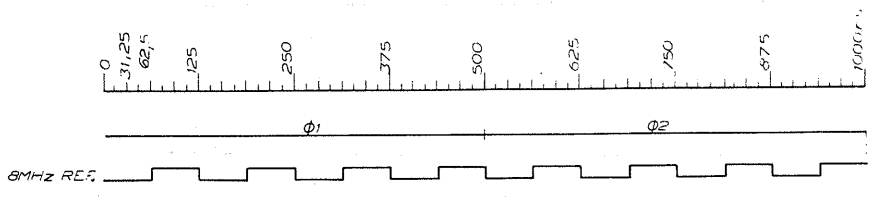
- Output ODD Latches



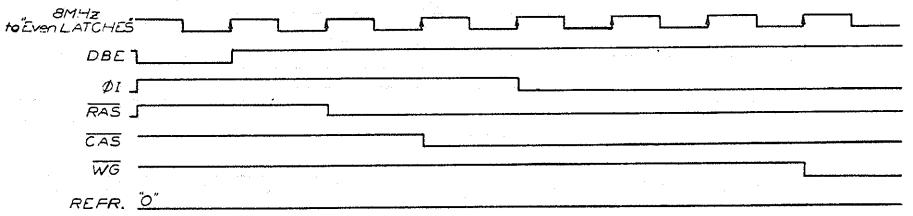
- Output Timing Logic



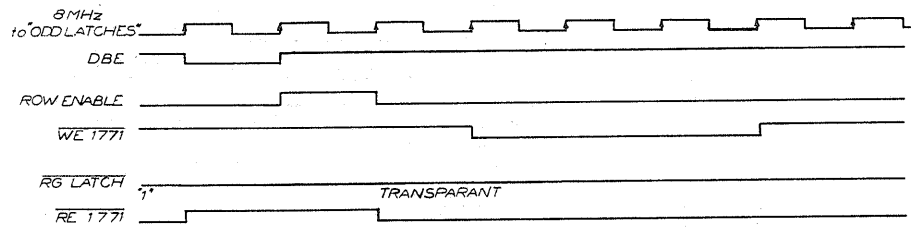
Timing pulses when ISO.MPU = "0"



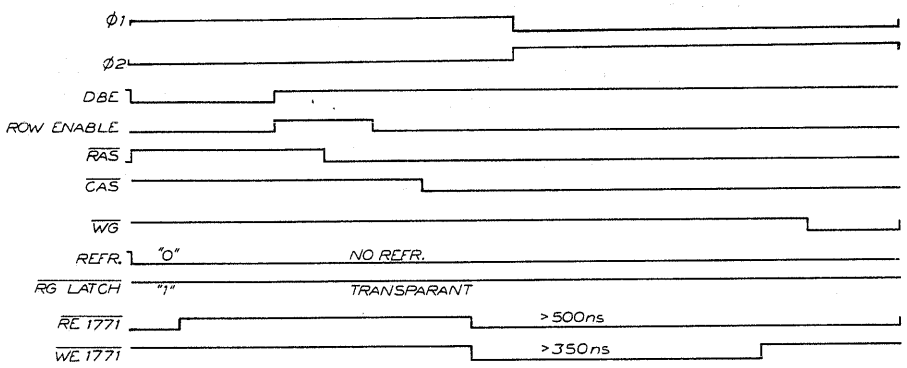
- Output EVEN Latches



- Output ODD Latches

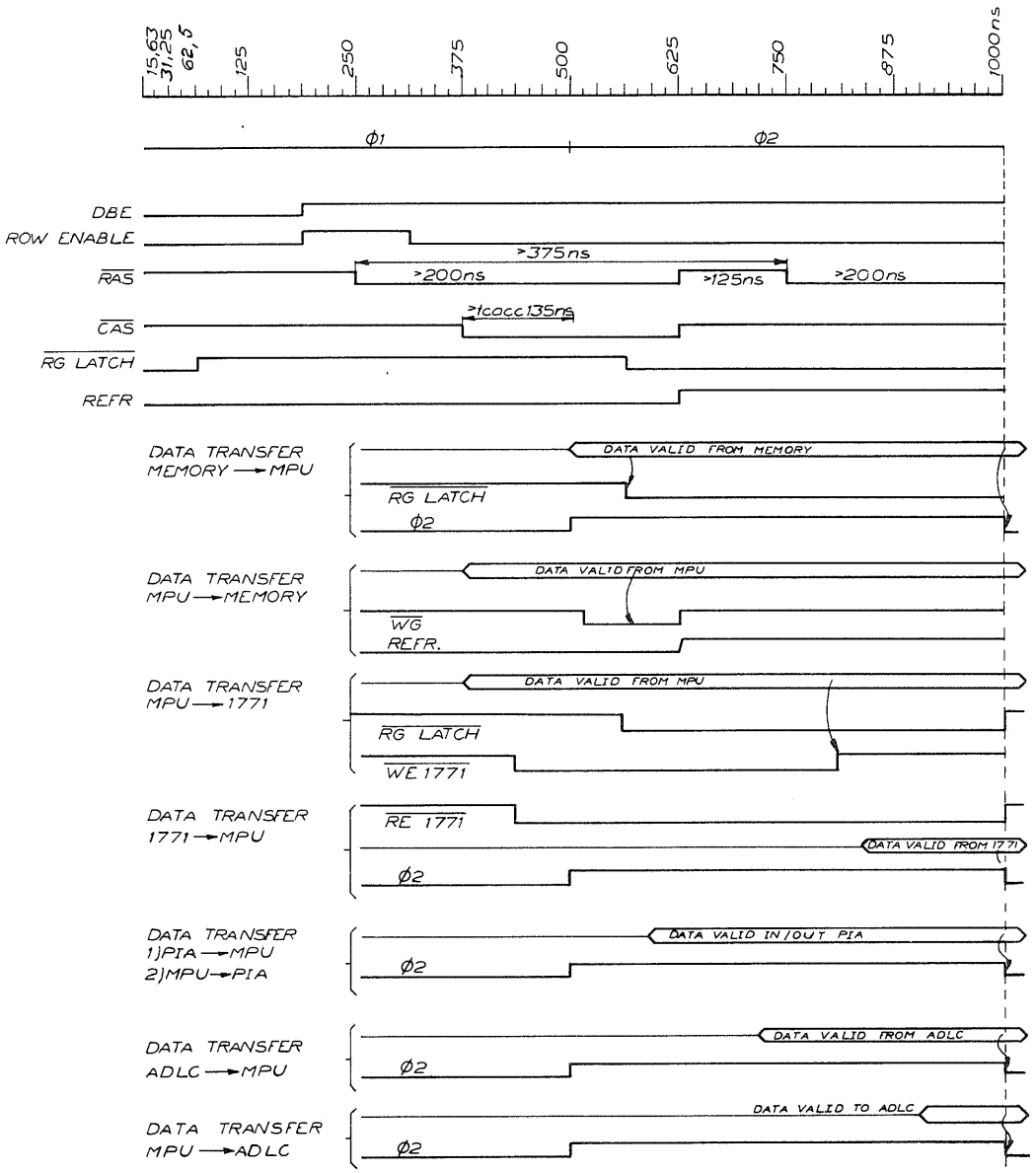


- Output Timing Logic

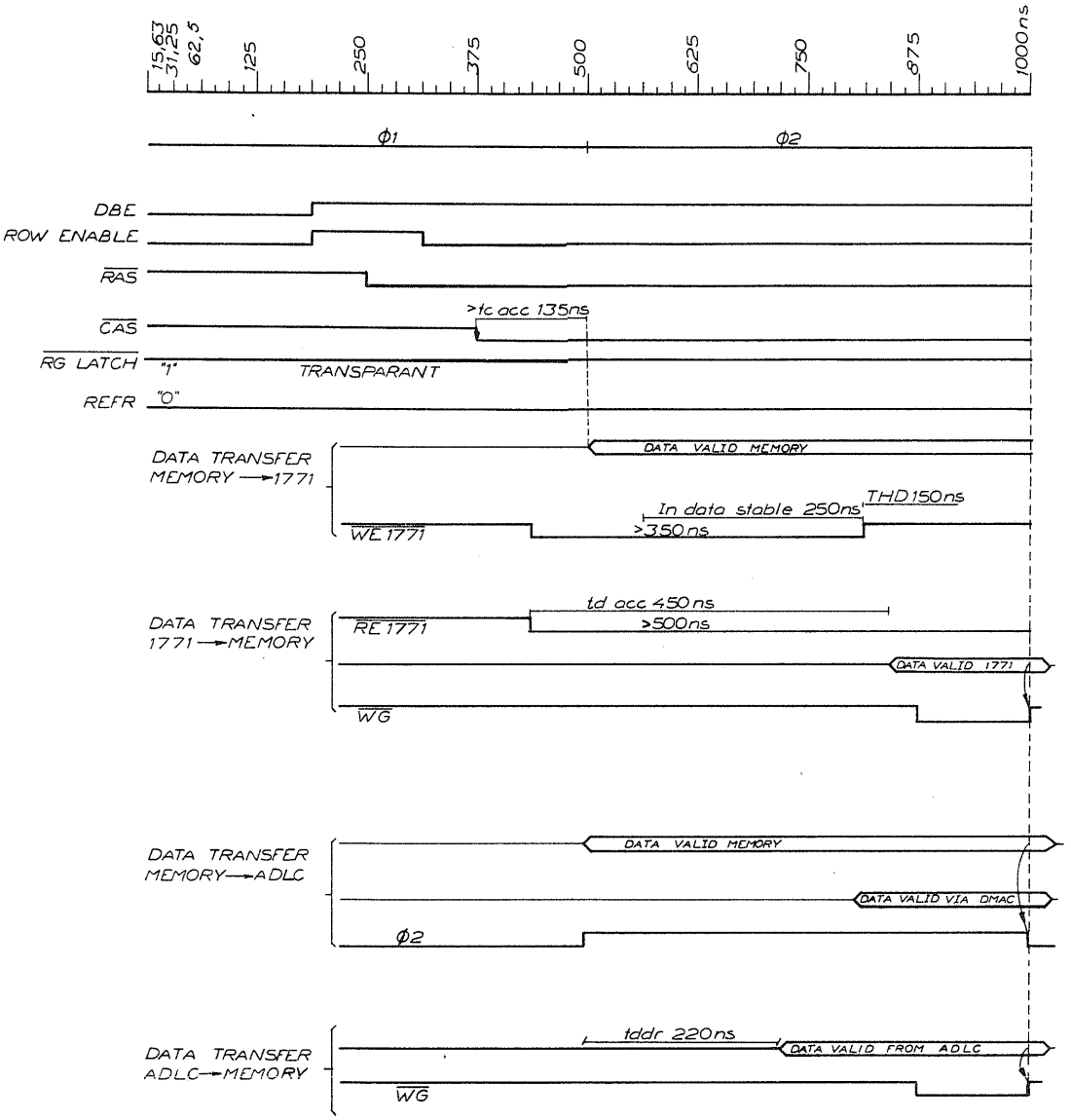


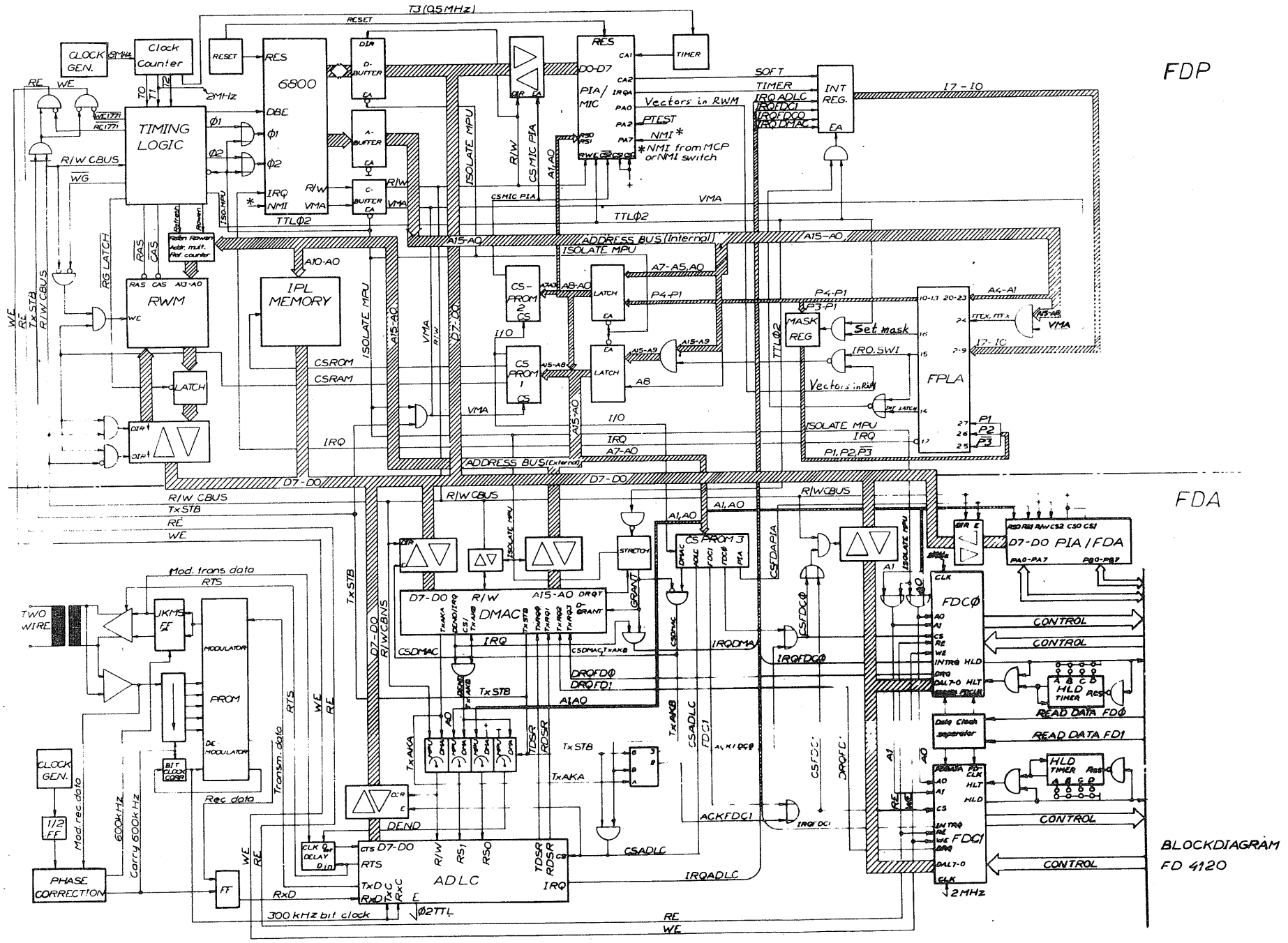
DATASAAB

Timing pulses when no DMA transfer



Timing pulses when DMA transfer




FDP
FDA
T3 (0.5 MHz)
RESET
CLOCK GEN.
Clock Counter
6800
TIMING LOGIC
RWM
IPL MEMORY
PIA/ MIC
Vectors in RWM
PTEST
NMI*
*NMI from MCP or NMI switch
INT REG.
MASK REG
Set Mask
IRQ.SWI
FPLA
ADDRESS BUS (Internal)
ADDRESS BUS (External)
ISOLATE MPU
CSROM
CSRAM
CS-PROM 2
CS-PROM 1
CS-PROM 3
LATCH
D-BUFFER
A-BUFFER
C-BUFFER
R/W CBUS
TxSTB
RE
WE
DMAC
ADLC
FDC0
FDC1
D7-D0 PIA / FDA
HLD TIMER
CONTROL
READ DATA FD0
READ DATA FD1
MODULATOR
PROM
DEMODULATOR
PHASE CORRECTION
BLOCKDIAGRAM
FD 4120

Flexible Disk Unit, FD 4122

Contents

Appendix

General

The Flexible Disk Unit (FD) 4122 in the Alfaskop System 41 shall serve as:

- a central processing unit, in a Personal Computer (PC) configuration, which includes a terminal and an optional printer.
- a storage unit for application programs, such as word processors etc.

Mechanics

The FD consists of four main parts, which are enclosed in a cabinet. These are:

- The flexible disk processor logic (FDPL) board which includes a microprocessor, up to 256 kbytes of R/W-memory, a flexible disk controller and a communications controller.
- The FD power supply (FPS), which provides the FD with the necessary voltages. The FPS is a switching device.
- Two slimline type 5 1/4" disk drives, each of which has a formatted capacity of 720 kbytes.

The mechanical layout of the FD can be seen in Figure 1.
Apart from the abovementioned features there are five LEDs mounted on the front panel of the cabinet.

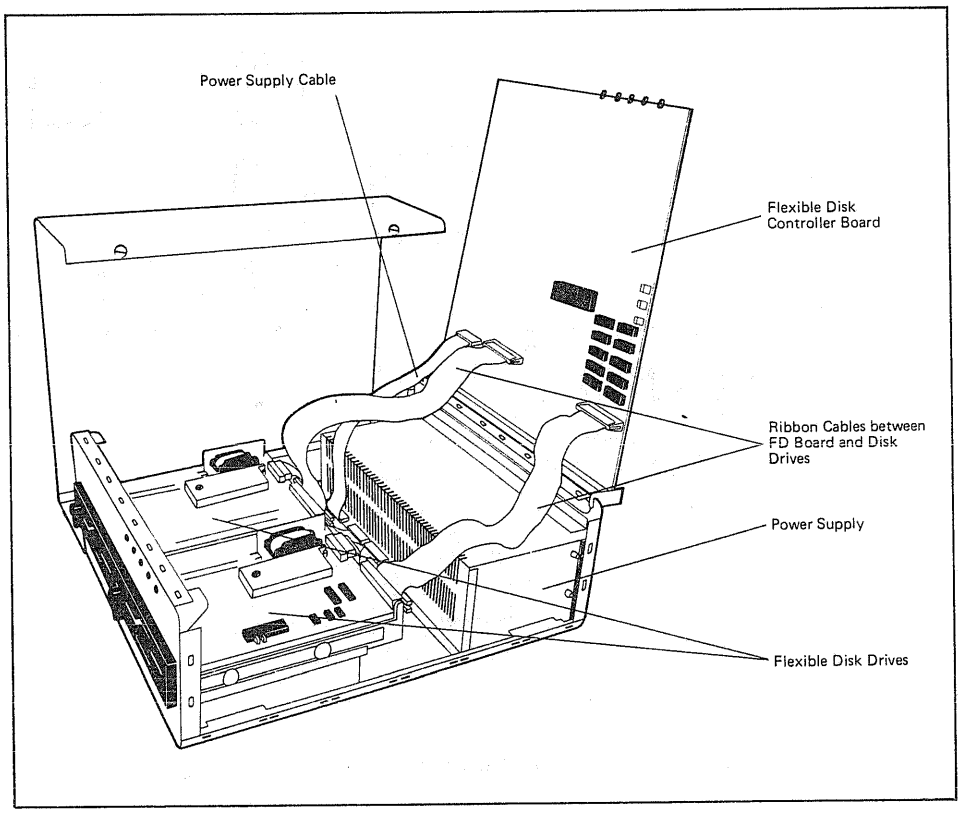


Fig. 1. FD 4122 mechanical layout

Block structure

The electronic circuits for the microprocessor, memory, disk controller and communications controller are all placed on one printed circuit board.
The major function blocks and the bus structure are shown in the coarse block diagram in Figure 2. This is followed by a brief description of the individual blocks.

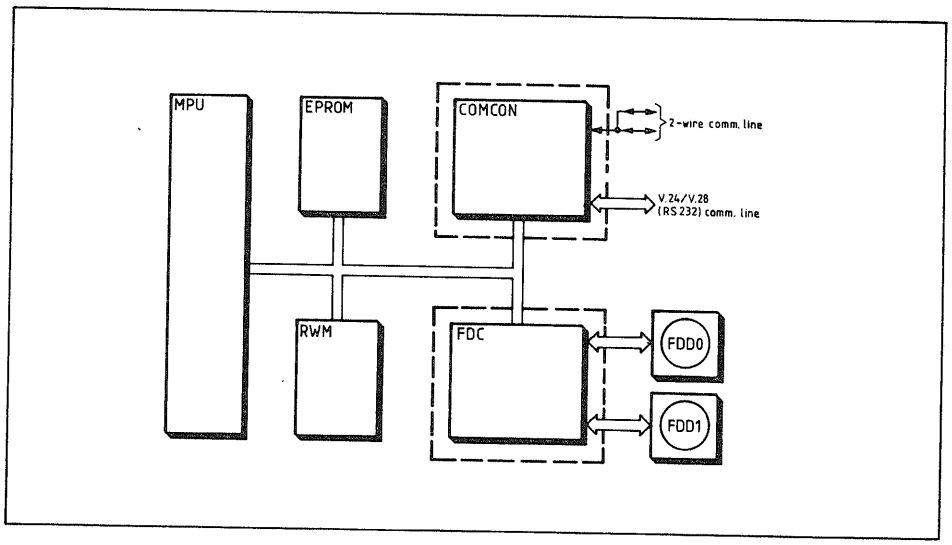


Fig. 2. Coarse block diagram

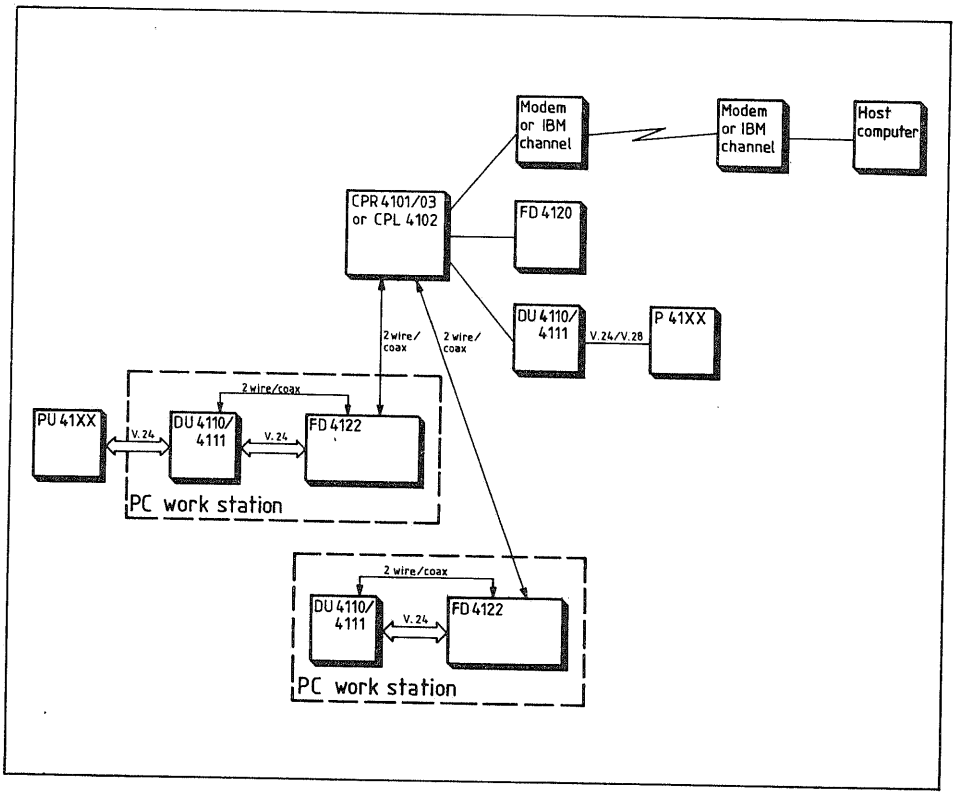


Fig. 3. Alfaskop System 41 with the FD 4122 as mass storage unit

Block Description

Microprocessor Unit, MPU

The MPU consists of an Intel iAPX 188 chip, with buffers and latches. It works primarily according to the programs stored in the EPROM and RWM circuits.

The iAPX 188 includes features like 2 DMA channels, clock generator, programmable interrupt controller, 3 programmable timers, programmable memory and chip select logic and a local bus controller. The central processing unit (CPU) in the iAPX 188 is object code compatible with the iAPX 86, 88 microprocessors and furthermore it adds 10 new instructions to the existing instruction set.

IPL-memory

The IPL memory can be an EPROM chip ranging from 4 to 16 kbytes. It contains the software for bootstrap loading of programs from flexible disk into the RWM.

RWM

The FD 4122 contains either 256 kbytes (or 64 kbytes) of RWM. In the 64 kbytes configuration the FD only works as a system or data FD (SC-mode, SC = shared cluster),that is as a terminal. With 256 kBytes it can be configured either as a personal computer (PC) or in the shared cluster (SC)-mode.

Flexible Disk Interface, FDI

The FDI part of the FD consists of a NEC 765 Floppy Disk Controller (FDC) with associated circuitry to enhance read and write signals and for interfacing the FDI to the floppy disk drives (FDD).

The FDC controls both FDD's and supports either single density format (FM) or double density format (MFM) including double sided recording.

The FDC also provides hand-shaking signals which make it possible to use DMA operation for data transfer between the FDC and the memory.

Additional features provide by the FDC are programmable record length, parallel seek on up to four drives (only two used).

Communications Interface, CI

The CI consists of an Intel 8274 Multi-Protocol Serial Controller (MPSC) chip and associated circuitry to transfer serial data on the two wire line. The transfer rate is 300 kbits per second.

The MPSC contains features like asynchronous, byte synchronous and bit synchronous operation, two full duplex transceivers, 4 DMA channels and handles baudrates up to 880 kbps.

If the FD is to be used as a personal computer an Intel 8254 Programmable Interval Timer, an RS-232 receiver and an RS-232 transmitter are added to the FDPL-board.

The timer contains functions like three 16-bit counters, binary or BCD-counting and six programmable counter modes.

Out Register

The Out Register block contains an eight bit register and five light emitting diodes (LED). Part of the register is used to initiate the LED's and the rest is used to set the signals Disk Drive Motor and Disk Drive In Use.

Address Register

The Address Register consists of an eight pole dip switch, pull up resistors and an octal buffer and line driver.

The Address Register is primarily used to identify a certain system/data FD in a cluster environment in a multidrop configuration.

Microprocessor Unit, MPU

This section deals with the application of the Intel iAPX 188 microprocessor in the Alfaskop Flexible Disk Unit 4122. For a functional description of the iAPX 188 you are referred to the manufacturer's data sheets. All circuit and signal names in this section refer to the block diagram in Appendix A. This appendix can be unfolded, so that it can be used for reference while you are reading the text.

The Intel iAPX 188 microprocessor is internally a 16-bit parallel device with an 8-bit external data bus, with three-state control. It has a 20-bit adress-bus which gives it an addressing capacity of 1 Megabyte. The 8 LSB are multiplexed with the data bus.

It should be noted that not all signals from the iAPX 188 are used in this application. Some signals are connected to hardware inside the MPU-block and some are not connected anywhere. To simplify the understanding of the use of the iAPX in this application we will therefore describe the signals from the MPU block first and thereafter the function of the MPU block. This will include information on registers and addresses in the MPU.

Signals

There are 20 address bus lines from the iAPX. In this application only 18 of these are used. The two most significant lines A18 and A19 are not used.
The 8 LSB of the address bus are multiplexed together with the databus. This is done in such a way that during an address access cycle, first the 8 LSB of the address bus are available on the bus and thereafter the 8 data bits. The timing for this can be found in the suppliers data sheets. The LSB of the address bus and the data bus is marked AB0–AB7. The remainder of the address bus is marked A8–A17.
Apart from the address and data bus lines there are a number of processor and bus control signals which lead out of the MPU-block. There are also some signals that remain inside the block. Some of these are connected to some form of hardware while others simply are not used in this application.

The unused signals in this application of the iAPX 188 are shown in table 1.

Table 1. Unused signals from the iAPX 188 in the FD 4122 application.

Signal	Pin
TMR OUT 1	23
A 18/S5	66
A 19/S6	65
S7	64
$\overline{\text{LOCK}}$	48
$\overline{\text{MCS0}}$-3	38,37,36,35
$\overline{\text{PCS5}}$/A1	31
$\overline{\text{PCS6}}$/A2	32

Connected signals used inside the MPU-block are shown in table 2.

Table 2. Signals used inside the MPU-block.

Signal	Pin	Connection
Vcc	9,43	System power: +5 Volt
Vss	26,60	System ground
X1,X2	59,58	Connected via a 16 MHz crystal
$\overline{\text{TEST}}$	47	Connected via a pull up resistor to system power
TMR IN 1	21	Connected via a pull up resistor to system power
ARDY	55	Connected to system ground
NMI	46	Connected via an inverter and a pull up resistor to system power
HOLD	50	Connected via an inverter and a pull up resistor to system power
HLDA	51	Enables octal line driver for memory and chip select signals, when the signal is low
ALE	61	After passing line driver ALE enables output latch for 8 LSB of the address bus

For testing purposes the following signals inside the MPU-block are furnished with test points (TP).(Table 3.)

Table 3. Signals with test points in the MPU-block

Signal
AD0–AD7
A8–A19
$\overline{S0}$–$\overline{S2}$
$\overline{PCS0}$–$\overline{PCS5}$
$\overline{LCS}$
$\overline{UCS}$
CLKOUT
SRDY
NMI
HOLD
TMR OUT 0
ALE
D $\overline{T/R}$
$\overline{RD}$
$\overline{WR}$
$\overline{DEN}$
$\overline{RESET}$

The remaining signals that are used in this application can be divided in two main groups. One group consists of signals that interrupt the normal program execution and the other group consists of bus control signals.

There are 7 different hardware interrupt signals with different priorities that are used in this application. Their chief characteristic is that they stop normal program execution. The interrupt signals, in order of priority are: $\overline{RESET}$, DRQ0–1 and INT0–3.

Interrupt Signals

- $\overline{RESET}$ going low for at least 4 clock cycles forces the processor into an reset process. This means that the processor immediately terminates the present activity, clears the internal logic and enters a dormant state. After the $\overline{RESET}$ becomes inactive approximately 7 clock cycles pass before the processor begins execution anew. On restart the processor begins execution with the instruction in memory location FFFF0(H). There are also some registers that are set to predefined values after a reset. Information on these can be found in the data sheets for the iAPX 188.
- TMR IN0 can be used either as a clock, (or as a control signal). It is active high and it is connected to the chip Select Request output to the NEC 765A Floppy Disk Controller.
- DRQ0–1 are DMA request signals driven by external devices. They are active high. DRQ0 is driven by the Intel 8274 Serial Controller's Transmit- or Receive-DMA-Request signals. DRQ1 is driven by the NEC 765A Floppy Disk Controller output DRQ via an time delay.
- INT0 is an interrupt signal that is active high. It is driven by software, ie. the operating system via the Out Register on address $\overline{PCS4}$, data line 7 (AD 7).
- INT1 is an interrupt signal that is active high. It is driven by the interrupt signal from the Intel 8274 Serial Controller via an inverter.

- INT2 is an interrupt signal that is active high. It is driven by the interrupt signal from the NEC 765A Floppy Disk Controller output INT.
- INT3 is an interrupt signal that is active high. It is driven by the output OUT 1 on the Intel 8254 Programmable Timer.

Bus Control Signals

The bus control signals that are used can be divided in two groups. One group consists of memory and chip select signals. The other group consists of signals that control data direction, read, write, data enable and adress enable. This group includes status signals, clock signal and synchronous ready signal as well as timer out signal and reset output.

Memory and Chip Select Signals

- $\overline{\text{LCS}}$ is an active low, memory select, to the lower part of the addressable memory (1–256 kbyte). It is used to address the RWM in the FD 4122. The amount of memory to be accessed is selected by software. This signal is also used to control the Refresh Timing during an MPU access of the RWM.
- $\overline{\text{UCS}}$ is an active low, memory select, to the upper quarter of the addressable memory (1–256 kbyte). It is used to address the ROM in the FD 4122. The amount of memory to be accessed is selected by software.
- $\overline{\text{PCS0}}$ is used to drive the Chip Select (CS) on the Intel 8274 Serial Controller. It is active low.
- $\overline{\text{PCS1}}$ is used as an DMA Acknowledge to the NEC 765A FDC. It is active low. The signal can also activate the Terminal Count (TC) on the FDC under certain circumstances.
- $\overline{\text{PCS2}}$ is used to drive the Chip Select (CS) on the FDC. It is active low.
- $\overline{\text{PCS3}}$ is used to drive the Chip Select (CS) on the Intel 8254 Programmable Timer. It is active low.
- $\overline{\text{PCS4}}$ is used to enable the Out Register together with the $\overline{\text{WR}}$ signal and the Address Register together with the $\overline{\text{RD}}$ signal. It is active low.

Control Signals

- DT/$\overline{\text{R}}$ is the data direction signal that controls the direction of the data through the octal bus transciever in the RWM block. When this signal is low, data is transferred from the RWM to the data bus. When it is high, data is transferred from the data bus to the RWM.
- $\overline{\text{RD}}$ is active when the iAPX is performing a memory or I/O read cycle. It is active low. The following devices are connected to this signal: ROM, Address Register, Programmable Timer, Serial Controller and FDC.
- $\overline{\text{WR}}$ is active when the iAPX is performing a memory or I/O write cycle. It is active low. The following devices are connected to this signal: RWM, Out Register, Programmable Timer, Serial Controller and FDC.

- ALE is active during the first clock cycle of a processor cycle. It is guaranteed that the 8 LSB of the address bus are available on the bus, on the trailing edge of ALE. This makes it possible to latch this part of the address into the appropriate device. (See timing diagram in manufacturer's data sheets.) The signal is active high. In this application this signal is connected to the refresh timing block.
- $\overline{DEN}$ enables the outputs of data bus transcievers. It is active low. It does not become active until data has been placed on the bus and the DT/$\overline{R}$ signal has settled in its new state.
- TMR OUT0 enables together with the $\overline{PCS1}$ the Terminal Count (TC) of the FDC. The timer should be set up so that the out signal goes low on terminal count. That means that the signal is considerd active low in this application.
- RESET is active during a MPU reset. It is used for system reset. It is active high but since the FD 4122 also needs an active low reset, such a signal has been included. The $\overline{RESET}$ $\overline{OUT}$ is generated by passing the RESET through an inverter.
- CLKOUT is an 8 MHz system clock signal. It is derived from the 16 MHz crystal oscillator.
- $\overline{S0}$–$\overline{S2}$ are bus cycle status signals. They are active low. For information on these signals you are referred the manufacturer's data sheets. In this application sre used for refresh timing control. (See section on RWM.)
- SRDY is generated by the refresh timing logic. It can put the MPU in a WAIT state while the RWM is refreshed. The maximum delay this signal can cause is 4 clock cycles. It is active high. It is active for approximately 4% of the active time of the running system. That means that it causes an approximate delay in the system of 4%.

Functions

The iAPX 188 consists of 7 main functional blocks. These are the Clock Generator, the Execution Unit, The Programmable Interrupt Controller, the Programmable Timers, the Bus Interface Unit, the Chip Select Unit and the Programmable DMA Unit. All units except the Clock Generator contains registers and controlling logic.
The registers in the different units of the iAPX are of two main types. Either they are control registers or they are data registers except in the Execution Unit. This Unit contains registers of the following categories: General Registers, Segment Registers and Status and Control Registers.

Clock Generator

The Clock Generator contains a crystal oscillator, an divide-by-two counter, asynchronus and Synchronus inputs and reset circuitry.
It provides the system with an 8 Mhz system clock. The Clock Generator also contains the logic that receives the external READY signals. In this application is only the Synchronus Ready Signal (SRDY) used.
This signal is used to insert WAIT states in the execution process. The WAIT states are needed to let the Refresh circuitry access the RWM and

Refresh the dynamic memory at appropriate intervals. The READY control consists of three bits (R0–R2) in the Control Word for each Memory Select or Chip Select line. (See section on Memory and Chip Select.) Since only the Refresh Function needs WAIT states to be inserted through hardware control, only the Low Memory Select Line Register needs special attention. (See section on RWM.)
This means that when the MPU is busy addressing either internal units or other external units than the RWM, the READY control bits can be set so that SRDY is ignored. During MPU access to the RWM, the SRDY on the other hand must be taken into account. Since the RWM used in this application requires extra WAIT states, only two settings of the READY bits remains, namely:

R2	R1	R0	
0	0	1	1 WAIT state, external SRDY used.
1	0	1	1 WAIT state, external SRDY ignored.

Execution Unit

The Execution Unit is an upgraded version of the iAPX 86, 88 microprocessor family. It contains the same basic set of registers, instructions and addressing modes as the 8086 and 8088 MPU's. The upgrading means that the iAPX 188 has 10 new instruction types other than those already present in the iAPX 86, 88 MPU's.
The Instruction Set of the iAPX 188 is enclosed in this manual as an appendix B. For detailed information on the workings of the Execution Unit we refer you to the manufacturer's data sheets. This includes information on the registers, instructions, addressing modes, data types and interrupt vectors.
There are no restrictions placed on the use of the Execution Unit other than those due to the fact that some signals to or from the MPU has been excluded in this application. (See section on Signals.)

Programmable Interrupt Controller

The Interrupt Controller can handle interrupts from inside the MPU as well as external interrupts. Internal interrupts are generated by the timers (3 timers merged into one interrupt line), or by either of the DMA Controllers (2 interrupt lines). The internal interrupts can be masked either in the Interrupt Controller or in the control registers of respective interrupt source.

The external interrupts are brought to the controller on lines INT0–INT3. The source of these is described in the section on signals above. These interrupts are all generated by circuitry external to the MPU. This circuitry is activated by the MPU and therefore the interrupts are in fact software generated. That is they are, or should be expected and there should be program routines that take care of these signals.

The external interrupt signals can either be disabled in the external circuitry or in the Mask Register in the Interrupt Controller.

The Interrupt Controller also contains a register that can be set to disable interrupts below a certain priority level (Priority Mask Register).

The remaining interrupt signal is the Non Maskable Interrupt (NMI). This signal is disabled through an inverter and a pull up resistor. However, it can be enabled by connection to system ground. This is possible since the signal is brought out to a test pin on the printed circuit board. It is only possible to enable this signal when the FD 4122 is dismantled for service or repair. That means that this signal is ideal for initiating test routines that should be available to service or repair personnel only. It is used at production tests.

All interrupts that are accepted by the Interrupt Controller are passed on to the Execution Unit on a priority basis. The priority can either be software selected or follow the default priority table (See manufacturers data sheets).

All interrupts are handled by indirect calls through a vector tabel. The vector tabel is indexed by using the interrupt vector type multiplied by 4.

Programmable Timers

There are 3 programmabel timers in the iAPX 188. Timers 0 and 1 can count external pulses while Timer 2 only can count on pulses from Timer 0 and 1. Timers 0 and 1 can also be set up to count pulses from an internal clock. Timers 0 and 1 can generate Timer Out pulses to external units and also internal interrupts to the Interrupt Controller. Timer 2 can either send an interrupt request to the internal Interrupt Controller or an DMA request to the internal DMA Controller.

All three timers contain one Mode/Control Word Register and one Count Register. Timer 0 and 1 contains two Max Count Registers each, whole Timer 2 only contains one Max Count Register.

In this application only Timer 0 is connected to an external device, namely the FDC. The Timer is meant to be used as a byte counter during DMA transmissions between the FDC and the memory (if the program does not want to read/write whole sectors of data, in which case this function is not used).

To accomplish this the Max Count Register A should be set up with a number N, which is one less than the actual number of transmitted bytes. The Max Count Register B is set up with the value 1. The timer then counts the number of DMA requests till an agreement has been reached between the Max Count Register A and the Count Register. The Timer then switches over the count to Max Count Register B. This causes the Timer to set the signal TMR OUT0 low. This will tell the FDC at next DMA-transfer (by signal TC) that enough data has been sent to memory and it will therfore terminate the DMA transfer.

The timer counts LOW-to-HIGH transitions. Timer 0 registers resides in offset addresses 50H–56H in the Internal Register Map. For a DMA transaction between the FDC and the memory the total amount of bytes to be transferred, minus 1, should be enterd in Max Count Register A, 1 should be entered into Max Count Register B and the following Word should be entered in Control Register 0, with the offset address 56H:

E006H

When the Timer 0 is unoccupied with DMA transfers it can be used together with Timers 1 and 2 for other counting purposes. Note that the Timers only can use an internal clock for such purposes since TMR IN0 is occupied by the FDC and TMR IN1 is not connected in this application.

Bus Interface Unit

The Bus Interface Unit generates the local bus control signals. The signals that are used in this application are ALE, $\overline{RD}$ and $\overline{WR}$ for memory and peripheral control, DT/$\overline{R}$ and $\overline{DEN}$ for data transciever control, HOLD and HLDA for control of the memory and chip select line control and $\overline{S0}$–$\overline{S2}$, SRDY, $\overline{RES}$ and RESET. The signals are described in the section on Signals above.

Memory and Chip Select Unit

The Memory and Chip Select Unit makes it possible for the programmer to generate chip select for both memory and peripheral units.

In this application the Upper Memory Chip Select (UCS) enables the ROM which contains bootstrap loader and assorted test programs. The address lines brought to the ROM-chip carrier makes it possible to address either 8 or 16 kbytes of data. Since the upper limit of the memory always is FFFFFH the lower limit must be programmable. In the FD 4122 application this means that for 8 K respectively 16 K of ROM the following values should be placed in the UMCS Register at offset address A0H:

8 K – FE3DH
16 K – FC3DH

The values above take into account the use of memory that needs 1 WAIT state and it ignores the SRDY signal. (See manufacturers notes on READY Bits Programming. R2=1,R1=0,R0=1) The current EPROM version needs 1 WAIT state.

The Lower Memory Chip Select (LCS) can either contain 64 K or 256 K of data depending on which version of the FD 4122 you are dealing with. Since the lower limit of the LCS-part of the memory always is 0H the upper limit must be programmable. This gives an upper address, for the system/data FD version of 0FFFFH and for the personal computer version 3FFFFH. To define this limit the following values should be placed in the UMCS Register:

64 K – 0FF9H
256 K – 3FF9H

The values above take into account the use of memory that needs 1 WAIT state. The values also acknowledge the SRDY signal, which will insert WAIT states in the execution to give the refresh circuitry time to complete initiated refresh cycles in the RWM. (See manufacturers notes on READY Bits Programming. R2=R1=0,R0=1).

The Mid Range Memory Chip Select is not used in this application.

The FD 4122 uses 5 of the available chip select lines, namely $\overline{PCS0}$–$\overline{PCS4}$. The start address of the Peripheral Chip Select Block is defined by the PACS Register. The PACS Register resides at offset address A4H. If the block is located in I/O space bit 12–15 of the Programmable Base Address (PBA) must be programmed zero. Bits 6–11 corresponds to bits 10–15 of the 16 bit PBA. The I/O space is only 16 bits wide. Each Peripheral Chip Select Block occupies 128 bytes of memory. These blocks are placed contiguous in I/O space. This means that the PBA only can be a multiple of 1 kbytes since there are 7 blocks of 128 bytes, which equals 896 bytes of memory. From this follows that the 10 LSB of the PBA should be programmed as zero.

This Register not only contains the PBA of the I/O space. It also contains The READY bits R0, R1 and R2, which control the number of WAIT states inserted in each I/O access. These control bits have to be set separately for each I/O unit. Since the SRDY signal has no connection with I/O access it should be ignored during I/O access cycles. This means that the R2 bit always should be one (1) during I/O access.

To select the working mode of the PCS the MS bit of the MPCS Register, which also sets the size of the mid range memory chip select, is set to one to place peripheral chip select in memory space and set to zero to place it in I/O space.
When the final addressing of a peripheral unit is done the PCS line chooses the unit and the seven (7) LSB of the address bus selects addresses inside the block. Here follows a listing of the PCS lines and the I/O units that they enabel. Also included are the address lines that are used by each unit.

Table 4. PCS Lines to I/O Units.

PCS Line	I/O Unit	Address Lines
$\overline{PCS0}$	Serial Contr., CS	A0,A1
$\overline{PCS1}$	FDC, DMA ACK	A0
$\overline{PCS2}$	FDC, CS	A0
$\overline{PCS3}$	Progr. Timer, CS	A0,A1
$\overline{PCS4}$	Register Out, CS	

Programmable DMA Unit

There are two independent DMA channels in the iAPX 188. In this application channel 0 is connected to the Serial Controller and channel 1 to the FDC.

Each channel contains 6 registers. There is a Control Register, a Transfer Count Register, two Destination Address Pointers and two Source Address Pointers. The offset address of these registers can be found in the DMA Control Block Format Table in the manufacturer's data sheets. The Transfer Counter defines the number of DMA transfers up to a maximum of 64 K transfers of 1 byte. The Destination and Source Pointers define the addresses for the Destination and Source of a DMA transfer, in memory or I/O space. They can be individually incremented or decremented after each transfer. The DMA controls the memory and chip select lines of the MPU in the same way as the Execution Unit does.

The Control Register controls the workings of the DMA Controller. It should be set up, according to the DMA Control Word Bit Descriptions in the manufacturer's data sheets, before a DMA transfer is requested by an external unit.

The FDC needs a DMA Acknowledge signal (DACK) to function properly. This signal is provided through placing the address for the PCS1 line in the Destination or Source Pointer of the DMA and sending this with a Write or Read signal to acknowledge that the FDC has requested a DMA transfer. The Source or the Destination Pointer is set to point to the appropriate start address in the RWM, with the Control Register set to increment or decrement automaticly after each DMA transfer. The placement of the PCS1 address and the RWM start address in Source or Destination pointers depends on whether the FDC shall perform a DMA Read or Write Cycle.

EPROM Memory

The FD 4122 has space reserved for one EPROM chip only. The chip is placed in a 28 pin DIL chip carrier. The pinout connections of the chip carrier makes it possible to use 2732, 2764 or 27128 or pin compatible EPROM's to store the Initial Programs like Bootstrap Loader and a variety of test programs. The 2764 EPROM gives 8 kbytes of program memory and the 27128 gives 16 kbytes. There are no provisions on the FD 4122 for programming of the EPROM chips, that is there is no programming voltage available.

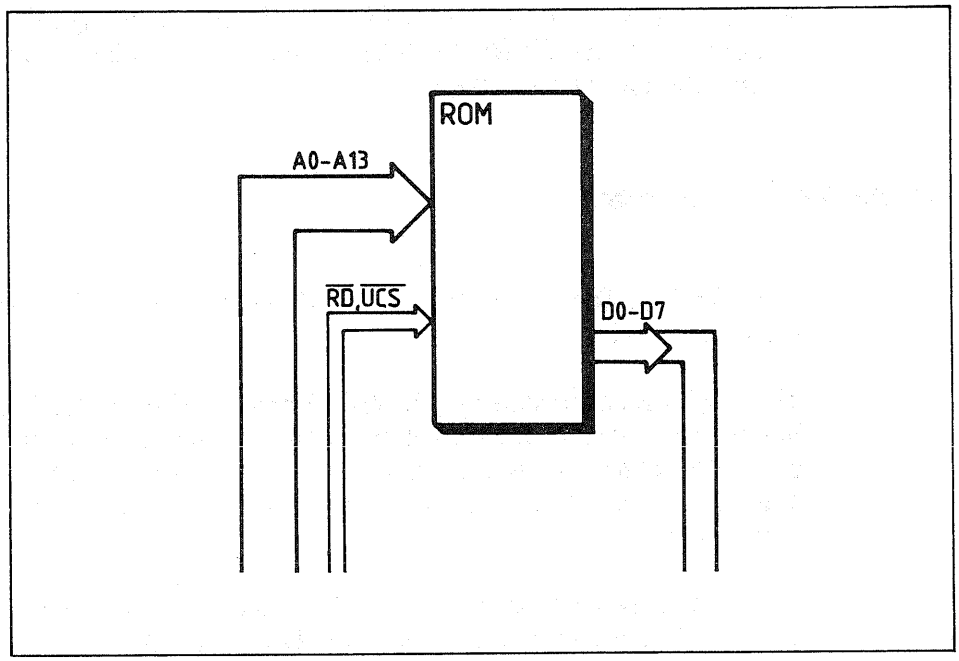


Fig. 4. ROM signals

Signals

The signals to and from the EPROM block can be divided in three main parts. First there are the adress lines, then the data lines and finally the control lines. Apart from these there are also System Power (Vcc) and System Ground (Vss). The $\overline{PGM}$ line is connected to System Power via an pull up resistor and the Program Voltage line (Vpp) is connected to Vcc.

- A0–A13 are the address signals. It should be noted that the lines A0 through A7 are latched so that they carry the adress during the whole execution cycle. The signal A13 is not connected to the proper pin in the standard configuration, that is with the 2764 EPROM in the socket. Instead the line from the chip carrier is connected to System Power. The A13 ends in a jumper position and in case the EPROM is changed to a 27128, then the A13 can be jumper connected to the chip. The original connection to System Power must be cut.
- AD0–AD7 are the multiplexed data and address lines. The timing in the EPROM makes sure that the data outputs from the EPROM are valid for enough time to give the MPU time to read the data bus. (For timing diagrams you are referred to the manufacturer's data sheets.)
- $\overline{CE}$ is generated by the signal $\overline{UCS}$ from the MPU. The amount of memory that this signal enables is software programmable. (See Section on the MPU above.) The EPROM resides in the upper part of the system memory, with the last byte of the EPROM in position FFFFFH whether it uses 8 K, 16 K or 256 K bytes of the upper memory area. The $\overline{CE}$ can also be disabled manually or through external test equipment by pulling the Test Point (TP) in the EPROM block low (grounded). This provision enables test equipment to run a test program in another PROM without the need to remove the EPROM.
- $\overline{OE}$ is generated by the MPU signal $\overline{RD}$. This signal enables the outputs of the EPROM. The timing for this can be found in the manufacturer's data sheets.

Read/Write Memory

The FD 4122 RWM consists of 256 K bytes of dynamic memory when fully equipped.

The RWM is controlled by a RWM Controller (DP 8408) from National Semiconductors. This Controller drives both memory configurations. It generates read and write signals to the memory, it controls column addresses, row addresses and bank select and it controls the refresh addressing.

Data transmission to or from the RWM is controlled separately through an transciever circuit controlled directly by the MPU.

The RWM refresh timing is generated in a separate hardware block. This block sends the refresh enable signal as well as a row address strobe signal to the RWM Controller. It also generates a SRDY signal that forces the MPU to place WAIT states in the execution cycle. This is done to allow the RWM Controller to finish an initiated refresh cycle before the MPU addresses the RWM again.

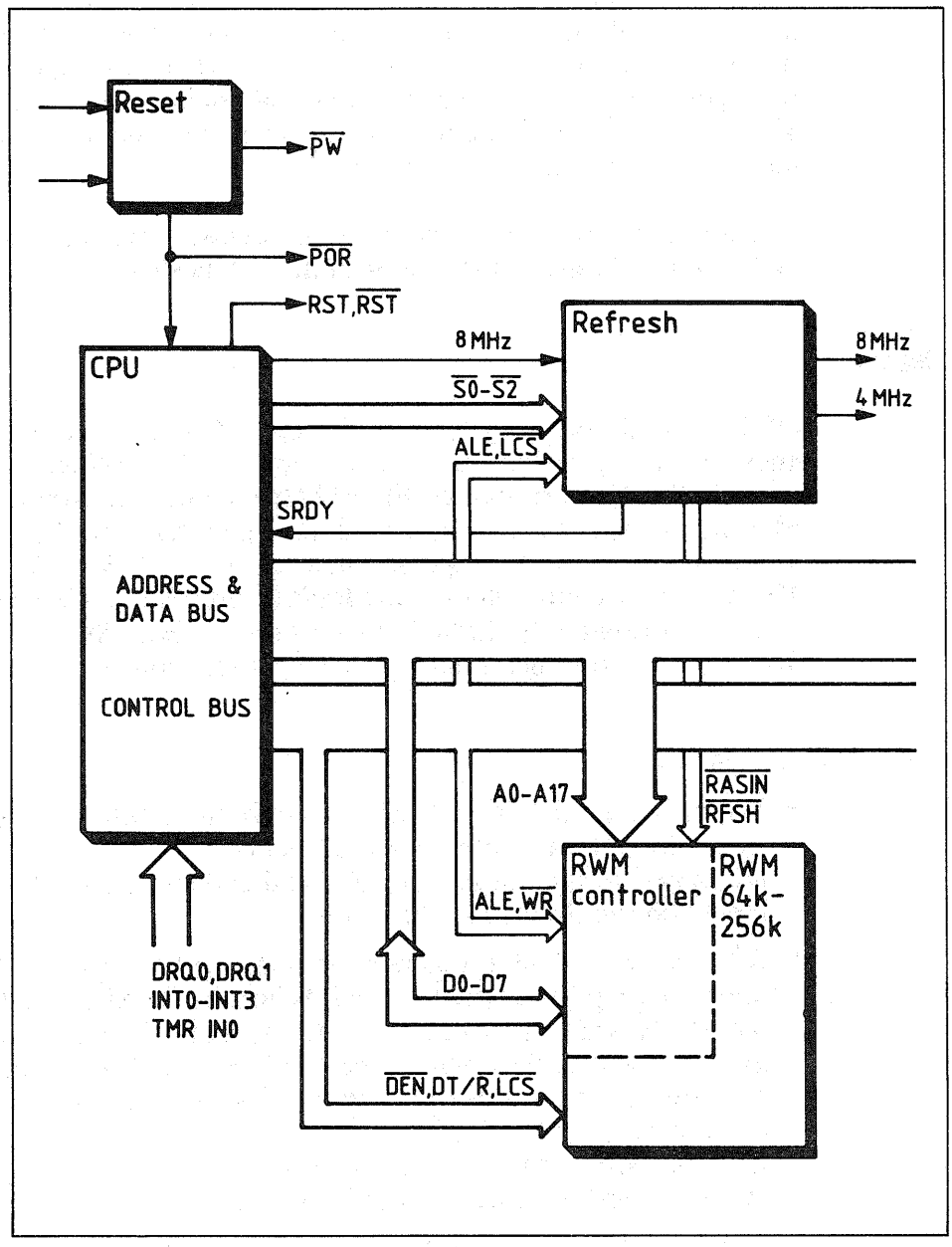


Fig. 5. RWM and Refresh signals

RWM Controller and Memory Block

The RWM Controller has two main functions in this application. Primarily it generates the proper signals that select memory bank, row and column, of the memory cell requested by the MPU. Secondly it generates the row address for the refresh cycles in aninternal 8 bit counter.
During a memory read or write cycle the RWM Controller generates the proper row and column addresses and latches these to the proper memory bank with the $\overline{RAS}$ and the $\overline{CAS}$ signals.
If the MPU requests a memory write the Controller enables the Write Enable ($\overline{WE}$) signal on the memory chips.
During a refresh cycle the RWM-controller generates the row address in an internal counter and then enables the row in all the memory banks, whereupon the row is refreshed. The counter is then updated.

The RWM Controller can operate in any of 8 modes. In this application the number of modes has been reduced to two. This has been achieved through connecting the mode lines M0 and M1 to system power respective system ground. The result is that only modes 1 or 5 remain active. The switch between these modes is controlled by the $\overline{\text{RFSH}}$ signal. (See table 1 in manufacturer's data sheets.)

The timing for both memory cell read access, write access and refresh cycles can be found in the manufacturer's data sheets.

Signals

Of the 20 address lines generated during a MPU access cycle the first 16 lines (A0–A15) are used to select a memory cell in a 64 K byte memory bank. In the PC version of the FD 4122 the memory consists of 4 banks of 64 K bytes each. To select those address lines A16 and A17 are used. There are only 3 control signals connected to the RWM Controller. The remaining control lines to the RWM Controller has been enabled or disabled permanently through connection to either System Ground or System Power through a pull up resistor. (See table 5.)

RWM Control Signals

- $\overline{\text{WE}}$ or WIN is the Write Enable Input of the Controller. The signal is transmitted to the buffered Write Enabel Output of the Controller. It is enabled by the output control signal and enables the Write Enable inputs on the RWM memory chips.
- $\overline{\text{RASIN}}$ has a double function. It is derived from the MPU Lower Memory Chip Select ($\overline{\text{LCS}}$) signal. When the $\overline{\text{RASIN}}$ signal is low and the $\overline{\text{RFSH}}$ is high the Controller addresses the Memory Banks according to the state of the address signals A16 and A17. This means that the contents of the Bank Select latch decides which memory bank should be addressed by the $\overline{\text{RAS0}}$–$\overline{\text{RAS3}}$ lines.
  If the $\overline{\text{RASIN}}$ and the $\overline{\text{RFSH}}$ goes low all the $\overline{\text{RAS}}$ outputs go active at the same time for a memory refresh cycle.
- $\overline{\text{RFSH}}$ or M2 going low initiates a memory refresh cycle. The M2 signal actually selects in which mode the Controller should work. From the Mode Select Tabel in the manufacturer's data sheets it can be seen that with the M0 tied to ground and M1 tied to System Power the Controller has only two modes in which to operate.
  With M2=0, the Controller is put into an external refresh mode (mode 1). The $\overline{\text{RFSH}}$ signal itself initiates the refresh cycle. First the contents of the Refresh counter are placed on the lines Q0–Q7 and are latched into the RWM. Then, due to that $\overline{\text{RASIN}}$ goes low at the same time as $\overline{\text{RFSH}}$, all the RAS lines are enabled simultaneously and thereby a whole row of the memory is refreshed in one stroke. The Refresh Timer is set up to deliver one refresh cycle every 14 microseconds which means that the whole memory is refreshed in approximately 4 milliseconds.
  With M2=1, the Controller is placed in a AUTO memory access mode (mode 5). In this mode all outputs from the Controller are initiated by the $\overline{\text{RASIN}}$ signal, except the $\overline{\text{WE}}$ signal. The use of this mode reduces timing problems that otherwise can occur during memory access.

Data Direction Control

The Data Direction Control consists of an 8 bit transceiver which controls the flow of the data placed on the data bus by either the MPU or the RWM cells during a memory read cycle. The enabling of the transciever and the direction control is governed by 3 MPU signals.

- $\overline{LCS}$ is the Low Memory Chip Enable signal which also enables the RWM controller through the Refresh Timer signal $\overline{RASIN}$. It is active low.
- $\overline{DEN}$ is the Data Enable signal from the MPU. It is active low. During a write cycle it is activated before data has been placed on the multiplexed address/data bus. During an memory or I/O read cycle the $\overline{DEN}$ signal goes low before the MPU tries to read the data placed on the data bus, but after the 7 LSB of the address bus have been active. That is, while the address/data bus is in its floating state.
- DT/$\overline{R}$ is the data direction signal from the MPU. It controls the flow of the data through the transciever. When this signal is high, the MPU performs a Write cycle. When it is low the MPU reads data from the data bus.

For the timing of these signals you are referred to the manufacturer's data sheets.

Table 5. Enabled or disabled RWM input control signals.

Signal	Pin	Connection
CASIN	2	Via pull up resistor to System Power
$\overline{ADS}$	6	Via pull up resistor to System Power
R/$\overline{C}$	1	Via pull up resistor to System Power
M0	3	Via pull up resistor to System Power (M0=1)
$\overline{M1}$	4	To System Ground (M1=0)
$\overline{CS}$	47	To System Ground
RF I/O	46	Not connected.

Refresh Timing Circuit Block

The Refresh Timing Circuit Block actually has two main purposes. Primarily it provides the RWM Controller with the Row Address Strobe Input signal ($\overline{RASIN}$) and the Refresh signal ($\overline{RFSH}$). It also generates the Synchronus Ready signal (SRDY), which places the MPU in WAIT state until the Initiated RWM Refresh cycle is complete. The second purpose of the Timing Block is to provide the FD 4122 with an alternate clock frequency of 4 MHz.

The Refresh Timer is set up to deliver a refresh request every 14 μs. If no MPU Low Memory access occurs simultaneously or is already initiated when this happens the RWM Controller then performs a memory refresh. While the RWM is refreshed the Refresh Timer enables the SRDY signal which places the MPU in a WAIT state if it tries to perform a RWM access. The SRDY can hold the MPU for a maximum of four system clock cycles, which amounts to 500 ns. This means a maximum delay in the system of 4% of the execution time.

The $\overline{\text{RFSH}}$ signal is latched in the Timer so that when it has been initiated and the MPU is performing a RWM access, then it remains active until the MPU cycle ends. When the MPU RWM access cycle ends the refresh signal then slips in the SRDY to halt the MPU if it attempts to address the RWM again. The $\overline{\text{RFSH}}$ signal then initiates the refresh cycle in the RWM.

It should be noted that the SRDY only halts the execution when the MPU tries to perform a RWM access. If the MPU accesses other parts of the memory or I/O and the READY Control Bits (R0–R2) have been set properly, then the MPU ignores the SRDY signal. (See section on MPU Memory Chip Select.)

Refresh Timing Signals

The signals concerned with the Refresh Timing Circuit Block can be divided in input and output signals. The input signals are CLKOUT, ALE, $\overline{\text{LCS}}$ and $\overline{\text{S0}}$–$\overline{\text{S2}}$. The output signals are 8 and 4 MHz, $\overline{\text{RASIN}}$, $\overline{\text{RFSH}}$ and SRDY.

Refresh Timing Input Signals

- CLKOUT is the system clock signal of 8 MHz. It is used for timing of the Refresh Request, the Row Address Strobe Input signal and the SRDY signal. It is also passed through a divider and driver to generate a 4 MHz signal.
- ALE is the Address Latch Enable signal. It is used together with the $\overline{\text{LCS}}$ signal to set a latch, which is activated whenever the MPU is performing an access to the lower part of the memory (RWM). The resulting signal blocks the Refresh Request until the MPU access cycle ends. The ALE is active high.
- $\overline{\text{LCS}}$ tells the Refresh Timer whether the MPU is performing an access to the RWM or not. It is active low.
- $\overline{\text{S0}}$–$\overline{\text{S2}}$ are the Bus Cycle Status lines. In this application their sole purpose is to set a latch which will activate the $\overline{\text{RFSH}}$ signal if a Refresh Request has been generated by the Timer. The latch is set when and only when all 3 signals are high, including a fourth signal that is derived from the $\overline{\text{LCS}}$ with the ALE. That is $\overline{\text{S0}}=\overline{\text{S1}}=\overline{\text{S2}}=1$, which indicates that the bus is passive. This means that the MPU has finished an access cycle. The time that the bus is inactive after a bus cycle gives the Refresh Timer a chance to slip in the SRDY to halt the execution and at the same time initiate a refresh cycle in the RWM. This is illustrated in the timing diagram in figure 6.

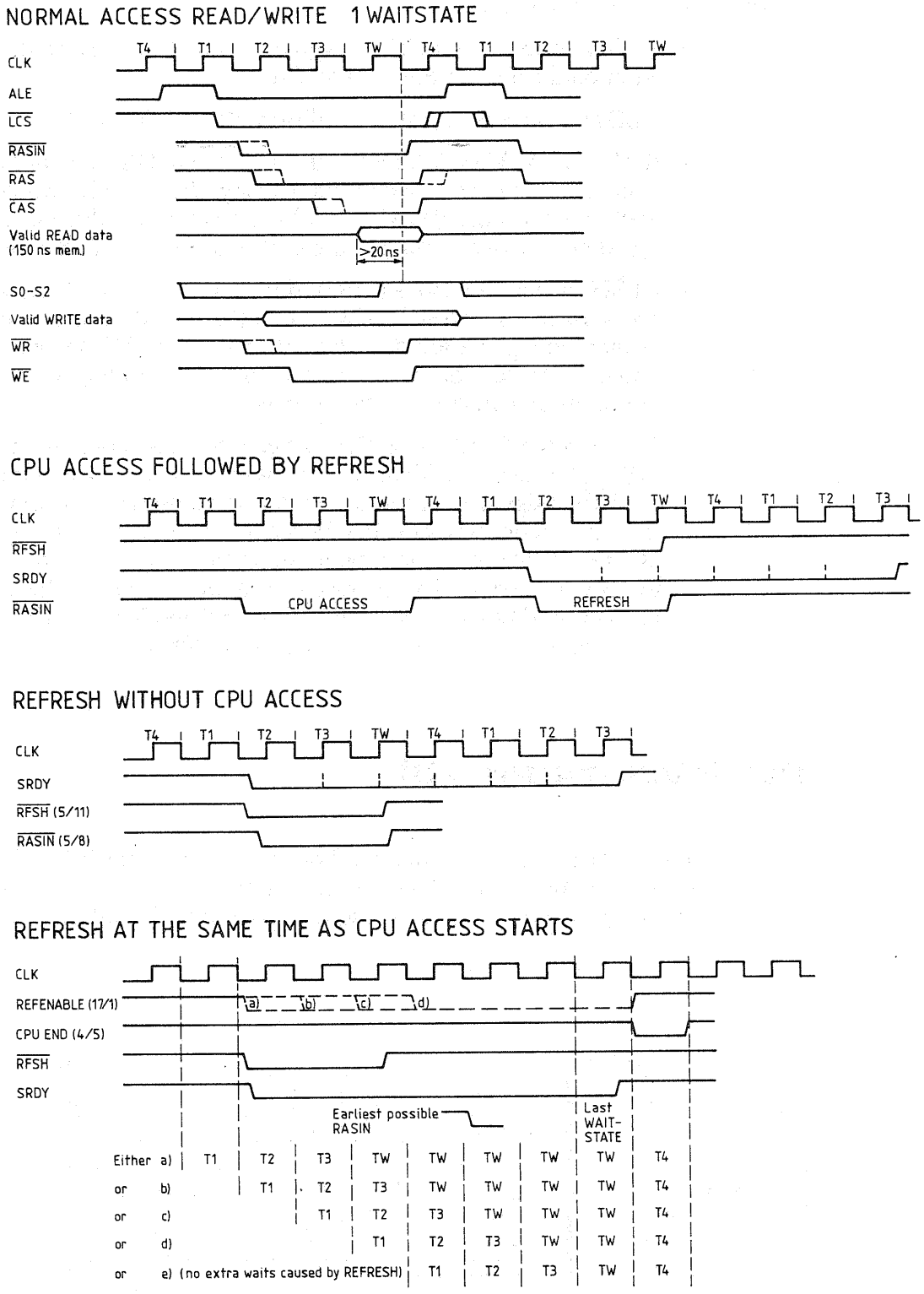


Fig. 6. Timing diagram for refresh circuits

Refresh Timing Output Signals

- 8 MHz is the system clock signal. It has been inverted and given additional drive capacity in the Timer.
- 4 MHz is a clock signal derived from the system clock.
- $\overline{\text{RASIN}}$ is the Row Address Strobe Input signal that initiates the RWM Controller. It goes active on the trailing edge of ALE every time the MPU accesses the RWM. It is also activated shortly after the $\overline{\text{RFSH}}$ signal goes active to initiate a memory refresh cycle. It is active low.
- $\overline{\text{RFSH}}$ is the Refresh Request signal. It is activated approximately every 14 μs to refresh a new row in the RWM. It is active low. For proper timing it should go low after the $\overline{\text{RASIN}}$ has gone high. It remains low for the whole refresh cycle. (See Diagram on Refresh timing in manufacturer's data sheets.)
- SRDY is the Synchronus Ready signal that halts the execution of the MPU. It inserts WAIT states in the MPU bus cycle. This is done to give the $\overline{\text{RFSH}}$ signal a chance to complete the initiated refresh cycle before the MPU requests a new RWM access. If the MPU is set up to access other parts of the memory than the RWM, it is possible to let it ignore the SRDY signal. To accomplish this the 3 LSB of the chip select register used has to be set to R2=1, R1=0 and R0=0. (See Tabel 12 in manufacturer's data sheets on the iAPX 188.)

Flexible Disk Interface, FDI

As mentioned in the Section on Block Structure for the FD 4122, the FDI consists of three main blocks. These are the Flexible Disk Controller, the circuitry to enhance read and write signals and the Interface circuitry.

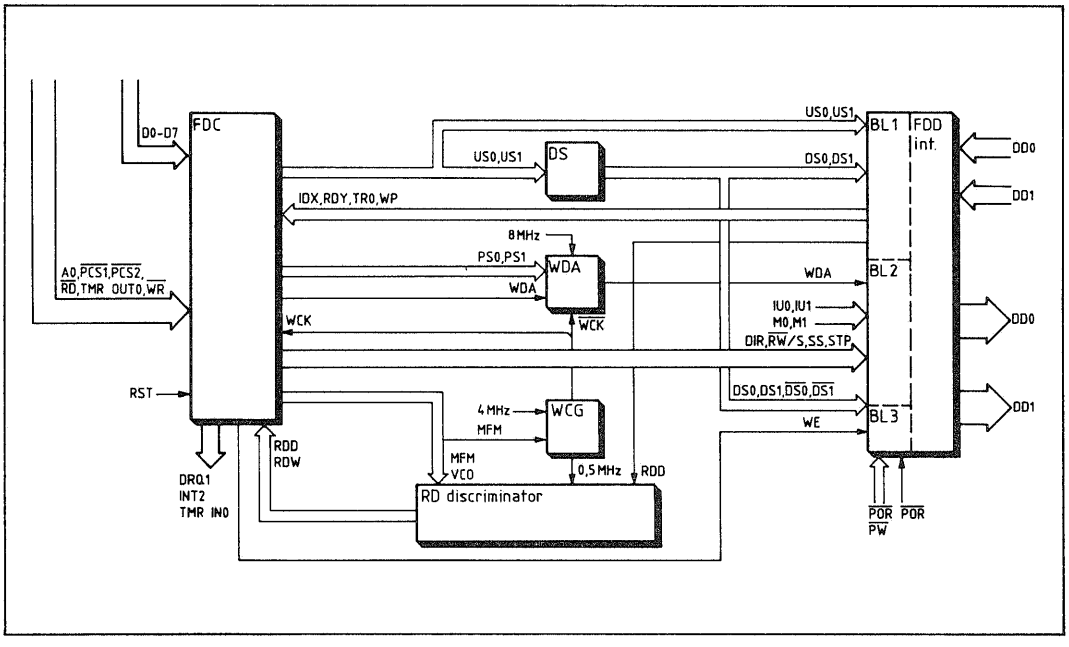


Fig. 7. Flexible Disk Controller and Interface signals

The FDI interfaces the FD 4122 to the Flexible Disk Drives (FDD), either in FM or in MFM. The FDD bus configuration supports either of these, but the FD 4122 is normally set up in the MFM format. This means that it uses double density and double sided recording of data. The FDD bus is duplicated but the FDD's logically "daisy-chained". This simplifies hardware connections for the disks.

Flexible Disk Controller, FDC

The Flexible Disk Controller is the NEC μPD765A or Intel 8272A chip. It is a disk controller which contains features like hand-shaking signals for DMA operation and a set of 15 specific commands, which the FDC will execute.

The instructions are sent from the MPU as multiple 8-bit Byte groups. These groups set up the FDC to perform a specific command. For detailed information on the Flexible Disk Controller you are referred to the manufacturer's data sheets.

The Interfacing circuits as well as the data read and write enhancing circuits place some restrictions on the use of the FDC.

The FDC can access 4 drives but there is a hardware Drive Select Decoder, which only recognizes drives 00(Bin) and 01(Bin), as acceptable disk drives.

The Write Precompensating signals PS0 and PS1 are connected through jumpers to an Write Data Pre-Compensation Unit which selects if the Write Data (WDA) signal is to be placed on the Interface Bus early, late or at a normal time during the Write cycle.

The Write Clock Generator is set up to generate a 0.5 MHz clock signal when the FDC is placed in MFM mode and 0.25 MHz when the FDC is in FM mode.

The Write Enable (WE) signal is used together with the Power On Reset, the Power Warning and the decoded Drive Select lines DS0 and DS1.

Signals

The signals to and from the FDC can be divided in two groups.
The first group contains system signals. That is signals that are received from or sent to the MPU, or other sources like the Power Warning ($\overline{PW}$) and Power On Reset ($\overline{POR}$) signals from the Reset Block.
The second group consists of signals sent to or received from the interface circuitry or the read and write enhancing circuitry.

System Signals

The system signals consists mainly of signals that control the FDC. There are only two signals sent from the FDC to the MPU. These are the DMA Request signal and the Interrupt signal. The 8-bit Data Bus is bidirectional so that data can be passed to the FDC for initiation or as data to be written to or read from the Flexible Disks. Data is also passed from the FDC to the MPU when the MPU performs Read Status Register instructions.

- 8 MHz system clock signal, clocks an 8-bit serial shift register that is incorporated to provide the FDC with a DMA Request signal. The delay is set to 7 clock cycles. This is done since the FDC-chip must not be accesed to soon after having issued a DMA Request.
- 4 MHz is a system clock that is used as a CLK input to the FDC.
- RESET is the System Reset signal from the MPU. It is active high. This signal places the FDC in an idle state.
- Vcc is the System Power signal (+5 V).
- Vss is the system ground signal (0 V).
- $\overline{RD}$ is the system read signal, which the FDC uses together with the $\overline{WR}$ and the A0 signals to set the status for the Data and the Main Status Register. It is active low. The legal settings of these signals can be found in the manufacturer's data sheets. It is obvious that the lines $\overline{WR}$ and $\overline{RD}$ cannot be active simultaneously, a fact which reduces the possible combinations of these signals to 6.
- $\overline{WR}$ is the system write signal. For use with the FDC see description of $\overline{RD}$ signal. It is active low.
- A0 is the least significant line of the Address Bus. It is separated from the multiplexed data line in the MPU block and is latched so that it stays active for a complete bus cycle. It is active high. In the FDC it is used together with $\overline{RD}$ and $\overline{WR}$ signals as described above.
- $\overline{PCS2}$ is the Chip Select signal that enables the FDC. It is active low.
- $\overline{PCS1}$ is the DMA Acknowledge signal from the MPU. It is sent to acknowledge the DMA Request signal every time the FDC requests a DMA transfer, that is for every byte that is to be transferred over the Data Bus. This signal is used together with the $\overline{TMR}$ $\overline{OUT0}$ signal from the MPU to enable the Terminal Count (TC) input line on the FDC. It is active low.
- $\overline{TMR}$ $\overline{OUT0}$ is the Terminal Count signal from the MPU. It enables the TC line of the FDC if the $\overline{PCS1}$ signal is active at he same time. The $\overline{TMR}$ $\overline{OUT0}$ signal is active low. Since this signal and the $\overline{PCS1}$ signal, are passed through an inverter the resulting TC signal becomes active high.
- AD0–AD7 are the system data lines. They are used for carrying data to and from the FDC. The multiplexed address lines are not used by the FDC since the FDC has no provision for separating the Address signals from the Data signals. The reading and writing of data to and from the FDC is controlled by the $\overline{RD}$ and the $\overline{WR}$ signals.

Interface Signals

The Interface Signals are signals to or from either the Interface Circuitry or the read and write data enhancing circuitry. The signals can be divided into Control Signals and Data signals.

- US0 and US1 are the FDD Unit Select lines. Since the FD 4122 only carries two Flexible Drives only the Drives 00(Bin) and 01(Bin) can be selected. These signals are passed to the Drive Select Decoder Block (DSD) which generates the Drive select lines. They are also passed to a multiplexer in the Interface circuitry. which selects from which drive the index, read data acknowledge, drive status(ready) and the write protect signals are to come from.
- PS0 and PS1 are the Write Pre-Compensation Shift signals that control whether the written data is to be placed on the Bus to the FDD's at an early, normal or late time during the write cycle.
- WE is the Write Enable signal, which enables the write heads in the FDD's. It is active high. This signal, the $\overline{\text{POR}}$, the $\overline{\text{PW}}$ and the decoded Drive Select (DS0–1) lines are used to create two Write Enable lines. These are guarded against premature initiation while the power is turned on and against the hazard of power failure.
- SS is the Side Select or Head Select line. When the SS is high, side 1 (head 1) is selected and when the SS is low, side 0 (head 0) is selected.
- STP is the signal that moves the Drive head to a new cylinder on the diskette. A positive pulse means that the head should move one track in the direction determined by the DIR signal.
- DIR is the signal that determines direction of travel for the read/write head, when the FDC is in Seek mode. It is active high.
- $\overline{\text{RW}}$/SEEK is the mode indicating signal. When the signal is high it indicates that the FDC is in Seek mode. When it is low it indicates that the FDC is in Read/Write mode. This fact is used in the Interface circuitry to enable different control signals in different modes. See section on Interface Circuitry below.
- VCO is the signal that enables or disables the Voltage Controlled Oscillator. It is active high, which means that the VCO is enabled when the signal is high.
- MFM is the signal that indicates whether the FDC is in FM or MFM mode. It is set to MFM mode when the line is high. This signal controls the Write Clock Generator Block (WCG) so that the output from this block, the WCK signal, is either 0.5 MHz (MFM-mode) or 0.25 MHz (FM-mode).
- IDX is the Index line which tells the FDC when the beginning of a track has been found. This information is used to initiate the specified command, i.e. the FDC starts to read from the track or write to the track after the Index signal has become active. The signal is active high.
- TR0 is the signal that informs the FDC when the zero track has been found. This information is stored in Status Register 2. It is active high.

- WP indicates Write Protect status in Read/Write mode and two sided media in the Seek mode. This information is placed in the Status Register 2. The signal is active high.
- RDY indicates whether the FDD is ready to send or receive data. This information is stored in the Status Register 2. The RDY signal is derived from the Drive Status line from the FDD's.
- WCK is the Write Clock signal generated in the Write Clock Generator. When the FDC is set up in MFM mode this signal has a frequency of 0.5 MHz and in FM mode it has a frequency of 0.25 MHz. In both cases the pulse width is 250 ns. • RDW is the Read Data Window signal generated by the Phased Locked Loop (PLL) in the Read Clock Discriminator block (RCD). The RDW gives a time window during which the FDC is set to sample data from the FDD. • WDA is the Write Data line on which the serialised data is sent to the FDD. The data is primarily sent to the WDP to be shifted according to the precompensation rules in the FDC. The WDA line also carries clock bits to the FDD.
- RDD is the Read Data line from the FDD after it has passed through some enhancing latches in the RCD. The RDD signal is responsible for generating the RDW signal with the aid of assorted other signals. See section on Read Clock Discriminator below.

Read Clock Discriminator

The Read Clock Discriminator Takes in the RDD signal and puts it through some latches to stabilize the signal. This signal is then fed to the FDC as the RDD signal. Another part of the RCD contains a Phased Locked Loop, (PLL) which generates a data window of variable length. This window opens the read window in the FDC which then reads the RDD signal. The RDD signal must be placed approximately in the middle of the Window. Since the Data has been stored on the disk at varying distances and the diskette rotation speed can vary, the data bits read from the diskettes can come at irregular time intervals. The PLL compensates for this shift and places the window centered around the data signal. The PLL is enabled by the VCO signal from the FDC. There are also a 0.5 MHz clock signal from the WCG, that delivers the central clock frequency to the PLL. The MFM signal is used to select which signal from the PLL that shall be used as the RDW signal.

Signals

- RDD is the input signal from the Interfacing Circuitry. It is active high. It is enhanced in the RCD and then retransmitted. It is also used to generate the phase shift signal that controls the PLL.
- RDD is the retransmitted RDD signal. It is sent to the FDC. The difference between the input RDD and the output RDD is that the output signal has received a short time delay and also a defined length.
- VCO is the enable signal for the VCO from the FDC. It enables the VCO when it is high and inhibits it when the signal is low.
- MFM is the MFM or FM mode select line. MFM mode is selected when the signal is high.

- 0.5 MHz is the PLL central clock signal. It is generated in the VCG BLock.
- RDW is the Read Data Window signal that opens the FDC read data input, so that the FDC can read the RDD signal. The leng of the RDW signal is selected by the MFM signal so that the proper window size is used for the RDW signal, whether the FDC is in MFM or FM mode.

Write Data Pre-Compensation

The Write Data Pre-Compensation Unit (WDP) actually consists of two hardware blocks. One is the WDP Unit and the other is the Write Clock Generator (WCG). The WCG has two purposes. It generates the 0.5 MHz signal that the RCD uses as the central frequency for the PLL. It also generates the Write Clock (WCK) signal that enables the WDA signal in the FDC and also it is used in the WDP block to shape the WDA signal. The WCG is controlled by one clock signal and one control signal from the FDC.

The WDP's sole purpose is to send the WDA signal through the Interface Circuitry to the FDD's. The WDA signal is pre-compensated, that is, when it is placed on the bus to the Interface it has been delayed according to rules set in the FDC. These rules are conveyed to the WDP by the PS0 and PS1 signals. The PS0 and PS1 control a multiplexer in the Write Data Pre-Compensation Block (WDP). They are connected to the multiplexer through a set of jumpers. The jumpers are labeled L5, L6 and L7 and the three different possible combinations of these are presented in table 6. The signals are active high. They are set in the Main Status Register. The setting of the multiplexer chooses the appropriately timed WDA signal from a shift register and sends it to the FDD's.

Table 6. Write Data Pre-Compensation jumpers.

Connected Links	Pre-Compensation	Delay	Sort
L5 + L7	125	125	ns
L7	250	250	ns
L6	0	500	ns

WCG Signals

- 4 MHz is the clock signal that is used to obtain the MFM or FM clock signal WCK.
- MFM is the control signal from the FDC, which controls the mode of the FDI. If the MFM signal is high then the FDI is set in MFM-mode and if the signal is low then the FDI is set in FM-mode.
- 0.5 MHz is a clock signal derived in the WCG. It is used in the RCD as a central clock frequency for the PLL.
- WCK is the Write Clock signal, that controls the write data rate of the FDC. In MFM-mode the frequency of this signal is 0.5 MHz and in FM-mode it is 0.25 MHz. The pulse width is 250 ns.
- $\overline{\text{WCK}}$ is the WCK signal inverted. This signal is used in the WDP Block to enable the shift register that shapes the delayed WDA signal's.

WDP Signals

- PS0 and PS1 are the precompensation or pre-shift signals from the FDC. They are connected through jumpers to the multiplexer in the WDP. The possible combinations are shown in table 6.
- 4 MHz is a system clock signal that is used as clock signal for the shift register in the WDP.
- WDA is the Write Data signal from the FDC. This signal is clocked by the WCK signal and latched into the WDP. The signal contains booth data and clock pulses according to the rules for the chosen mode of storage (MFM or FM).
- $\overline{WCK}$ is the clock signal that clocks the WDA signal into the WDP shift register. This signal is the inversion of the WCK signal, which clocks the WDA signal from the FDC to the WDP.
- WDA is the Pre-Compensated Write Data signal from the WDP to the Interface Logic and the FDD's.

Interface Logic

The Interfacing Logic can be divided into the Drive Select Decoder (DSD) and the Interfacing Circuitry (IC). The DSD simply selects drive 00 and 01 from the possible 4 FDD units that the FDC can control and ignores the other two.

The Interfacing Circuitry consists of three main blocks. Block 1 receives the signals from the FDD's and multiplexes these on to a common Bus. Block 2 places the FDC Control signals on the FDD Bus. This Block includes the In Use and Motor On signals from the Out Register. The Third Block is the Write Enable Block. This Block controls the Head enable signals to the FDD's.

The FDD Bus is equal for both Flexible Disk Drives. The signals and pinout for this Bus is shown below.

Table 7. FDD Bus signals and pinout.

Signal	Pin	Description
$\overline{GND}$	1–33	Ground
$\overline{DL}$	2	Door Lock
$\overline{IU}$	4	In Use
	6	Not Used
$\overline{IDX}$	8	Index
$\overline{DS}$	10	Drive Select; always active
	12	Not used
	14	Not Used
$\overline{MO}$	16	Motor On
$\overline{DIR}$	18	Direction
$\overline{STP}$	20	Step
$\overline{WDA}$	22	Write Data
$\overline{WE}$	24	Write Enable
$\overline{TR0}$	26	Track Zero
$\overline{WP}$	28	Write Protect
$\overline{RDD}$	30	Read Data
$\overline{SS}$	32	Side Select
$\overline{DS}$	34	Drive Status (Ready)

Drive Select Decoder, DSD

The Drive Select Decoder receives the US0 and US1 signals from the FDC. The circuitry in the DSD only accepts the coding on these lines for the drives 00 and 01. See table 8.

Table 8. Drive Select Decoding.

No.	US1	US0	DS0	$\overline{DS0}$	DS1	$\overline{DS1}$	Comments
1	0	0	0	1	1	0	Drive 00 select
2	0	1	1	0	0	1	Drive 01 select
3	1	0	1	0	1	0	No Drive select
4	1	1	1	0	1	0	No Drive Select

DSD Signals

- US0 and US1 are the Unit Select Signals from the FDC. They can encode up till four separate FDD's. See table 8 above.
- $\overline{DS0}$ and DS0 are the decoded Drive select 00 signals.
- $\overline{DS1}$ and DS1 are the decoded Drive Select 01 signals.

Block 1

Block 1 receives signals from the FDD's. The signals are multiplexed to one set of signal lines. The multiplexing is done by the US0, US1, DS0 and DS1 signals. Some enabling of signals is also carried out by the $\overline{RW}$/SEEK mode signal.

Block 1 Signals

- US0 with the aid of US1 selects signals from drive 00 or 01. US1 is active low. US0 low will chose signals from Drive 00. US0 high will chose signals from Drive 01.
- DS0 and DS1 are used to select the Track Zero Signal from either Drive.
- IDX is the selected Index Signal.
- RDD is the selected Read Data Signal.
- RDY is the selected Ready or Drive Status Signal.
- $\overline{WP}$ is the selected Write Protect Signal. It is gated with the mode signal $\overline{RW}$/SEEK from the FDC.
- $\overline{TR0}$ is the selected Track Zero Signal. It is enabled by the mode signal $\overline{RW}$/SEEK.
- $\overline{RW}$/SEEK is the mode signal from the FDC. When this line is high, the SEEK-mode is selected. When it is low, the Read/Write-mode is selected. In this application it is used to enable the Track Zero signal when the FDC is in SEEK-mode. It is also used to set the Write Protect line high when the FDC is in SEEK-mode. In SEEK-mode this line on the FDC is used to indicate Two side media (normally only implemented on 8" disks).

Block 2

Block 2 directs the control signals from the FDC to the chosen FDD. The signals that are handled are those that are concerned with placement of the disk head.

Block 2 also contains a buffer/inverter that takes care of the signals from the Out Register. These signals consists of Motor On, In Use and Door Lock.

Block 2 Signals

- $\overline{DS0}$ and $\overline{DS1}$ control which FDD is to receive the Output signals from the FDC. $\overline{DS0}$ sends the signals to FDD 00 and $\overline{DS1}$ sends them to FDD 01. The signals are active low.
- DIR is the Direction signal, which controls the direction of travel for the Read/Write head in the FDD in SEEK-mode. If DIR is high the Head will move towards the center of the diskette and vice versa. This signal is enabled by the $\overline{RW}$/SEEK signal.
- STP is the signal that moves the Read/Write head to a new track. In SEEK-mode the direction is determined by the DIR signal. STP is active high. This signal is enabled by the $\overline{RW}$/SEEK signal.
- SS is the Side Select Signal. That is, it selects the proper Read/Write Head on the selected FDD. When the signal is high, Head 1 is selected and when it is low Head 2 is selected.
- WDA is the Write Data signal from the WDP Unit. This signal is passed to the selected FDD.
- $\overline{RW}$/SEEK is the mode select signal. It is used to enable the signals DIR and STP in Block 2. It is active when high, which corresponds to SEEK-mode.
- M0 and M1 are the Motor On Signals from the Out Register. They are active high. For Instructions on setting these signals you are referred to the section on the Out Register.
- IU0 and IU1 are the In Use signals from the Out Register. They are active high. These signals are also used as Door Lock signals. For setting up you are referred to the section on the Out Register.

Block 3

Block 3 controls the Write Enable Signals (WE) to the FDD's. To enable the proper drive the WE signal is enabled by either the DS0 or the DS1 signal depending on which drive the FDC has activated. It is imperative that no write head is activated before the system power is on since there are lots of spurious noise on the control and data lines during Power Up. This noise could cause unwanted data to be placed on a disk. It is also important that the Write Enable signal can not be enabled during power down or during a power failure.

To ensure this the IC is furnished with circuits that depending on the state of the Power On Reset ($\overline{POR}$) and the Power Warning ($\overline{PW}$) signals, control the WE signal. The signals either disables or enables the WE signal. The timing diagram in fig. 8 illustrates how this is done.

As long as $\overline{POR}$ is active the WE signal is blocked off. This is done in two Ways. Primarily the $\overline{POR}$ signal being low disables the WE output drivers. Secondly the $\overline{POR}$ disables the WE signal through a Flip Flop arrangement, the WE disable circuits.

If the system is up and running and a power failure occurs this will cause the Power Warning ($\overline{PW}$) signal to become active. The Power supply is so dimensioned, that when the $\overline{PW}$ signal becomes active the system has at least 20 ms in which to complete a write current sector and then disable the write head. The maximum time for a complete sector write cycle, that is to write 512 bytes of data to a sector, is approximately 18 ms. The maximum time will occur if a $\overline{PW}$ signal arrives immediately after a sector write instruction has begun to execute. If a $\overline{PW}$ signal occurs during a write instruction, the circuitry is set up in such a way as to let the FDC complete the sector. When the WE signal goes low after the write cycle is finished the WE disable circuits will come into action and bar the FDC from enabling the write heads again. See fig 8.

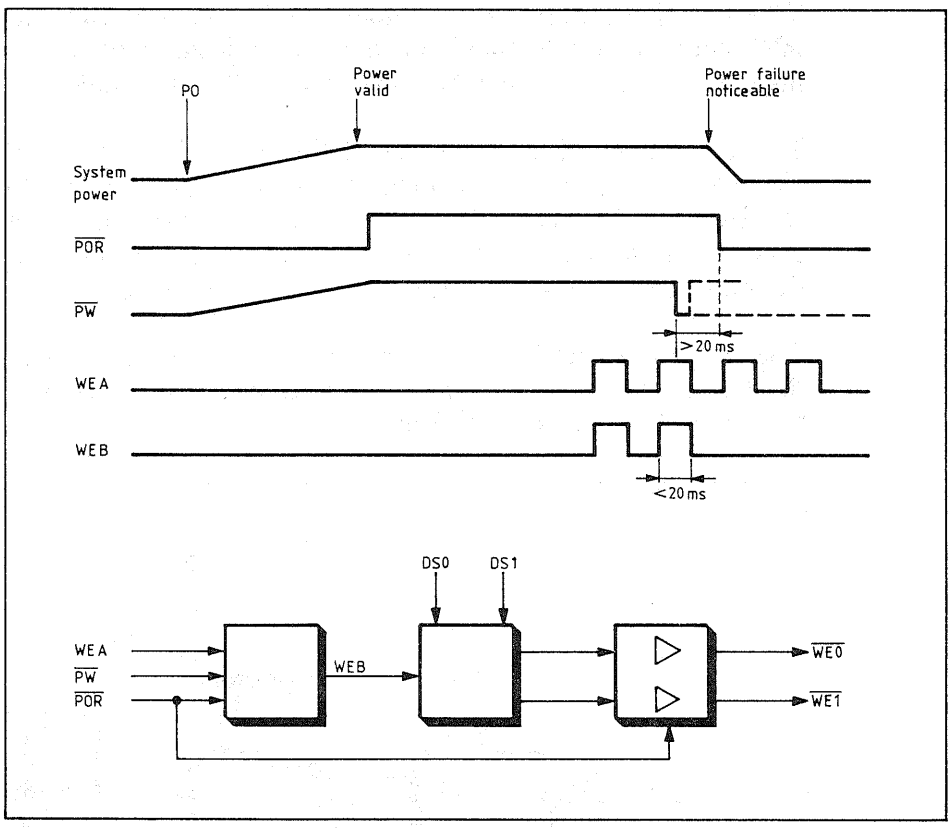


Fig. 8. Timing diagram and schematic circuit diagram for the Write Enable Protect circuits

Block 3 Signals

- DS0 and DS1 enables the WE signal to FDD 00 and FDD 01 respectively. The signals are active high.
- $\overline{POR}$ is the Power On Reset signal from the Reset circuits. This signal is active low. When the signal is active it has two functions. It disables the final inverting drivers for the WE signal by pulling their bases to ground.
  The $\overline{POR}$ signal also has the power to disable the WE signal with the second flip flop latch in the WE disable circuits.

- $\overline{PW}$ is the Power Warning signal. The signal is active low. It is activated $\geq$ 20 ms before +5V falls below its −5% tolerance, due to power off or power failure. The $\overline{PW}$ signal affects the first flip flop in the WE disable circuit. If the WE is active when the $\overline{PW}$ becomes active the first Flip flop changes state. This cannot alter the WE, however. Not until the WE has gone low itself. When this occurs the second flip flop will also go low and thereby the WE signal is disabled. If the WE is low when the $\overline{PW}$ occurres the WE is disabled immediately.
- WE is the Write Enable Signal from the FDC. This signal is active high. It enables the Read/Write Heads of the FDD's. The proper head to be enabled is singled out by the DS0 and DS1 signals.

Communications Interface

When used as a personal computer, the FD 4122 carries 256 K bytes of RW-Memory and furthermore it communicates with the Alfaskop 41 display unit on an asynchronous serial RS-232 data link.

The Communications Interface of the FD 4122 consists of the Serial Controller (SC) and the Modem, the Crystal Controlled Oscillator and the Two-wire Interface. As PC, the FD 4122 Communications Interface also uses additional circuitry in the form of a Programmable Timer and the necessary hardware to form the RS-232 interface.

The Programmable Timer makes it possible to create a real time clock for the PC. It also generates the clock frequency that is used to clock data from the SC to the RS-232 Bus.

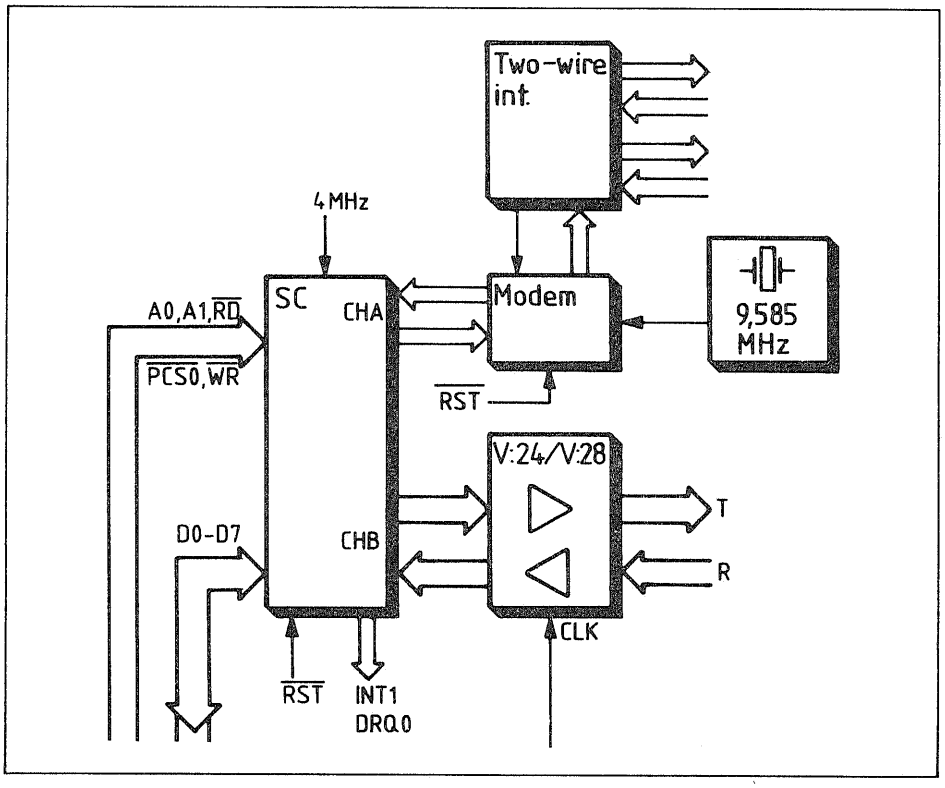


Fig. 9. Serial Controller and Communications Interface signals

The Serial Controller generates all the necessary signals for two serial data links. In this application they are used to control one asynchronous data link in form of an RS-232 interface and one synchronous data link in form of the Two-wire Interface.

The communication protocol for the Two-wire Interface is described in the Chapter on Communications in this folder.

The communication protocol for the RS-232 interface varies. It is dependent of the construction of the receiving unit. (We look upon the FD 4122 as the transmitting unit.)

Serial Controller, SC

The Serial Controller (SC) is an Intel 8274 Multi-Protocol Serial Controller. It can be set up in both synchronous and asynchronous mode. It contains two independent serial receiver/transmitter channels A and B. In this application Channel A is used for synchronous communication over a Two-wire Interface link. Channel B is used for asynchronous communication over a RS-232 serial link.

For a thorough description on the Intel 8274 Serial Controller you are referred to the manufacturer's data sheets.

In the chapter on Communications in this folder you will find a discussion on the Message Format for the Two-wire Communications Link. To set up channel A in the SC for this Format the registers should be set according to the rules for the chosen Message Format and the instructions in the section on Synchronous Operation in the manufacturer's data sheets.

In the abovementioned chapter you will also find a description on Asynchronous Communication. The principles discussed there are also applicable to the RS-232 interface principle. To set up Channel B for the RS-232 Communication the registers in the SC should be set according to these principles and the instructions given in the section on Asynchronous Operations in the manufacturer's data sheets.

A short description on how the SC functions in Synchronous respectively in Asynchronous Operation follows. In both cases the channels has to be initiated to the proper modes of operation. Also all values in the registers for respective serial protocol must be set before a data transfer is initiated.

The Synchronous Communication link normally uses DMA transfer for both Receive and Transmit Data communications. The Asynchronous link uses the MPU Interrupt line, provided between the SC and the MPU to effectuate data transfers between the SC and the MPU.

The DMA Request signals for both Transmit and Receive Data are connected through an OR-gate to let the SC use only one of the available DMA Channels in the MPU. This is possible because the Two-wire interface is only half duplex.

Table 9. Data in WR and RR Registers set for Synchronous Communication with DMA transfer of data.

Register	Transmit mode Data	Receive mode Data	Comments
WR0			Select Write register + command
WR1	0BH	08H	Interrupt setup
WR2	11H	11H	System configuration and interrrupt
WR3	C0H	09H	
WR4	20H	20H	Parity, sync mode, clock mode
WR5	EBH	60H	Transmitter setup
WR6			Unity address in Two-wire config.
WR7	7EH	7EH	Flag character
RR0	Transm. status	–	
RR1	–	Receiver status	
RR2	–	–	Interrupt vector status

Table 10. Data in WR and RR registers set for Asynchronous Communication with RS-232 protocol.

Register	Transmit mode Data	Receive mode Data	Comments
WR0	–	–	Select write register + command
WR1	14H	14H	Interupt setup
WR2	00H	00H	System configuration and interrupt
WR3	40H	41H	
WR4	47H	47H	Parity, sync mode, clock mode
WR5	AAH	A0H	Transmitter setup
WR6	–	–	
WR7	–	–	
RR0	Transm. status	Receive sign available	
RR1	All transm. flag	Receiver status	
RR2	–	–	

System Signals

There are a number of signals that are not brought out to the System Bus or any Interface Bus from the SC. They are either connected to System Ground or to System Power through a pull up resistor. These signals are presented in table 11.

Table 11. Signals to System Power or Ground.

Signal	Connection
$\overline{CDA}$	Vss
$\overline{INTA}$	Vcc via Pull up
$\overline{IPI}$	Vss
$\overline{IPO}$	Vss
$\overline{DTRA}$	Vss
$\overline{SYNDET}$	Vss

The Remaining System Signals are as follows:

- $\overline{RST}$ is the System Reset Signal. It is active low.
- CLK is the 4 MHz System Clock Signal.

- AD0-AD7 are the 8 LSB of the Address/Data Bus Signals. The SC can only access these signals when the Data Bus is active. To enable the SC to read or write data from or to the Bus, the SC has to be enabled by the $\overline{WR}$ or $\overline{RD}$ and the $\overline{PCS0}$ Signals.
- A0 is the LSB of the Address Bus. If this signal is low during a data or command transfer channel A will be selected. If A0 is high channel B will be selected.
- A1 is the second LSB of the Address Bus. If this signal is low during a data transfer it signals that the data should be treated as a Data Word by the SC. If the signal is high the data should be treated as a Control Word by the SC.
- $\overline{PCS0}$ is the Peripheral Chip Select Signal from the MPU. It is active low. This signal enables the SC to accept either the $\overline{RD}$ or the $\overline{WR}$ signals for data or control word transfer.
- $\overline{RD}$ is the System Read Signal. It is active low. Together with the $\overline{PCS0}$ signal, this signal places the SC in read mode, which enables the MPU or DMA to read data or status words from the Controller registers.
- $\overline{WR}$ is the System Write Signal. It is active low. Together with the $\overline{PCS0}$ signal, this signal allows the MPU or DMA to place data or control bytes in the registers of the SC Channel A or B. Which channel to be selected and whether the data transmitted shall be treated as Data or Control Word is controlled by the states of signals A0 and A1.
- INT1 is the System Interrupt Signal generated by the SC. It is active high. This signal indicates that the SC needs attention from the MPU.
- DRQ0 is the DMA Request Signal from the SC to the DMA Controller in the MPU. It is generated by either of the SC signals TxDRQA and RxDRQA. When Channel A of the SC is placed in either mode 1 or 2 this signal going high requests a DMA transfer to or from the SC, channel A.

V.24 (RS 232) Interface Signals

- TxD is the Transmit Data Signal. This line transmits serial data to the RS-232 Interface.
- $\overline{DTR}$ is the Data Terminal Ready Signal. It is active low. It is a general purpose output, which has to be set when the SC is set up for RS-232 transmission.
- $\overline{RTS}$ is a Request To Send Signal that is used to indicate that the SC is ready to transmit data. It is active low. The bit controlling this signal has to be set when the SC is used for RS-232 transmission.
- $\overline{TxC}$ is the Transmit Clock Signal. It is generated by the Progammable Timer and is used to provide the RS-232 interface with a signal which will clock data from the SC to the Interface. The TxD signal can be rate programmed in the SC to 1, 1/16, 1/32 or 1/64 of the $\overline{TxC}$ Signal. The Clock signal is also sent to the receiving Unit to provide this with a Receive Clock Signal.
- RxD is the Receive Data Signal. It receives serial data from the Interface. The data is clocked into the SC by the Receive Clock Signal.
- $\overline{CD}$ is the Carrier Detect Signal. It is active low. It signals to the SC that data is being sent from the RS-232 Interface.

- $\overline{\text{CTS}}$ is the Clear To Send Signal. It is active low. It signals the SC that the receiving Unit is ready to accept data from the SC.
- $\overline{\text{RxC}}$ is the Receiver Clock Signal. It is active low. It is used to clock in data on the RxD line. The clock signal can be submitted by the connected Unit or it can be supplied from the Programmable Timer.

Two-wire Interface Signals

These signals are identical to the signals described in the Chapter on Communications in this folder. The signals are described in the section on ADLC Signals in that chapter.

- $\overline{\text{RTS}}$ is the Request To Send Signal from the SC to the Modem. It is active low. This signal indicates that the SC is ready to send data to the Modem.
- $\overline{\text{CTS}}$ is the Clear To Send Signal from the Modem to the SC. It is active low. The signal indicates that the Modem is ready to start sending data via the Two-Wire link.
- TxD is the Transmit Data Signal, which sends the serialised data to the Modem. The data is clocked out by the Bit Clk signal.
- RxD is the Receive Data Signal, which receives the demodulated serial signal from the Modem. The data is clocked into the SC by the Bit Clk Signal.
- Bit Clk is the Clock signal derived from the 9.585 MHz clock signal to the Modem. The Bit Clk has a frequency of 300 KHz. It is connected to both the Transmit Clock Input ($\overline{\text{TxC}}$) and the Receive Clock Input ($\overline{\text{RxC}}$) on the SC.

Programmable Timer

The Programmable Timer is the Intel 8254 Programmable Interval Timer. This Timer contains three independent timers. Each timeris controlled by a Control Word stored in the Control Word Register. The Control Word determines the timer mode and which timer the Control Word is associated with.

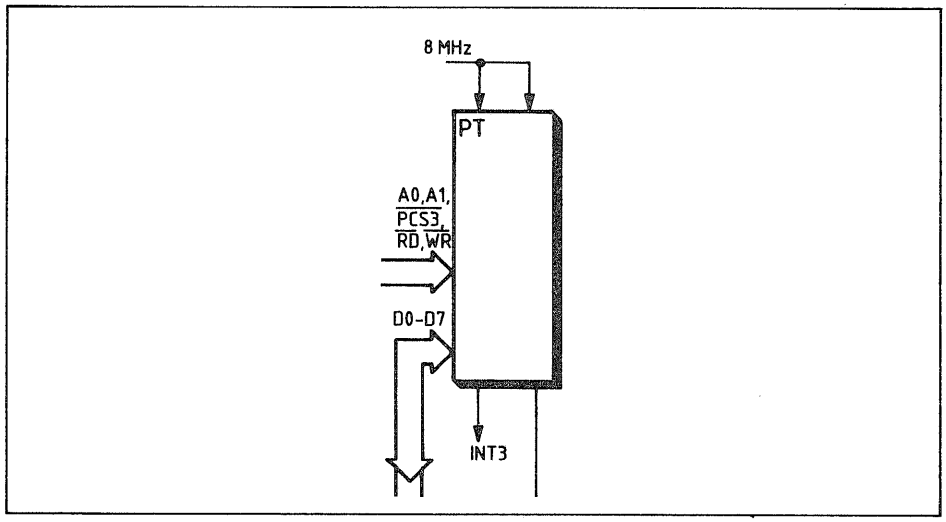


Fig. 10. Programmable Timer signals

Each timer can be set to count down from an initial count stored in a counter associable with the timer.

For a detailed description on the Intel 8254 you are referred to the manufacturer's data sheets.

Table 12. Use of address lines A1 and A0.

Address Line A1	A0	Selection	Control signal
1	1	Control Word	$\overline{WR}$
0	0	Counter 0	$\overline{WR}$, $\overline{RD}$
0	1	Counter 1	$\overline{WR}$, $\overline{RD}$
1	0	Counter 2	$\overline{WR}$, $\overline{RD}$

To initiate a timer the proper Control Word must be written in the Control Word Register. Thereafter the counter associate with the timer has to be set to the initial count. In this application timers 0 and 1 are ordinarily used as a real time clock. The clock will interrupt the MPU at set intervals and cause the MPU to enter a Real Time Interrupt Routine which updates the Real Time Clock Data.

To set the Timer up for this the Timer 0 and Timer 1 has to be set to mode 2. To set the timers to this mode the respective Control Words has to be set with the following hexadecimal data:

Timer 0 34H
Timer 1 74H

This setting indicates that both the LSB and the MSB of the Counters has to be set, in that order. If only one of these registers, for each counter need to be set, the Control Words should be changed accordingly.

The associated counters has to be set with the following data to let the PT interrupt the MPU at appropriate intervals.

Table 13. Example for settings for Real Time Clock Interrupt

Timer Counter	MSB	LSB	Interrupt sequence (sec)
0	00H	04H	
1	9CH	40H	0.02

To set Timer 2 up as a baud rate controller it has to be placed in mode 3. This means that the Control Word has to be set with the following hexadecimal data:

Timer 2 B6H

This setting indicates that both the LSB and the MSB of the Counter has to be set, in that order. If only one of these registers need to be set, the Control Word should be changed accordingly.

The counter associated with this timer has to be set with different data depending on which baudrate the RS-232 Interface shall be set up for.

Note that the SC divides the incoming baudrate signal TxC (Transmit Clock Signal) by 16 before using it as a clock for the TxD (Transmit Data Signal). This means that the actual clock signal from the PT should be set to a value 16 times greater than the desired baudrate.

Table 14. Setting of Timer 3 for different baudrates.

Baudrate setting	Counter LSB	MSB
300	83H	06H
600	41H	03H
1200	A1H	01H
2400	D0H	00H
4800	68H	00H
9600	34H	00H
19200	1AH	00H

System Signals

- AD0-AD7 are the 8 LSB of the System Address/Data Bus. Since the registers in the PT can both be written to and read from, the data bus can be activated by the $\overline{RD}$ or the $\overline{WR}$ signals.
  The data needed to set up registers and counters is passed into the PT this way.
- $\overline{PCS3}$ is the peripheral Chip Select Signal that enables the Programmable Timer. It is active low. When this signal is active the $\overline{RD}$ and $\overline{WR}$ signals are acknowledged by the PT.
- $\overline{RD}$ is the System Read Signal. It is active low. When this signal is active, together with the $\overline{PCS3}$, the MPU can access the contents of the Control Word Register for one of the Timers. Which register the MPU wants to read is determined by the states of address lines A0 and A1.
- $\overline{WR}$ is the System Write Signal. It is active low. When this signal is active, together with the $\overline{PCS3}$, the MPU can write data into either a Control Word Register or initial counter for one of the timers. Which timer and whether it is to write a Control Word or an Initial Count is determined by the settings of address lines A0 and A1.
- A0 and A1 are the two LSB of the System Address Bus. These signals are used to indicate whether the current data is to be written into the Control Word Register or into one of the initial counters. See Table 12. above.
- CLK0 and CLK2 are both connected to the 8 MHz System Clock. This means that both Timer 0 and Timer 2 has a clock frequency of 8 MHz to be divided into the preset signal. Each Timer can be set up in any one of 6 available modes.

  The Output of Timer 0 is connected to the CLK Input of Timer 1. The Gates of all three timers are connected to System Power through a pull up resistor.

Output Signals

- INT3 is a System Interrupt Signal to the Interrupt Controller in the MPU. It is the Output signal from Timer 1 in the PT. It is possible to set up the Registers and counters in Timer 0 and 1 to let this signal

generate an interrupt every 1 s, 0.1 s or 0.01 s or any other timing between 2 clock pulses and 256*256 clock pulses. The clock frequency in Timer 0 is 8 MHz.

- Out2 is the Output Signal from Timer 2. It is active high. The Timer should be set up to generate a transmitter Clock Signal, which will be used to clock data from the SC to the RS-232 Bus.

V.24 (RS-232) Interface

The RS-232 Interface contains standard RS-232 Receiver and Transmitter circuits. The Signals to or from the SC are simply buffered or put through drivers in the Interface. The Standard RS-232 Signal names and the pinout for these can be found in Table 15.

Table 15. V.24 (RS-232) Interface signal names and pinout.

Signal name	Pin no.	Comments
Data	2	Data receive line
Data	3	Data transmit line
RTS	4	Request To Send receive line
RFS	5	Ready for sending
DSR	6	Data set ready
GND	7	System Ground
DCD	8	Data carrier detected
TSET	15	Transmitter signal element timing (Transmitter clock)
RSET	17	Receiver signal element timing (Receiver clock)
DTR	20	Data Transmit receive line
TSET	24	(Transmit clock out)

SC Signals

The signals between the RS-232 Interface Circuits and the SC are identical to the ones described in the section on the SC Signals with the same names. Therefore only the names, of the identical signals will be given in this section. Also the standard RS-232 interface signals will not be explained outside what is said in Table 15.

- OUT 2 is the Output Signal from the third timer in the PT circuit. It is used as a clock signal for the RS-232 Interface. It is connected to the TxCB signal to the SC Block and to the TSET and RSET signals on the output side of the RS- 232 Interface.
- TxD
- DTR
- RTS
- TxC
- RxD
- CD
- CTS
- RxC

Modulator/Demodulator

The Modulator/Demodulator (Modem) is a custom made chip. It converts the serial data stream from the SC into a frequency shifted signal which is fed to the Two-Wire interface. It also converts the frequency shifted signal from the Two-Wire Interface to a Received Data stream and a Bit Clock signal and sends these to the SC. The principles of the Two-Wire Interface are shown in fig. 5. in the chapter on Communication in this Folder.

The signal is modulated according to the Two-Wire Receive Protocol. See fig. 11.

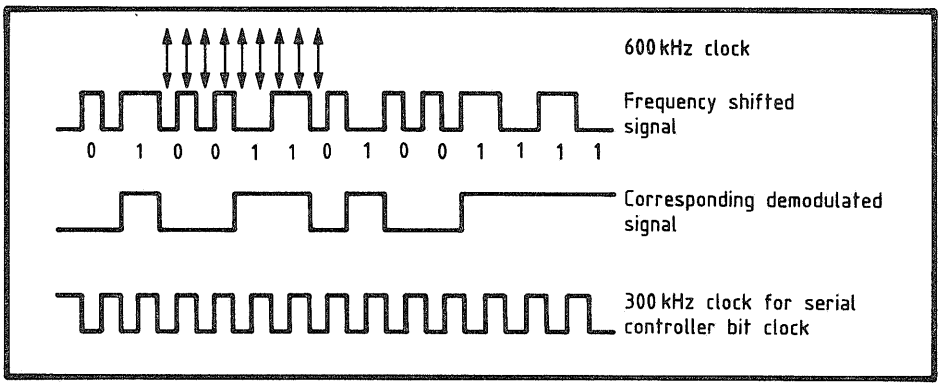


Fig. 11. Modulation, demodulation and Bit Clock.

The Modem receives the clocking frequency for the frequency shifted signal from a crystal controlled oscillator circuit. The clocking frequency is set to 9.585 MHz. The clock frequency is divided by 15, 16 or 17 in the Modem to generate the different frequencies that are needed in the Modem. See Fig. 11.

Signals

- CLK is the 9.585 MHz signal from the crystal controlled Oscillator. This clock signal is used in the Modem to generate the frequency shifted output data signal to the Two-Wire interface.
- $\overline{\text{RST}}$ is the System Reset Signal from the MPU. This signal is active low.

SC Signals

The signals between the Modem and the SC are identical to the ones with identical names described in the section on the SC Signals. Therefore only the names, of the identical signals will be given in this section.

- $\overline{\text{RTS}}$
- $\overline{\text{CTS}}$
- TxD
- RxD
- Bit Clk

Two-wire Interface Signals

- FS and $\overline{FS}$ are the Frequency Shifted Data Signals from the modem to the Two-Wire Interface. These signals are enabled inside the Modem block by the Request To Send (RTS) Signal.
- MRxD is the Frequency Shifted Signal from the Two-Wire Interface to the Modem. The Signal is demodulated in the Modem and sent to the SC.

Two-wire Interface

The Two-wire and Coaxial Interface consists mainly of an line transformer and and some comparators and amplifying circuits.

The Interface has two input/output connectors for the Two-wire link and another two connectors for the Coaxial version of the link.

The output frequency shifted signal from the Modem is fed directly to the Two-wire link through the line transformer. The input signal over the Two-wire link is detected, amplified and thereafter fed to the Modem.

The communications protocol for the Two-wire Interface is presented in the chapter on Communication in this folder.

Signals

- MRxD is the Receive Data Signal to the Modem. It carries the modulated signal, which is demodulated in the Modem.
- FS and $\overline{FS}$ are the lines that carry the frequency shifted signal from the Modem to the Interface. The construction of the modulated signal can be seen in Fig. 11.

Out Register

The Out Register (OR) contains an 8-bit register and some additional circuits to drive 4 Light Emitting Diodes. The Register can only be accessed in write mode by the MPU. It is enabled by the Peripheral Chip Select line $\overline{PCS4}$. It is only one byte wide and therefore it doesn't need any extra address decoding. There are 5 LED's in the block. One is dedicated as a Power Indicator and is turned on when the Power Supply is switched on. Of the four remaining LED's, one is driven by the LSB of the Data Bus (AD0). It can be used as a Ready Signal. To light up this LED the AD0 bit has to be set High. The three remaining LED's are driven by the lines AD1 and AD2 from the Address/Data Bus. Table 16. gives the settings to turn on the LED's 1-3.

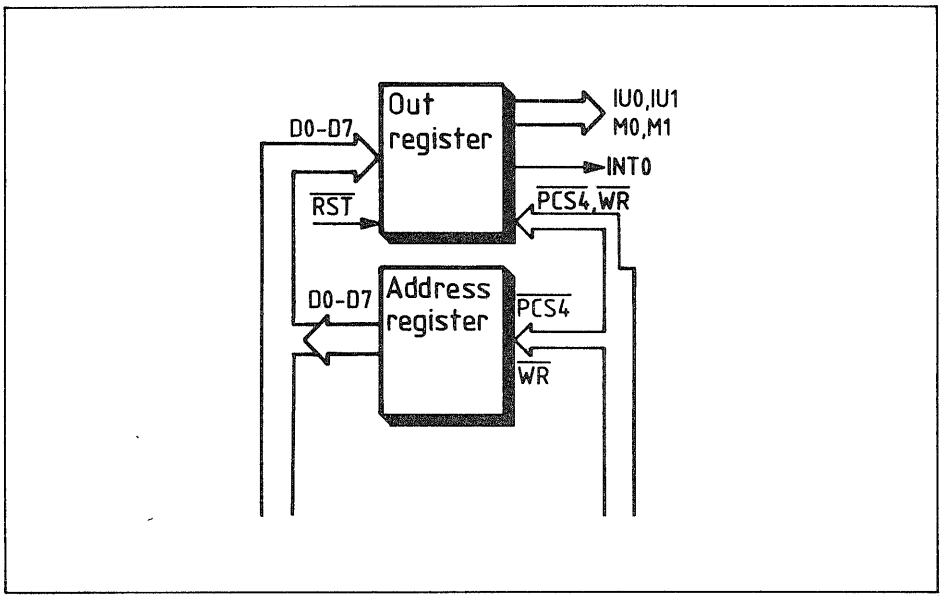


Fig. 12. Out Register and Address Register signals

Table 16. LED lighting settings.

Data bits AD1	AD2	Lighted
1	1	LED 1
0	1	LED 2
1	0	LED 3

Signals

- $\overline{WR}$ is the Write Enable Signal from the CPU. It is active low. Together with the $\overline{PCS4}$ signal from the MPU this signal clocks the 8 LSB of the Address/Data Bus into the Out Register.
- $\overline{PCS4}$ is the Peripheral Chip Select Signal from the MPU. It is active low. Together with the $\overline{WR}$ signal from the MPU it clocks the 8 LSB of the Address/Data Bus into the OR.
- $\overline{RST}$ is the System Reset Signal from the MPU. It is active low. On system reset this signal will reset the register so that all outputs are set to zero.
- AD0-AD7 are the 8 LSB of the combined Address/Data Bus. The $\overline{WR}$ and the $\overline{PCS4}$ clock the Data signals into the register. The Address signals are ignored. When the Data has been clocked into the register the outputs from the register remain in their set states until the Out Register is accessed the next time.
- MO 0 and MO 1 are the Motor On Signals to the FDD's. Both signals are active high. They are set up in the Out Register, to give you the possibility to chose which Drive should be running. This arrangement also makes it possible for you to set up a timer to cause a time out for the drives and subsequently stop the drives. Also both drive motors can be enabled at the same time.

- IU0 and IU1 are the In Use Signals to the FDD's. These signals are active high. They are set up in the same way as the MO 0 and MO 1 signals.
- INT0 is an Interrupt Signal. It is connected to the Interrupt Controller Input on the MPU. The signal is active high. To use a register to set this interrupt provides you with a possibility to test the interrupt controller. It can also be used as a fictive interrupt in test routines.

Address Register

The Address Register (AR) consists mainly of a DIL-switch and an octal buffer. It is primarily used to let the FD 4122 Data Terminal check the identity of itself. This has to be done when the Terminal receives data over the Two-wire communications Link. The data sent over the link carries with it the address of the receiving Terminal and this has to be checked before the Terminal can start to receive the data. The DIL-switch contains 8 switches which makes it possible to set up to 256 addresses.

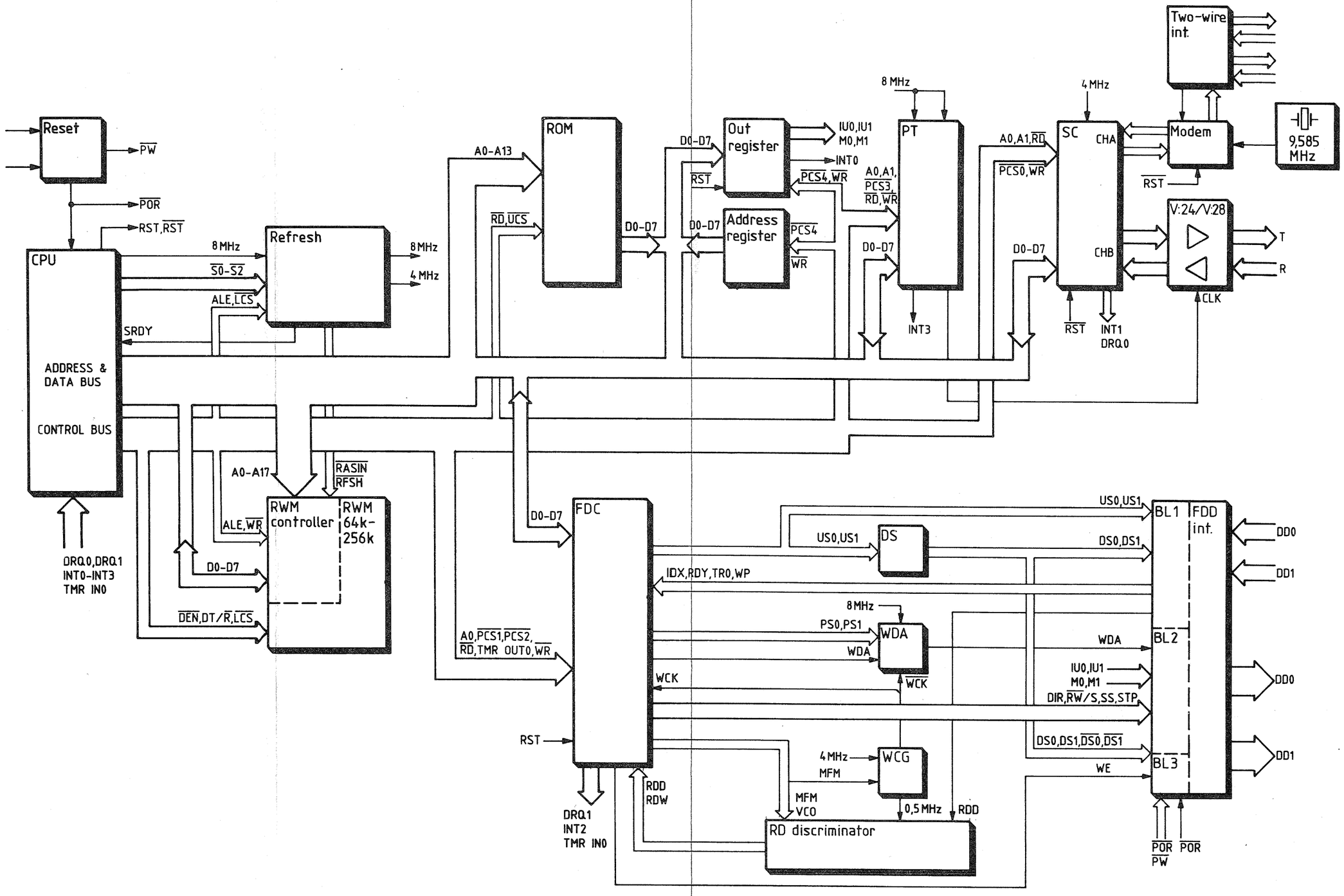
The AR can also be used for testing purposes. It is possible to construct a initial start up routine, which will read the state of the AR and depending on the setting, the routine will jump to a certain test routine. This makes it possible to test out different parts of the FD 4122 one by one. It also makes it possible to use the LED's in the Out Register and to let them have different meanings in different test routines.

The AR can only be read by the MPU. Therefore it can use the same Chip Select Signal as the Out Register.

The DIL-switches are all connected to ground. This means that a closed switch (ON), is signaled as a low-state line to the Buffer. The Buffer inverts the signal. Therefore a DIL-switch in the ON-position means that the corresponding Data Bus signal will be set high. Inversly an DIL-switch in the OFF-position means that the corresponding Data Bus signal will be set low.

Signals

- $\overline{RD}$ is the MPU Read signal. It is active low. In the AR it is used together with the $\overline{PCS4}$ signal to enable the octal buffer to place the 8 data bits on the Data Bus.
- $\overline{PCS4}$ is the Peripheral Chip Select Signal from the MPU. It is active low. Together with the $\overline{RD}$ signal it enables the octal buffer in the AR and places the 8 data bits on the Data Bus.
- AD0–AD7 are the 8 LSB of the Address/Data Bus. The 8 data bits in the AR are put on to this bus as if they where read from an ordinary memory location with the exception that the AR is addressed with a Chip Select Signal. The data placed on the Address/Data Bus are then read by the MPU. When the enabling signals are released by the MPU the buffer in the AR disconnects the data lines in the AR from the Address/Data Bus.



Appendix 1
Block diagram for FD 4122

Appendix 2

iAPX188 Instruction Set

intel iAPX 188 ADVANCE INFORMATION

GENERAL PURPOSE	
MOV	Move byte or word
PUSH	Push word onto stack
POP	Pop word off stack
PUSHA	Push all registers on stack
POPA	Pop all registers from stack
XCHG	Exchange byte or word
XLAT	Translate byte
INPUT/OUTPUT	
IN	Input byte or word
OUT	Output byte or word
ADDRESS OBJECT	
LEA	Load effective address
LDS	Load pointer using DS
LES	Load pointer using ES
FLAG TRANSFER	
LAHF	Load AH register from flags
SAHF	Store AH register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack

ADDITION	
ADD	Add byte or word
ADC	Add byte or word with carry
INC	Increment byte or word by 1
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition
SUBTRACTION	
SUB	Subtract byte or word
SBB	Subtract byte or word with borrow
DEC	Decrement byte or word by 1
NEG	Negate byte or word
CMP	Compare byte or word
AAS	ASCII adjust for subtraction
DAS	Decimal adjust for subtraction
MULTIPLICATION	
MUL	Multiply byte or word unsigned
IMUL	Integer multiply byte or word
AAM	ASCII adjust for multiply
DIVISION	
DIV	Divide byte or word unsigned
IDIV	Integer divide byte or word
AAD	ASCII adjust for division
CBW	Convert byte to word
CWD	Convert word to doubleword

MOVS	Move byte or word string
INS	Input bytes or word string
OUTS	Output bytes or word string
CMPS	Compare byte or word string
SCAS	Scan byte or word string
LODS	Load byte or word string
STOS	Store byte or word string
REP	Repeat
REPE/REPZ	Repeat while equal/zero
REPNE REPNZ	Repeat while not equal/not zero

LOGICALS	
NOT	"Not" byte or word
AND	"And" byte or word
OR	"Inclusive or" byte or word
XOR	"Exclusive or" byte or word
TEST	"Test" byte or word
SHIFTS	
SHL SAL	Shift logical arithmetic left byte or word
SHR	Shift logical right byte or word
SAR	Shift arithmetic right byte or word
ROTATES	
ROL	Rotate left byte or word
ROR	Rotate right byte or word
RCL	Rotate through carry left byte or word
RCR	Rotate through carry right byte or word

FLAG OPERATIONS	
STC	Set carry flag
CLC	Clear carry flag
CMC	Complement carry flag
STD	Set direction flag
CLD	Clear direction flag
STI	Set interrupt enable flag
CLI	Clear interrupt enable flag
EXTERNAL SYNCHRONIZATION	
HLT	Halt until interrupt or reset
WAIT	Wait for $\overline{TEST}$ pin active
ESC	Escape to extension processor
LOCK	Lock bus during next instruction
NO OPERATION	
NOP	No operation
HIGH LEVEL INSTRUCTIONS	
ENTER	Format stack for procedure entry
LEAVE	Restore stack for procedure exit
BOUND	Detects values outside prescribed range

CONDITIONAL TRANSFERS	
JA/JNBE	Jump if above/not below nor equal
JAE/JNB	Jump if above or equal/not below
JB/JNAE	Jump if below/not above nor equal
JBE/JNA	Jump if below or equal/not above
JC	Jump if carry
JE/JZ	Jump if equal/zero
JG/JNLE	Jump if greater/not less nor equal
JGE/JNL	Jump if greater or equal/not less
JL/JNGE	Jump if less/not greater nor equal
JLE/JNG	Jump if less or equal/not greater
JNC	Jump if not carry
JNE/JNZ	Jump if not equal/not zero
JNO	Jump if not overflow
JNP/JPO	Jump if not parity/parity odd
JNS	Jump if not sign
JO	Jump if overflow
JP/JPE	Jump if parity/parity even
JS	Jump if sign

UNCONDITIONAL TRANSFERS	
CALL	Call procedure
RET	Return from procedure
JMP	Jump
ITERATION CONTROLS	
LOOP	Loop
LOOPE/LOOPZ	Loop if equal/zero
LOOPNE/LOOPNZ	Loop if not equal/not zero
JCXZ	Jump if register CX = 0
INTERRUPTS	
INT	Interrupt
INTO	Interrupt if overflow
IRET	Interrupt return

Addressing Modes

The 80188 provides eight categories of addressing modes to specify operands. Two addressing modes are provided for instructions that operate on register or immediate operands:

- *Register Operand Mode:* The operand is located in one of the 8- or 16-bit general registers.
- *Immediate Operand Mode:* The operand is included in the instruction.

Six modes are provided to specify the location of an operand in a memory segment. A memory operand address consists of two 16-bit components: a segment base and an offset. The segment base is supplied by a 16-bit segment register either implicity chosen by the addressing mode or explicitly chosen by a segment override prefix. The offset, also called the effective address, is calculated by summing any combination of the following three address elements:

- the *displacement* (an 8- or 16-bit immediate value contained in the instruction);
- the *base* (contents of either the BX or BP base registers); and
- the *index* (contents of either the SI or DI index registers).

Any carry out from the 16-bit addition is ignored. Eight-bit displacements are sign extended to 16-bit values.

Combinations of these three address elements define the six memory addressing modes, described below.

- *Direct Mode:* The operand's offset is contained in the instruction as an 8- or 16-bit displacement element.
- *Register Indirect Mode:* The operand's offset is in one of the registers SI, DI, BX, or BP.
- *Based Mode:* The operand's offset is the sum of an 8- or 16-bit displacement and the contents of a base register (BX or BP).
- *Indexed Mode:* The operand's offset is the sum of an 8- or 16-bit displacement and the contents of an index register (SI or DI).
- *Based Indexed Mode:* The operand's offset is the sum of the contents of a base register and an index register.
- *Based Indexed Mode with Displacement:* The operand's offset is the sum of a base register's contents, an index register's contents, and an 8- or 16-bit displacement.

Data Types

The 80188 directly supports the following data types:

- *Integer:* A signed binary numeric value contained in an 8-bit byte or a 16-bit word. All operations assume a 2's complement representation. Signed 32- and 64-bit integers are supported using the iAPX 188/20 Numeric Data Processor.
- *Ordinal:* An unsigned binary numeric value contained in an 8-bit byte or a 16-bit word.
- *Pointer:* A 16- or 32-bit quantity, composed of a 16-bit offset component or a 16-bit segment base component in addition to a 16-bit offset component.
- *String:* A contiguous sequence of bytes or words. A string may contain from 1 to 64K bytes.
- *ASCII:* A byte representation of alphanumeric and control characters using the ASCII standard of character representation.
- *BCD:* A byte (unpacked) representation of the decimal digits 0–9.
- *Packed BCD:* A byte (packed) representation of two decimal digits (0–9). One digit is stored in each nibble (4-bits) of the byte.
- *Floating Point:* A signed 32-, 64-, or 80-bit real number representation. (Floating point operands are supported using the iAPX 188/20 Numeric Data Processor configuration.)

In general, individual data elements must fit within defined segment limits. Figure 7 graphically represents the data types supported by the iAPX 188.

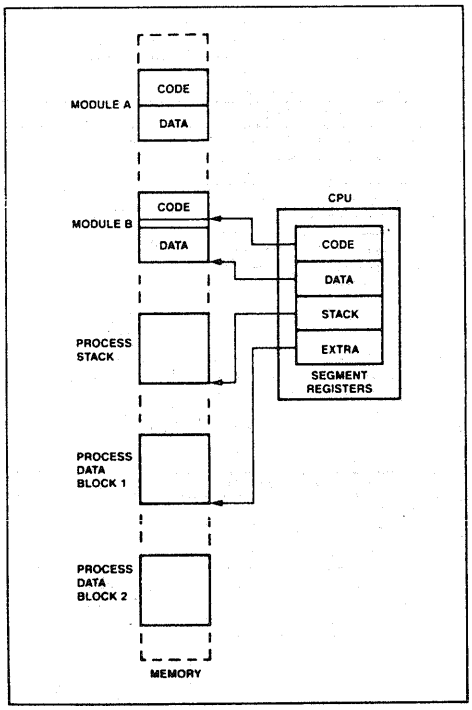


Figure 6. Segmented Memory Helps Structure Software

Appendix 3

FDC Instruction Set

μPD765A

INSTRUCTION SET ① ②

PHASE	R/W	DATA BUS D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	REMARKS
READ DATA			
Command	W	MT MF SK 0 0 1 1 0	Command Codes
	W	X X X X X HD US1 US0	
	W	C	Sector ID information prior to Command execution. The 4 bytes are commanded against header on Floppy Disk
	W	H	
	W	R	
	W	N	
	W	EOT	
	W	GPL	
	W	DTL	
Execution			Data-transfer between the FDD and main-system
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information after Command execution
	R	H	
	R	R	
	R	N	
READ DELETED DATA			
Command	W	MT MF SK 0 1 1 0 0	Command Codes
	W	X X X X X HD US1 US0	
	W	C	Sector ID information prior to Command execution. The 4 bytes are commanded against header on Floppy Disk.
	W	H	
	W	R	
	W	N	
	W	EOT	
	W	GPL	
	W	DTL	
Execution			Data-transfer between the FDD and main-system
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information after Command execution
	R	H	
	R	R	
	R	N	
WRITE DATA			
Command	W	MT MF 0 0 0 1 0 1	Command Codes
	W	X X X X X HD US1 US0	
	W	C	Sector ID information prior to Command execution. The 4 bytes are commanded against header on Floppy Disk.
	W	H	
	W	R	
	W	N	
	W	EOT	
	W	GPL	
	W	DTL	
Execution			Data-transfer between the main-system and FDD
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information after Command execution
	R	H	
	R	R	
	R	N	
WRITE DELETED DATA			
Command	W	MT MF 0 0 1 0 0 1	Command Codes
	W	X X X X X HD US1 US0	
	W	C	Sector ID information prior to Command execution. The 4 bytes are commanded against header on Floppy Disk.
	W	H	
	W	R	
	W	N	
	W	EOT	
	W	GPL	
	W	DTL	
Execution			Data-transfer between the FDD and main-system
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information after Command execution
	R	H	
	R	R	
	R	N	

PHASE	R/W	DATA BUS D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	REMARKS
READ A TRACK			
Command	W	0 MF SK 0 0 0 1 0	Command Codes
	W	X X X X X HD US1 US0	
	W	C	Sector ID information prior to Command execution
	W	H	
	W	R	
	W	N	
	W	EOT	
	W	GPL	
	W	DTL	
Execution			Data-transfer between the FDD and main-system. FDC reads all data fields from index hole to EOT.
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information after Command execution
	R	H	
	R	R	
	R	N	
READ ID			
Command	W	0 MF 0 0 1 0 1 0	Commands
	W	X X X X X HD US1 US0	
Execution			The first correct ID information on the Cylinder is stored in Data Register
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information read during Execution Phase from Floppy Disk.
	R	H	
	R	R	
	R	N	
FORMAT A TRACK			
Command	W	0 MF 0 0 1 1 0 1	Command Codes
	W	X X X X X HD US1 US0	
	W	N	Bytes/Sector
	W	SC	Sectors/Track
	W	GPL	Gap 3
	W	D	Filler Byte
Execution			FDC formats an entire track
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	In this case, the ID information has no meaning
	R	H	
	R	R	
	R	N	
SCAN EQUAL			
Command	W	MT MF SK 1 0 0 0 1	Command Codes
	W	X X X X X HD US1 US0	
	W	C	Sector ID information prior to Command execution
	W	H	
	W	R	
	W	N	
	W	EOT	
	W	GPL	
	W	STP	
Execution			Data-compared between the FDD and main-system
Result	R	ST 0	Status information after Command execution
	R	ST 1	
	R	ST 2	
	R	C	Sector ID information after Command execution
	R	H	
	R	R	
	R	N	

Note: ① Symbols used in this table are described at the end of this section.

② A_0 should equal binary 1 for all operations.

③ X = Don't care, usually made to equal binary 0.

μPD765A

INSTRUCTION SET (CONT.)

PHASE	R/W	DATA BUS D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	REMARKS
SCAN LOW OR EQUAL										
Command	W	MT	MF	SK	1	1	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	C								Sector ID information prior Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data-compared between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								
SCAN HIGH OR EQUAL										
Command	W	MT	MF	SK	1	1	1	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	C								Sector ID information prior Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data-compared between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								

PHASE	R/W	DATA BUS D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	REMARKS
RECALIBRATE										
Command	W	0	0	0	0	0	1	1	1	Command Codes
	W	X	X	X	X	X	0	US1	US0	
Execution										Head retracted to Track 0
SENSE INTERRUPT STATUS										
Command	W	0	0	0	0	1	0	0	0	Command Codes
Result	R	STO								Status information at the end of seek-operation about the FDC
	R	PCN								
SPECIFY										
Command	W	0	0	0	0	0	0	1	1	Command Codes.
	W	SRT						HUT		
	W	HLT						ND		
SENSE DRIVE STATUS										
Command	W	0	0	0	0	0	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
Result	R	ST 3								Status information about FDD
SEEK										
Command	W	0	0	0	0	1	1	1	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W	NCN								
Execution										Head is positioned over proper Cylinder on Diskette
INVALID										
Command	W	Invalid Codes								Invalid Command Codes (NoOp – FDC goes into Standby State)
Result	R	ST 0								ST 0 = 80 (16)

COMMAND SYMBOL DESCRIPTION

SYMBOL	NAME	DESCRIPTION
A_0	Address Line 0	A_0 controls selection of Main Status Register (A_0 = 0) or Data Register (A_0 = 1)
C	Cylinder Number	C stands for the current/selected Cylinder (track) number 0 through 76 of the medium.
D	Data	D stands for the data pattern which is going to be written into a Sector.
D_7-D_0	Data Bus	8-bit Data Bus, where D_7 stands for a most significant bit, and D_0 stands for a least significant bit.
DTL	Data Length	When N is defined as 00, DTL stands for the data length which users are going to read out or write into the Sector.
EOT	End of Track	EOT stands for the final Sector number on a Cylinder. During Read or Write operation FDC will stop date transfer after a sector # equal to EOT.
GPL	Gap Length	GPL stands for the length of Gap 3. During Read/Write commands this value determines the number of bytes that VCOs will stay low after two CRC bytes. During Format command it determines the size of Gap 3.
H	Head Address	H stands for head number 0 or 1, as specified in ID field.
HD	Head	HD stands for a selected head number 0 or 1 and controls the polarity of pin 27. (H = HD in all command words.)
HLT	Head Load Time	HLT stands for the head load time in the FDD (2 to 254 ms in 2 ms increments).
HUT	Head Unload Time	HUT stands for the head unload time after a read or write operation has occurred (16 to 240 ms in 16 ms increments).
MF	FM or MFM Mode	If MF is low, FM mode is selected, and if it is high, MFM mode is selected.
MT	Multi-Track	If MT is high, a multi-track operation is to be performed. If MT = 1 after finishing Read/Write operation on side 0 FDC will automatically start searching for sector 1 on side 1.

Display Unit

Contents

Figures

GENERAL

This functional description of the display unit (DU 4110) is to be used for the DTC and DTC-A versions, i.e. both for DU 4110-001 and DU 4110-002. The section "DTC Principles" describes DU 4110-001 functions. "DTC-A Principles" covers DU 4110-002 functions.

The main tasks of the display unit (DU 4110) are

- Presentation of visual information on the CRU (Cathode ray tube unit). This information may be text entered via the keyboard, messages from the computer or the operating system, forms from a library on a flexible disk etc.
- Communication with a host computer and other system units like keyboard, selector pen, printers, flexible disk units and communication processor.
- Editing and processing of data, e.g. form handling operations defined by Alfaform, code conversion etc.

The main parts of the display unit are

- Cathode ray tube unit (CRU)
- Microcomputer, consisting of the display terminal controller (DTC), the two-wire interface adapter (TIA) and other optional printed circuit boards.
- Power supply
- Basic mechanics

MECHANICS

As can be seen in Fig.1 the tiltable and swivelable upper part of the display unit contains four basic modules

- Cathode ray tube unit (CRU)
- Display terminal power supply (DPS)
- Housing (and EMI shield) for the DTC, TIA (and SPA if a selector pen is used).

- Board expansion module (BXM) with four electrically identical places for logic boards. The BXM is an option, only used for either

1. Single display unit configuration, synchronous communication (SCA used).
2. More than 32 kbytes memory needed (MRW or MRO used) or
3. Asynchronous communication needed (e.g. to Printer Unit 4154 or in TTY configuration) (ACA used).

The base ("square part" of the foot) of the display unit contains a fan and connectors for keyboard and two-wire cable to communication processor or flexible disk unit.

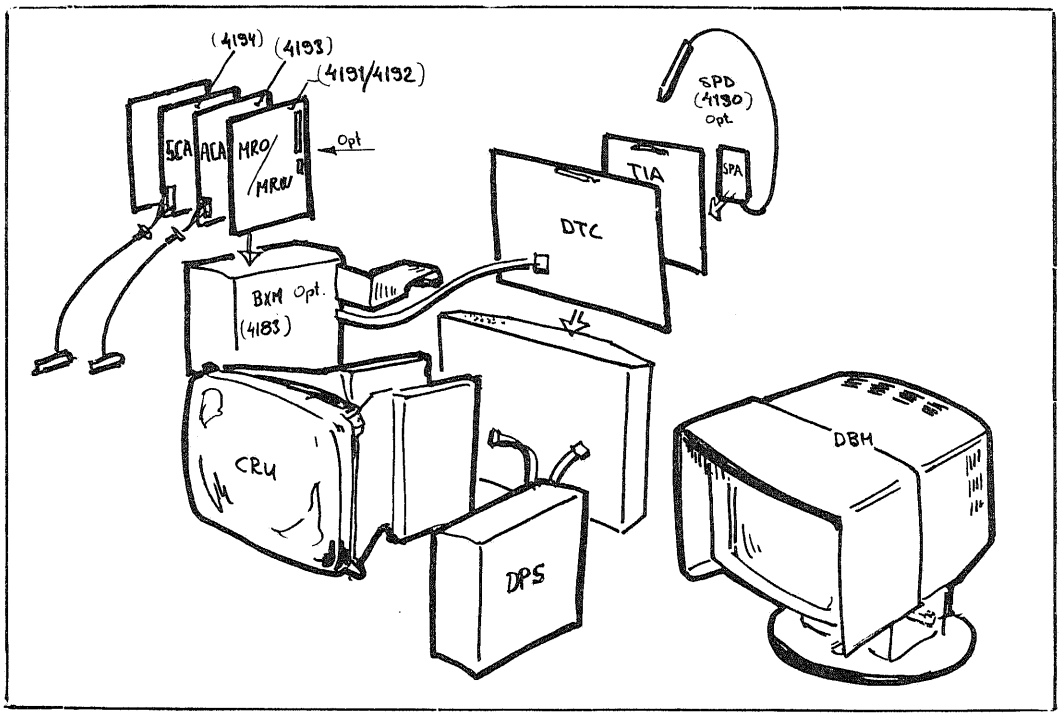


Fig. 1 . Mechanical layout of DU 4110

PRESENTATION

Presentation of characters on the screen is made using a raster scan method, i.e. the electron beam (cathode ray) makes a number of consecutive horizontal sweeps over the screen. The beam is modulated by the video information, that is the dots

that build up the characters, the cursor, the underlines etc.

The regeneration rate is 50 Hz, i.e. the electron beam builds up a whole picture on the screen every 20 milliseconds.

The total number of sweeps is 426. Depending on screen format, 400 (up to 25 text lines) or 396 sweeps (33 or 44 text lines) are used for presentation, spacing sweeps included. Thus, 26 or 30 sweep times are used for vertical retrace.

The sweep time is 46.95 µs

Each sweep is divided into 100 character positions

Out of which up to 80 may be used for presentation

Thus, the character position time is 0.469 µs

Each character position is divided into nine dot positions or columns (col 0 to 8) Seven columns (col 1 to 7) are used for character presentation. With DTC-A, columns 0-8 can be used for presentation.

The dot time is thus 52.2 ns

The corresponding dot rate is 19.17 MHz

This is equal to 1/18 of the system clock (Ø1) period.

Thus, the dots of a specific sweep for two characters are presented during one system clock period.

Presentation Geometry, Character Layout

The effective screen size is 180 by 258 mm. The horizontal distance between two adjacent dot positions is ≈ 0.36 mm. The vertical distance between

two adjacent dot positions (sweeps) is≈ 0.45 mm.

(Note that horizontal lines are not, as vertical lines, formed by discrete dots but by a continuous stroke of the electron beam.)

Thus, the ratio $\frac{\text{vertical dot spacing}}{\text{horizontal dot spacing}} = 1.25$

This is valid for the case of 9 horizontal dot positions (columns) per character position (Alpha-numeric display).

The number of vertical dot positions (sweeps) per text line is programmable and depends on the number of text lines on the screen.

Number of textlines	13 or 17	25	33	44
Number of sweeps/textline	22	16	12	9
Number of sweeps used for characters	13	13	9	8

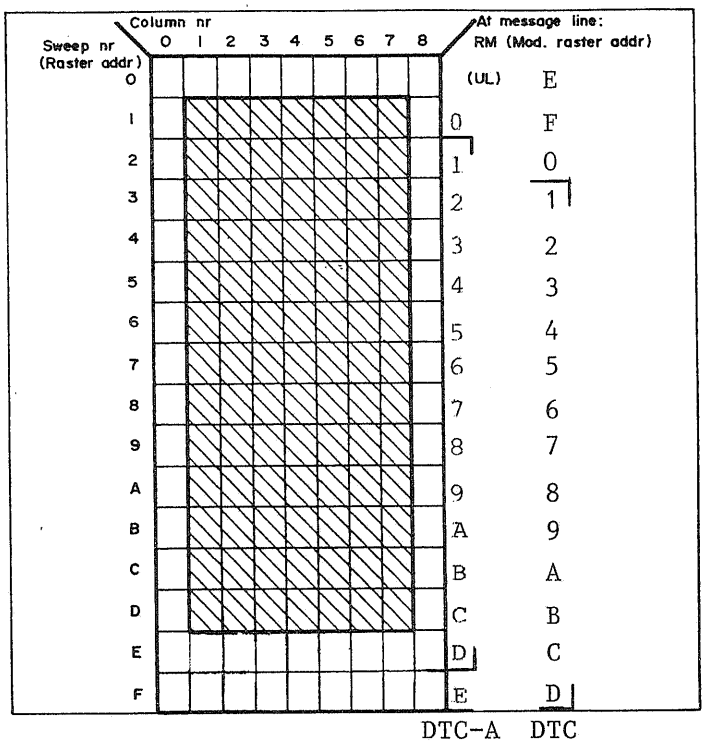


Fig. 2. 7x13 dot matrix in 9x16 character cell

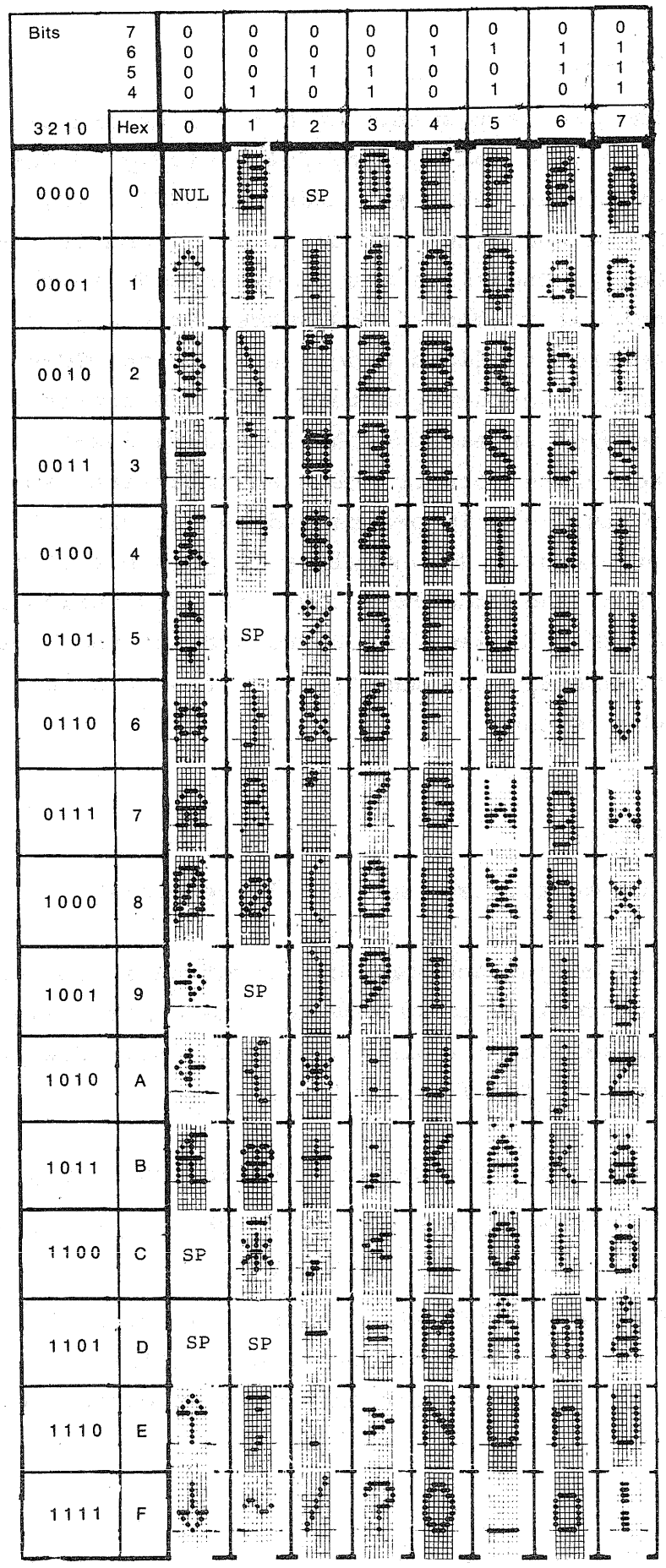


Fig. 3. Example of character layouts in character gen. code table.

- Underline

The two last sweeps of a textline are used for underline and spacing respectively. Underline starts at Col 1 and ends with Col 7 (DTC). For DTC-A underline starts at Col 0 and ends with Col 8. Underline may be generated via char. generator (underlined space) or via attribute generator (field underline). (With DTC-A all underlines are generated via the attribute generator except for the 33 and 44 (32+1, 43+1) text-line formats where an underlined space (Cols. 1-7) is programmed in the character generator.) No underline is possible at message line. The underline logic is used during sweep 0 (RM = E, see Fig. 2) to produce the line separating the message line from the rest of the screen.

- Cursor

The cursor is individually generated.

Its height and vertical location in the character cell is programmable. The cursor blink rate is selected; either 1.6 or 3.1 Hz. (See CRTC description for details.)

Two types of cursor may be selected by program:

Cuform = 0 : Width 9 dots. Character not visible at steady cursor.

Cuform = 1 : Width 9 dots. Character at cursor position always visible.
(Text inversion at cursor position.)

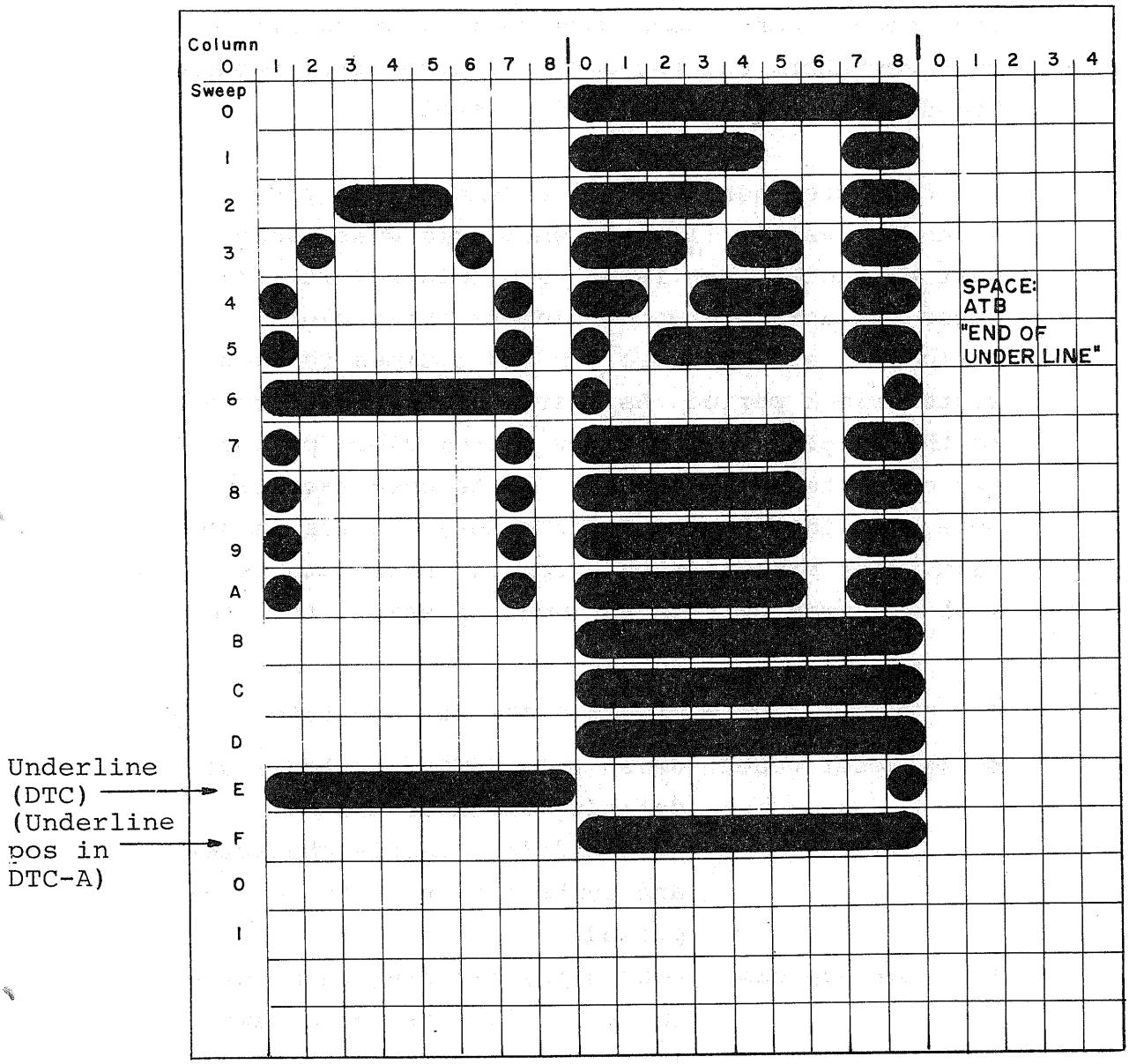


Fig. 4. Character, underline and cursor layout example

Note: Inversion of "4" caused by maximum height cursor with Cuform = 1. Field underline of "A" and "4". Valid for formats with 25 textlines, DTC.

Principles of Character Generation

See DTC block diagram Fig. 5 and character generator code table Fig. 3. The eight-bit codes for character and symbols to be displayed, as well as for attribute characters (ATB), are stored in the part of the RWM area that is defined as display memory at a certain time.

Note that these codes may differ both from the line codes (codes used when communicating with the host computer) and the codes received from the key-board when keys have been depressed.

The character generator is a memory that produces seven dot values (1 or 0) when told what sweep of what character that is to be displayed. It produces the dots of two characters every clock period (Ø) as there are 2 x 9 dot times to each system clock period. As there is only one access to the display memory every system clock period, two character codes have to be fetched each time. Note that the LS bit of the memory address is not used. Thus presentation unconditionally starts with a character from an even address in the memory.

The address inputs of the character generator are:

- Character code — Seven bits (DTC-A:eight bits), defining which of the 128 (DTC-A:256) possible characters and symbols that is to be displayed.
- Raster address — Four bits, defining which sweep (0, 1 ... 15 (max) of a text-line that is being presented. For DTC-A the raster address consists of five bits.

Example: The character codes for A and 4 (see Fig. 4) have been entered for the fifth time into the double byte latches. (See block diagram) Thus the binary value of the raster address is 00100. First the seven bit code for A (41_{16}) is applied to the character generator. The seven outputs of the character generator will present 1000 001, where the ones correspond to the two dots of that sweep. As the code for A is a valid displayable character code, the video register will be loaded with the seven bits from the character generator (plus a Col 0 zero for spacing). Next, while these dots are

being shifted out, the character generator will produce 0100010, i.e. the dots of the fifth sweep of the number "4". Next, the two eight bit codes for the following two characters of the textline are addressed and entered into the double byte latches, etc.

The "display memory" addresses as well as the raster addresses are provided by the programmable LSI circuit CRTC (Cathode ray tube controller).

DISPLAY TERMINAL CONTROLLERS

General

A general description of the components and properties that are similar in the System 41 microcomputers (communication processor (CP), display terminal controller (DTC) and to some extent flexible disk processor (FDP) is found in the chapter "Microcomputer". Here the functions that are unique to the DTC as well as the specific use of certain functions will be described.

The main functional parts of the DTC are:

- MPU
- Memory part
- Address decoder
- Interrupt logic
- Timing logic
- Presentation logic
- Keyboard interface

Fig. 5 contains a block diagram of the DTC, the two-wire interface adapter (TIA) and the optional synchronous communication adapter (SCA), which is used in single display units communicating synchronously with a host computer. Fig. 9 presents the block diagram of a display unit including DTC-A and TIA-A.

The MPU is described in the "Microcomputer" chapter.

Memory

The display terminal controller RWM is built up of two parallel blocks of dynamic memory, (2 x 16 kbytes).

There is a refresh and memory access multiplexing logic, governing the memory function.

It provides

- Row address accompanied by Row address strobes
- Column address accompanied by Column address strobe
- Write strobe
  according to the state of the timing logic, the address bus (odd or even block addressed), and the R/W signal from MPU.

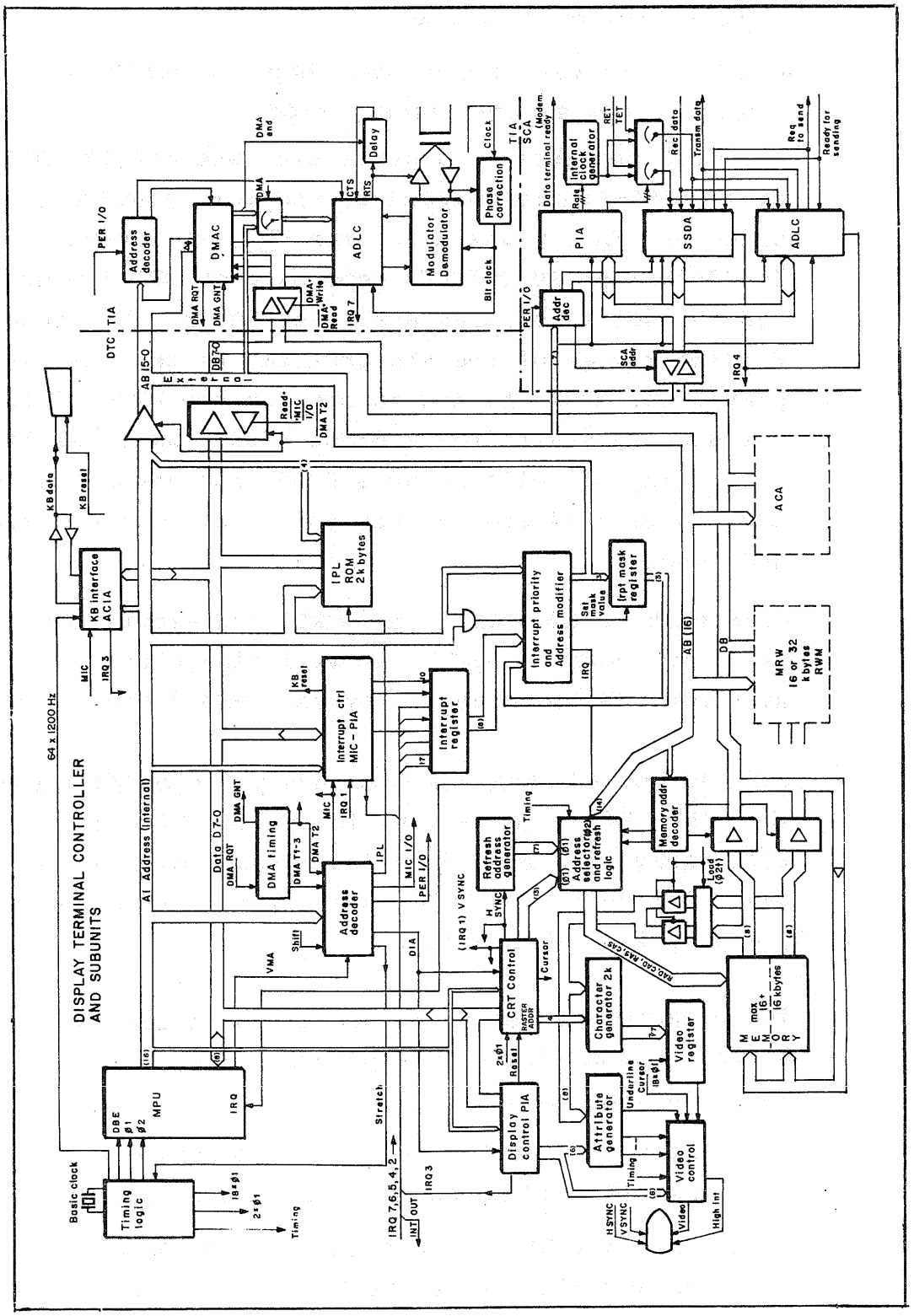


Fig. 5. DTC and subunits. Block diagram

The memory is addressed by

- MPU (or DMA logic of TIA, two-wire interface adapter). This happens during the Ø2 period of the system clock cycle.
- CRTC or refresh logic. This happens during Ø1 half of the system clock period.

The CRTC provides 14 address bits whereof 13 (DTC) or 12 (DTC-A) are coupled to the memory address selector. This means that, for DTC, 16 kbytes (0000-3FFF or 4000-7FFF, depending on DTC jumper J1) can be used as display area. For DTC-A, the highest 8 (4+4) kbytes of the RWM area can be used as display area. Thus, trying to program the DTC-A CRTC for display start from address 4000 will result in display of cells in area 6000 and upwards since address bits 14 and 13 sets to "1" as soon as CRTC addresses.

Note that as the LSB of the CRTC address is not used, two consecutive bytes will always be accessed. (Bytes 0 and 1, bytes 2 and 3 etc.)

Refresh addressings are made every cycle (Ø1 time) when the CRTC reset signal is active. Five refresh addressings are also made during each display sweep following the HSYNC signal, i.e. when no presentation takes place.

The dynamic RWM cells are organized as 128 rows by 128 columns (16 kbytes blocks). A cell is accessed like this:

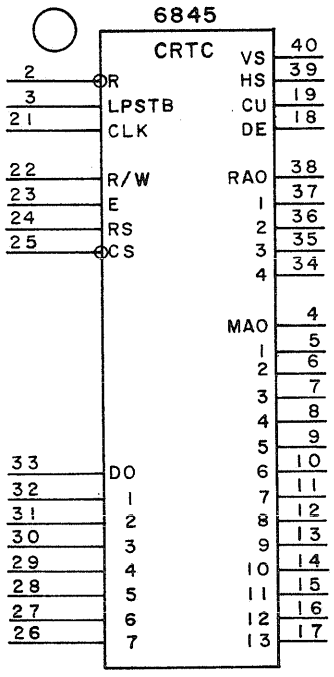
1. Address from CRTC or External address bus is chosen (depending on half of system clock cycle).
2. Address bus bits 7 through 1 are strobed in as Row address by the Row address strobe RAS (For accesses from the External bus, only one block gets RAS).

3. The appropriate column is selected when CAS (Column address strobe) strobes in address bits
   1. 13 or 12 through 8 (CRTC access). (Bit 14 (DTC) or bits 14 and 13 (DTC-A) are provided as constant bits.)
   2. 14 through 8 (External bus access).

4. A write operation is carried out (in case of External bus addressing) as Write strobe is applied.

According to the LSB of the External address, the memory address decoder produces a row address strobe enable for the proper memory block and, in case of a read memory operation, an output enable for the proper set of output buffers. The address decoder consists of a ROM. It has also the R/W signal and the most significant address bus bits and display area strap as input signals. Please refer to the memory map, Fig. 8 and DU 4110-002 memory map, Fig. 10.

Cathode Ray Tube Controller (CRTC)



MPU_Interface

The CRTC interfaces the Microcomputer (MIC) via the data bus, R/W, enable lines, a reset line and one address line. The reset line (R) must be kept low during initial programming of the CRTC.

It contains one write-only address register and 15 other registers of variable length, that are accessible from the MIC bus. As there is only one address line, either the address register or one of the other locations can be addressed at a certain time. The value of the five bit address register decides which one of the other locations is addressed. Register nr. 5 (R5) is e.g. addressed when contents of address register is XXX00101 and (RS) (address line) is high.

Table 1. CRTC registers

Register	Nr of bits	Name	Function
R0	8	Ch.times per sweep	Tells total nr of horizontal char. pos.times per sweep minus one.
R1	8	Characters per line	Tells how many char. should be displayed on a line.
R2	8	HSYNC pos.	Tells at which horizontal character position HSYNC signal should be generated.
R3	4	HSYNC width	Tells the nr of character times HSYNC should be active.
R4	7	Lines total	Total nr of lines (displayed + non-displayed) minus one.
R5	5	Vertical total adjust	Tells how many sweep times should be added to the "Lines total" time to form a full frame time.
R6	7	Lines displayed	Tells how many lines should be displayed.
R7	7	VSYNC position	Tells after what line VSYNC should be generated. (VSYNC active during the time of one textline)
R8	2	Interlace control	(00 or 10 for non-interlace)
R9	5	Max raster address	Tells the number of sweeps per text line minus one.
R10	B+P+5	Cursor, start + blink	Tells on what sweep of line top of cursor is. Tells whether cursor blinks and frequency.
R11	5	Cursor end	Tells what sweep of a line cursor bottom is on.
R12 R13	14	Display start address	Tells at which address (with which character) in a display memory the display should start.
R14 R15	14 R/W	Cursor location	Points out at which linear address (character position in memory) cursor should be displayed).
R16 R17	14	Light pen hit address	Latches linear (character) address when LP hit (LPSTB).

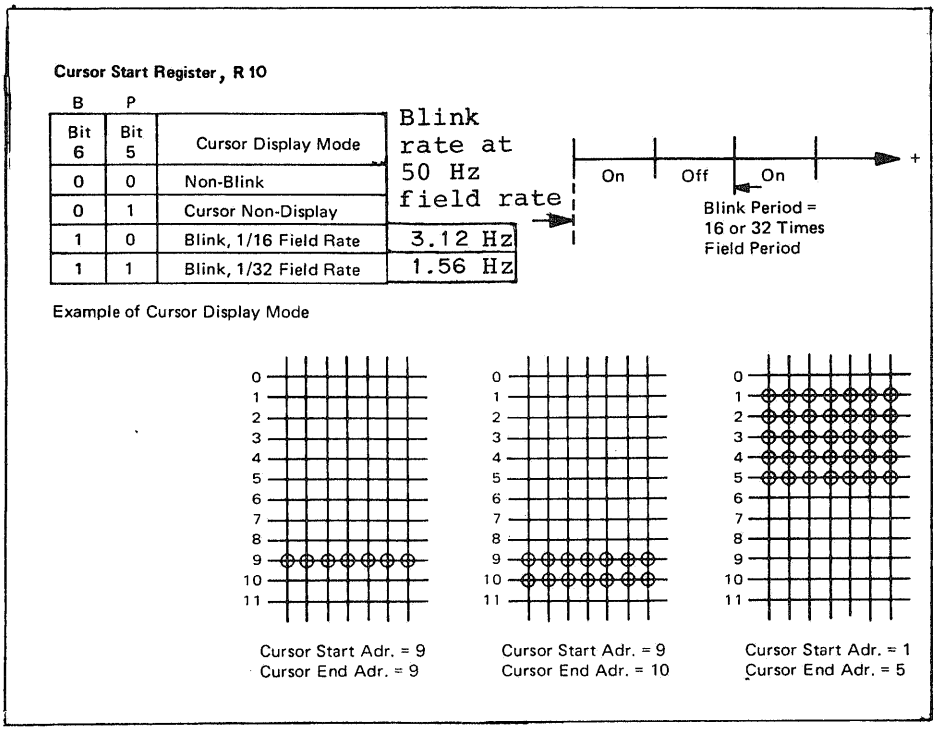


Fig. 7. Cursor control

LPSTB (Light pen strobe) When this signal goes high, linear address (display memory address) is latched into R16 R17, thus detecting the selector pen position on the screen.

CLK signal is the basic clock for linear address counting etc., i.e. the character clock, which should provide one clock period for each horizontal character position.

Video Control and CRT Interface

DE is the master enable signal for the video generating circuitry. This signal is active during the display time of the displayed lines, i.e. during a total part of the frame time decided by R1 and R6.

CU is active during the time when cursor dots should be produced by the video generating circuits.

RA4-0 is the raster address bus to be fed to the character generator together with the code for the actual character. For each text line it tells the actual sweep number,i.e. provides values from 0 up to the programmed value of R9.

HS(YNC) is the signal which tells the CRT logic that the cathode ray should retrace and start next sweep over the screen.

VS(YNC) is the signal governing the vertical retrace of the ray, that precedes the regeneration of an image on the screen.

Linear Address Generation

MA13-0 provide the address to the display memory of the actual character to be displayed, i.e. the actual value of the internal linear address counter. This counter is started at a value decided by R12 R13 and is stepped as many times as decided by R1 for each sweep. For each text line it counts through the same address sequence as many times as decided by the value in R9.

Then it goes on after next HSYNC and makes the same sequences for each displayed line i.e. as many times as decided by R6. This whole sequence starts over again after next VSYNC, unless the programming of the CRTC has been changed.

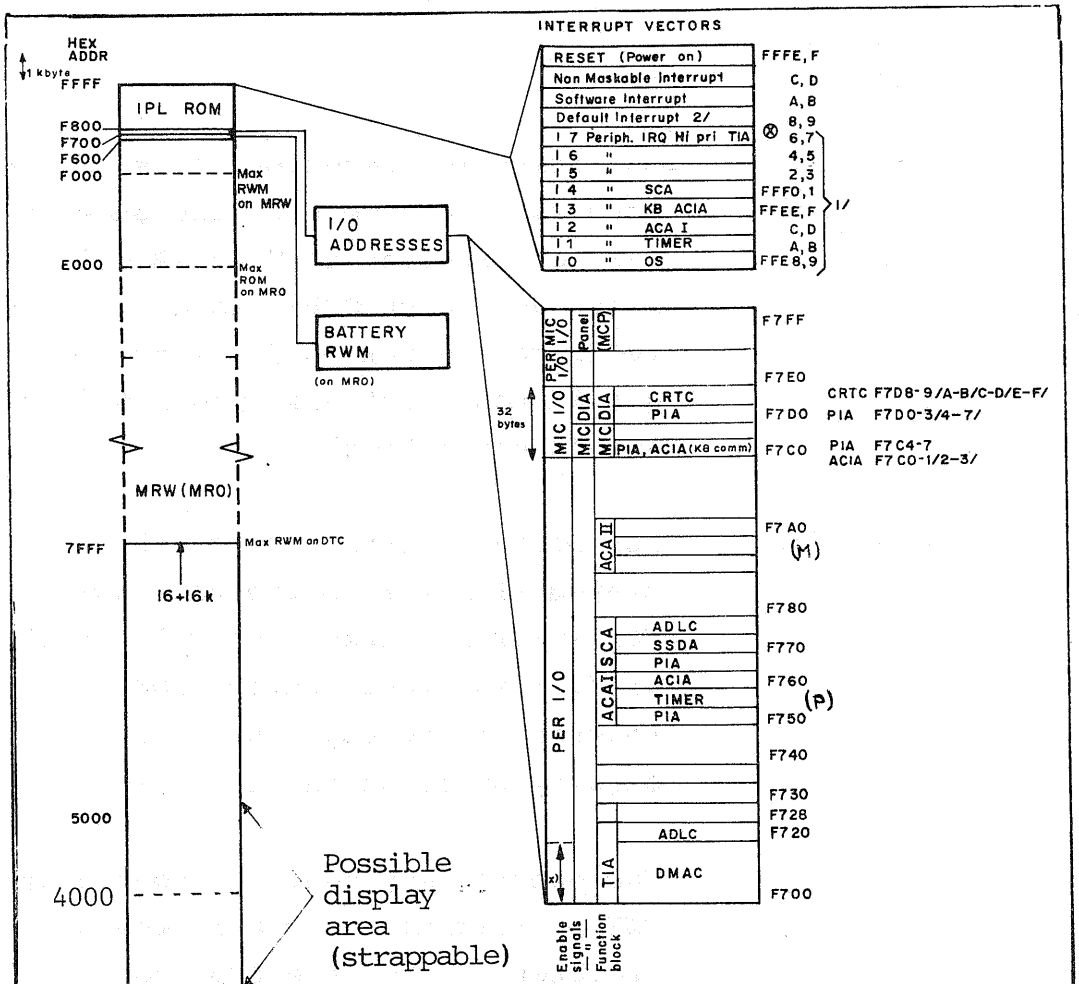
The MA bus must be multiplexed with the MIC address bus, to enable the MPU to write new characters into the display memory.

Interrupt Logic

The interrupt system principles are described in the "Microcomputer" chapter. The locations of interrupt vectors are found in the Memory map. Here follows a description of the origin, properties and use of the different interrupts in the DTC with subunits.

- Int. out — is an active low signal that may be sent from DTC to interrupt another controller. It is generated by setting MIC-PIA per reg. A bit 5.

- Power on — An active low Reset pulse is generated at power on. This Reset signal is used to reset circuits throughout the system and to initialize the MPU. The MPU will fetch contents of cells FFFE and FFFF to the program counter when Reset is deactivated.



x) Not present if during one of the three DMA cycles DMA T1-T3

ACA = Asynchronous comm. adapter
DIA = DTC, Display adaptation part
MCP = Microprocessor control panel
MIC = (DTC) Microprocessor part
SCA = Synchronous comm. adapter
TIA = Two-wire interface adapter

x Note:

1/ A (read or preferably) write operation into one of the cells FFE8-FFF7 will result in a setting of a corresponding mask value. As an example, FFEE or FFEF from MPU will disable interrupts of pri <3.

2/ MPU reading FFF8-9 will make contents of two out of cells FFE8-F9, depending on highest pri valid interrupt, transfer into MPU.

Fig. 8. Display terminal controller and subunits, memory organization.

- Reset button (DTC) — See NMI (DTC) below

- Reset button (DTC-A) — An active low Reset pulse is generated to reset the MPU. The same signal is also sent to MIC-PIA CB1, which activates CRB-7 to a "1". The software can read MIC-PIA to find out if the start was "power on" or the "Reset button".

- NMI (DTC) — Non-maskable interrupt may be generated either from the test unit MCP (Microprocessor control panel) or from the reset button located underneath the upper part of the display unit, on the left side.

  The program may test the source of NMI by reading the MSB of the A register of the interrupt control PIA (MIC-PIA):
  Test MSB (b7) of cell F7C4. If

1	0
NMI from pushbutton	NMI from MCP

- NMI (DTC-A) — Non-maskable interrupt may be generated either from the test unit MCP (Microprocessor control panel) or from the PC-logic.

  The program may test the source of NMI by reading the LSB of the A register of the interrupt control PIA (MIC-PIA):
  Test LSB (b0) of cell F7C4. If

1	0
NMI from PC-logic	NMI from MCP

- SWI — Software interrupt is an instruction (see MPU description).

- Default interrupt — Interrupt vector is fetched from cells FFF8 and FFF9 when no valid interrupt signal (I7 to I0 of a higher level than or same level as the present mask value) is present at the time of the vector fetch. See "Microcomputer" Figs. 2 and 11.

- I7 — is the highest level maskable interrupt. It will always (irrespective of interrupt mask register value) result in an IRQ signal to MPU. An IRQ is however, only serviced if MPU I-bit is reset. I7 originates from the ADLC (see ADLC description) on TIA (Two-wire interface adapter).

- I6 — Spare

  No latching possibility

- I5 — Spare

- I4 — is an interrupt that may originate from a non-steady source (short pulse interrupts). It may be latched by the interrupt logic. If MIC-PIA A-side bit 6 is set, an incoming interrupt will be latched. If it is reset any latched interrupt will be reset and the state of the I4 line to the interrupt register will be equal to state of the incoming interrupt line. This line is used by a "modem or computer adapter" e.g. SCA or ACA.

- I3 — is the interrupt used by the KB interface ACIA and the selector pen function (if used).

- I2 interrupt line functions as lines I7-I5 and I3. It is typically used by a printer interface ACA. (ACA I)

- I1 is initiated by the VSYNC signal if enabled (by MIC-PIA CRA bit 0 = 1) Bit 1 of the same control register should be = 1 for the PIA to produce an interrupt at the positive edge of VSYNC.

- I0 If MIC-PIA CRA (at address F7C5) bits 5 and 4 = 11 the program will produce an interrupt at this level when writing CRA bit 3 to a zero.

Keyboard Interface

The interface between DTC and KBC (Keyboard controller) consists of

- + 5V/0V (to KBC/KXB/MID)

- KB reset line from DTC, used to initialize the KBC. KB reset is generated at power on and by programming MIC-PIA A-side register (see above, "MIC-PIA Programming").

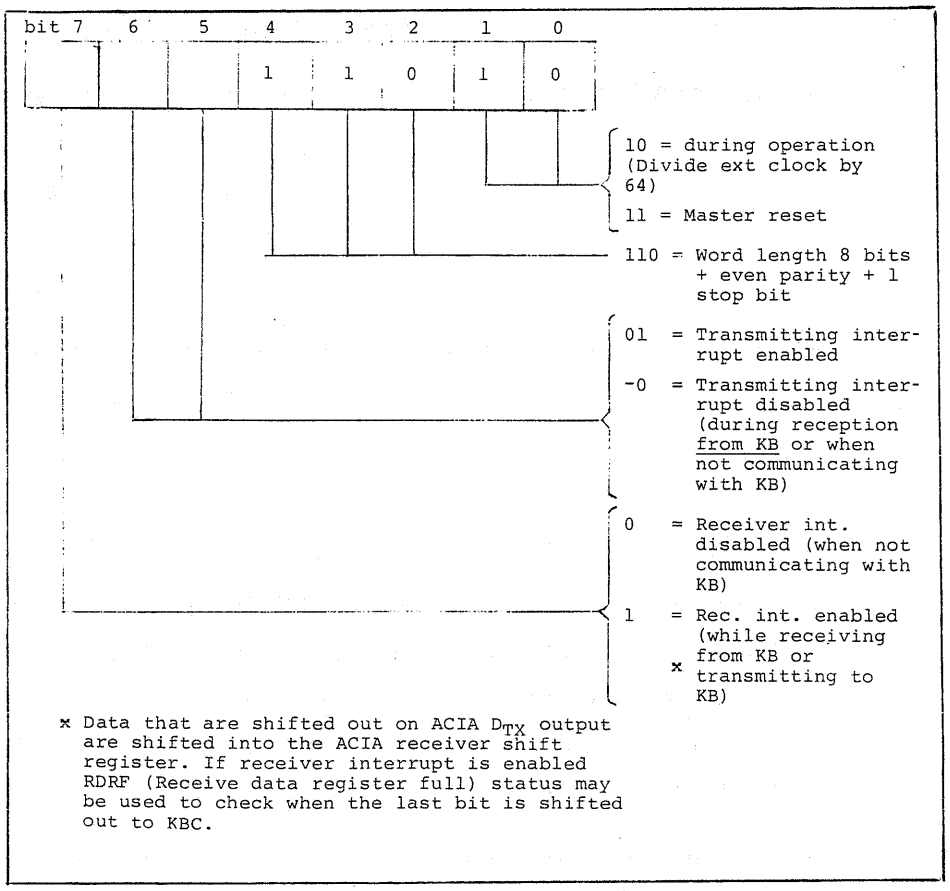
- KB data, bidirectional line used for asynchronous transfer of
  - Order, e.g. "alarm on/off" or "Transmit ID bytes" and data, e.g. lamp data to KBC.
  - Status, e.g. "shift key depressed" or "ID card in" and, if needed, key number or ID characters from KBC.

This communication is initiated by DTC, i.e. no transmission from keyboard takes place unless DTC has sent a message to keyboard.

The communication DTC-KBC is described in chapter "Keyboard". It takes place via the KB interface ACIA on DTC. The ACIA is described in chapter "Microcomputer". A clock of 76 kHz is provided for the ACIA by the timing logic. ACIA divides this clock by 64 (Bit transfer rate 1188 Hz) and synchronizes internally to received data. Furthermore ACIA may provide interrupt (at level 3) when transmit register is empty, receive register is full or an error has occured.

KB Interface ACIA Programming

ACIA control register at address $F7C0_{16}$



Attribute Characters and Registers Controlling the Display

The eight bit words of the "display memory" may be character codes or attribute character codes (ATB). The ATB are used together with static control bits to define various properties, other than the text pattern, of the different fields on the screen.

Note that one ATB may result in more than one new attribute of the associated field. The field attributes that may be caused by attribute characters (ATB) are

ATB:	If no ATB:
• Underline	Underline may only result from code $5F_{16}$
• Blinking text (1 Hz)	Steady text
(Cursor has independent blink function)	
• Inverted field	Inversion may be caused by 1. Tot. inv. bit (whole screen inverted) 2. Cursor with Cuform = 1
• Extinguished field	Presentation according to display memory contents.

- High intensity — Normal intensity
- Message line (no other attributes, including field underline, permitted. Codes 80_{16}, defining unprotected field an $A0_{16}$, defining protected field can of course be used as they are interpreted by the software).

Note that an ATB in the display memory will result in one empty space on the screen (UL (if not reset by the ATB) and cursor may still be displayed).

Underline of a field starts at the character pos. following the space resulting from the UL ATB. It ends before the space corresponding to the ATB defining "End of underline". (See Fig. 4.) Inverted field starts at ATB position.

Apart from the control registers of the CRTC, defining the display area, vertical cursor format and cursor blink etc. there are two eight bit registers of a PIA that either directly control some video attributes or define how to interpret an ATB.

Note that the PIA definitions are static, i.e. cannot be made for specific fields on the screen as there is no simple connection between display timing and program but by the VSYNC signal.

The directly controlled attributes are:

- Cursor format: Cuform = 0 or 1, the latter with text at cursor pos. inverted
- Total inversion: Whole screen (but inverted field and, if Cuform = 1, text at cursor position) presented as dark text on a clear background.
- Underline position (should be programmed to point to the next but last sweep of the text-line).
- Inhibit video (whole screen dark)
- Wraparound (line and page end) of ATB.

The ATB definition bits are:

- Enable reset of underline by space/nul (2 bits)
- Enable underline of field by character code $5F_{16}$

- Interpret ATB bit 1 as Inverted field/Underline/ Blinking field (2 bits)
- Select: 64 or 128 possible ATB:s

Furthermore, reset of the CRT controller is ordered via the display control PIA, (DIA-PIA).

Display Control Registers. Detailed Description

Register	Address (hexadecimal)
PIAA (Peripheral reg. A)	F7D0 (PIA CRA (at address F7D1) bit 2 must be one)
PIAB (Peripheral reg. B)	F7D2 (PIA CRB (at address F7D3) bit 2 must be one)
CRTC	F7D9 Address to several registers[x] controlling screen format including textline height, location, height and blink rate of cursor, display start address etc. See CRTC description above.

x) The contents of the CRTC address register (at address F7D8) decide which one of the CRTC registers that is accessed.

DIA-PIA (A-side Peripheral Register $F7D0_{16}$). (DTC)

PIAA bits	Output on pin nr.	Signal name	Function if = 1	Function if = 0
PA 7	(9)	____	(Not used)	
6	8	Reset CRTC	The CRTC will produce no mem.addr. VSYNC, etc.	CRTC will act according to programming (!)
5	7	5F function	1. Field underline only by char. code $5F_{16}$ 2. Char. codes 00 and 20_{16} will also act as "End of 5F underline" char. if PB2,1 = 01	Field UL only by ATB with[x] DL1 = 1 (if PA 3,2 = 10). 00,20 and $5F_{16}$ only affect char. generator.
4	6	Cuform	Cursor with inversion of underlying character	Cursor covering character
3,2	5,4	DL1 function select bits	00: Attribute without INV/UL/BLK when with DL1 = 1 01: Inverted field because of ATB with DL1 = 1[x] 10: Field underline because of ATB with DL1 = 1 if PA 5 = 0 x 11: Blinking field because of ATB with DL1 = 1[x]	
1	3	ATB code def.	Char. codes 80-BF_{16} interpreted as ATB (bits 7,6 = 10)	Char. codes 80-FF_{16} interpreted as ATB (bit 7 = 1)
0	2	Total inversion	Normal pres = "Black on white" (whole screen image inverted)	Normal pres = "White on black" Only text of inv. field or text at cursor pos. may be dark on clear background.

x Notes: 1) The attribute character bit (DL1) is not effective if appearing on message line

2) DL refers to the character code (8 bits; DL7 - DL0)

DIA-PIA (B-side Peripheral Register $F7D2_{16}$). (DTC)

PIAB bits	Output on pin nr.	Signal name	Function if = 1	Function if = 0
PB 7	(17)	(Not used)		
6,5	16,15	UL sweep	00: UL pos activated at sweep E_{16} (15:th sweep) 01: UL pos activated at sweep A_{16} (11:th) 10: UL pos activated at sweep 7 (8:th) 11: Not used. UL pos never activated.	
4	14	°Inhibit video	Presentation of display area contents except for extinguished fields (defined by ATB)	Screen dark
3	(13)	(Not used)		
2,1	12,11	Enable reset of underline by SPACE/NUL	00: Undef. function (No NUL/SP UL reset) 01: NUL/SPACE ($00/20_{16}$) may act as UL reset characters if PA5 = 1 10: Function undefined (No UL reset by SP/NUL) 11: Function undefined (No UL reset by SP/NUL)	
0	10	°Wraparound	ATB valid until "off" ATB (including 00 and 20_{16}, if selected for reset of "5F underline" or until line end, whichever occurs first.	ATB valid until an "off" ATB (including 00 and 20_{16} if selected) irrespective of line or page end. If all ATB are erased, (overwritten), though, attributes will be reset before less than 40 ms.

(DTC-A: See pages 44-45)

How Do You Do .... to Assign an Attribute to a Field on the Screen

Desired field attribute	Preparatory programming of DIA-PIAA	1/ "On" ATB from display memory
	PA76543210	DL76543210
Inverted field (dark text on bright background if PIAA:0=0)	-0--0100 or-0--0110	1-----1- 10----1-
Blinking field	-0--110- or -0--111-	1-----1- 10----1-
Underline	-00-100- or -00-101- or -01-----	1-----1- 10----1- 01011111 ($5F_{16}$)
High intensity	-0----0- or -0----1-	1---10-- 10--10--
Extinguished field x)	-0$\overset{x)}{-}$--0- or -0----1-	1---11-- 10--11--

DIA-PIAB should be programmed ---1---- to permit presentation.
DIA-PIAB bits 6 and 5 must also be properly programmed, defining UL sweep. Bit 0 defines whether attributes should wrap around line end. See below.

Note: 1/ Not effective on message line

How and When Does the Attribute End?

All field attributes will be reset by

1. Msg line ATB: 11111111. If wraparound bit (PB0) active (=0), attributes will wrap around past message line (from next but last to first text-line).

2. Valid ATB, lacking "on" bit (combinat.): 1------- with PA1 = 0 or 10------ with PA1 = 1

3. Textline end if PB0 = 1 (No attribute wrap-around)

Underline may also be reset by SP or NUL (20_{16} or 00_{16}) if PA5 = 1 and PB2,1 = 01.

The list of "on" ATB:s above shows that the possibilities of combining different attributes for a field at a certain time are limited. A field that is both blinking and inverted may e.g. not be generated, whereas a blinking field with high intensity may be generated. Note that inverted field or blinking field may be combined with field underline by $5F_{16}$ if inverted field/bl. field ATB precedes $5F_{16}$.

MIC-PIA Programming (DTC)

Initial condition: All registers reset.

1)

DDRA at address $F7C4_{16}$ is set to 011--01-: PA7 and 2 inputs; PA6,5 and 1 outputs. (See below, point 3).)

2)

CRA at address $F7C5_{16}$ is set to

--11-111

- 1 because of VSYNC enabled
- 1 because of positive transition on VSYNC (↑)
- PIA per. register A accessible
- Bit 3 value will be present at CA2 output "0" = Interrupt on level 0 "1" = No interrupt on level 0
- CA2 functions as output

3)

PIA A-side register (address $F7C4_{16}$) bit functions

PIAA bit	Pin no	Input/ Output	Signal name	If = 1	If = 0
PA 7	9	Input	Button/MCP NMI	NMI from DU button	NMI from MCP (P4: 1=low)
6	8	Output	I4 latch enable	I4 will be latched	No I4 latch
5	7	Output	Int. out	Int. out on P1:7	No Int. out
4	(6)	——Not	Used (IN DTC)———		
3	(5)	——Not	Used (IN DTC)———		
2	4	Input	MCP test	MCP not in test mode	MCP in test mode
1	3	Output	KB reset	KB reset to KBU	No KB reset *
0	(2)	——Not	Used———		

*) Note: At initialization a KB reset pulse will be sent as DDRA contains all zeros: PA 1 functions as a high impedance input: "active level" for KB reset generation.

DTC-A PRINCIPLES

DTC-A has the following, additional functions, as compared to the DTC board:

- parity check (PC) in dynamic R/W memory
- increased memory beyond 64k bytes
- presentation from two display memory areas at the same time.
- program loadable character generator (dot. matrix memory). (TIA-A must be used)
- changed "Reset" function, (see Interrupt Logic).
- screen format $<$ 24 lines is possible.

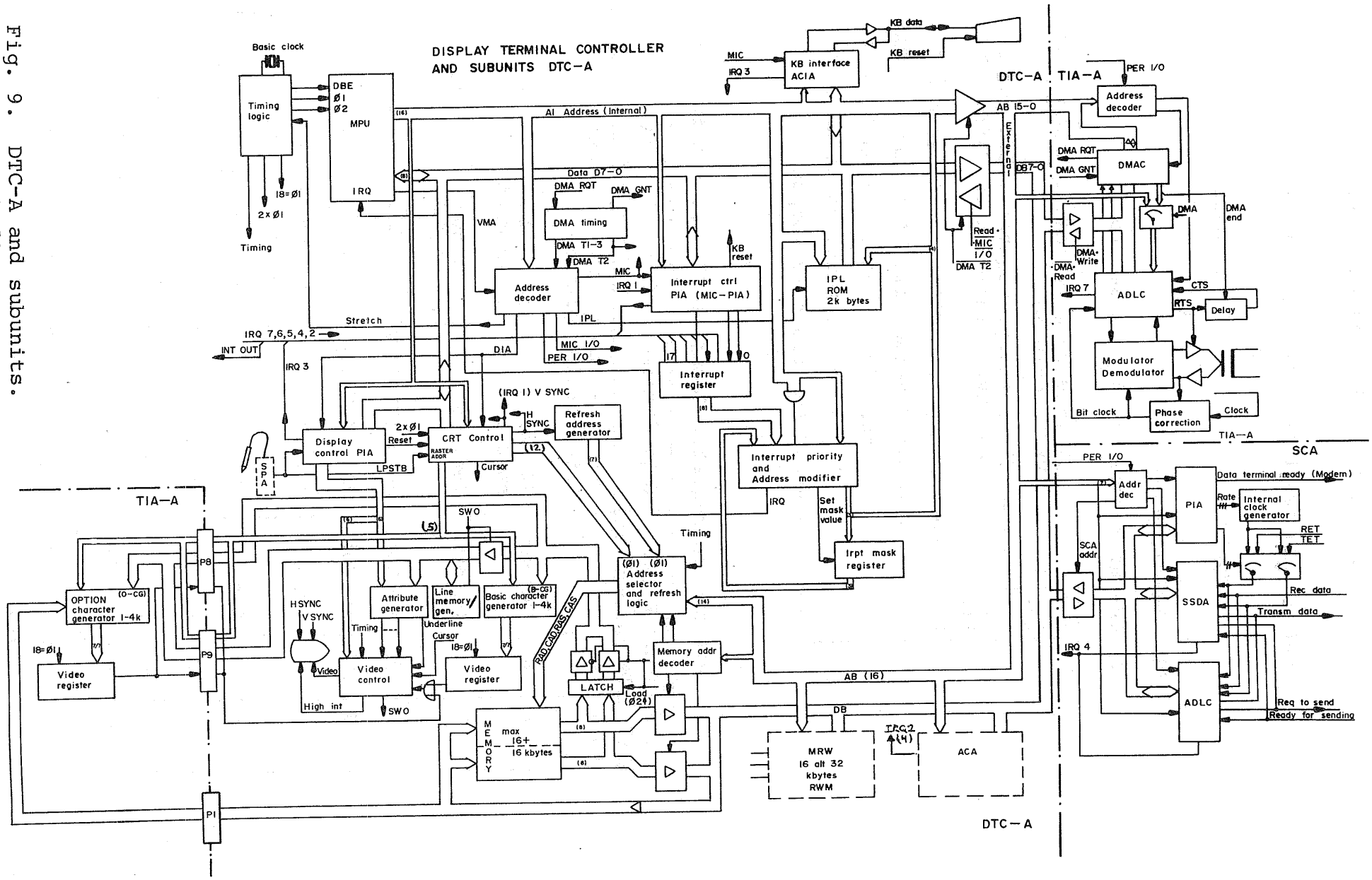


Fig. 9. DTC-A and subunits. Block diagram

MIC-PIA Programming (DTC-A)

Initial condition: All registers reset.

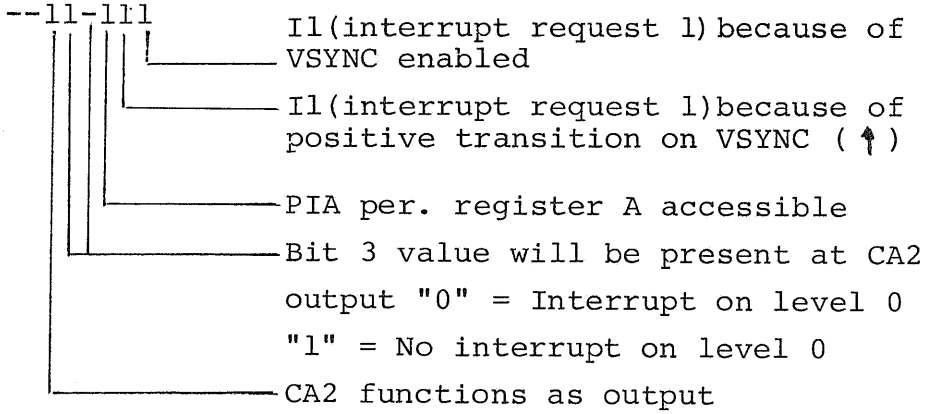
1)

DDRA at address $F7C4_{16}$ is set to 011--010:

PA 7		PA 6	
2	Inputs	5	Outputs
0		1	

(See below, point 3).

2) CRA at address $F7C5_{16}$ is set to



3)

PIA A-side register (address $F7C4_{16}$) bit functions

PIAA bit	Pin no	Input/ Output	Signal name	If = 1	If = 0
PA 7	9	Input	Connected to 0V to be compatible with earlier program		
6	8	Output	I4 latch enable	I4 will be latched	No I4 latch
5	7	Output	Int. out	Int. out on P1:7	No Int. out
4	(6)	— Not	Used (IN DTCA)		
3	(5)	— Not	Used (IN DTCA)		
2	4	Input	MCP test	MCP not in test mode	MCP in test mode
1	3	Output	KB reset	KB reset to KBU	No KB reset *
0	(2)	Input	Parity Error	Parity Error	No Parity Error

*) Note: At initialization a KB reset pulse will be sent as DDRA contains all zeros: PA 1 functions as a high impedance input: "active level" for KB reset generation.

4)

CRB at address $F7C7_{16}$ is set to -111 1111 (Bit 7 is the flag bit set from the DU reset button).

5)

PIA B-side register (address $F7C6_{16}$) bit functions

PIA B bit	Pin No	Input/ Output	Signal name	If = 1	If = 0
PB 7	(	not used			)
6	16	output		Reset all circuits except MPU and MIC-PIA	No reset of all circuits
5	15	output		MPU Addr.-Mode 1	MPU Addr.-Mode 0
4	14	output	OC-G enable	0-CG x) enabled to Mic bus	0-CG disabled from Mic bus
3	13	output	Display memory	Display memory 4k	Display memory 8k
2	12	output	VMAX/VMA 1	VMAX generated by DMA	VMA 1 generated by DMA
1	11	Output	VMAX/VMA 1	VMAX generated by MPU	VMA 1 generated by MPU
0	10	Output	Reset PC-error	Reset PC-error FF or PC not used	Memory PC used

x) 0-CG = Option Character Generator

Memory Expansion

With the DTC-A board it is possible to access another 28 k over the 64 k that is possible to access from the DTC-board. DTC containing 32 k and MRW 32 k and out of which it is possible to access 28 k because of the IPL PROMs (see Fig. 10). The DTC-A board contain 32 k and the MRW board 64 k and it is possible to access 56 k depending on the IPL PROM. The base module with DTC-A (32 k) and a part of the MRW (28 k) is addressed when VMA 1 is active, the rest (28 k) on the MRW board is addressed when VMAX is active.

The software program decides which VMA that is to be activated when MPU or DMA addresses.

When VMAX is active, only addresses between 8000 and EFFF are influenced. Addresses out of this area, always activate address VMA 1, (see the memory organization Fig. 10).

The control by VMA is performed by:

MIC-PIA B-bit 1	"0" = MPU gives VMA1		
	"1" = MPU gives VMAX	Note MIC-PIA DTC-A	
MIC-PIA B-bit 2	"0" = DMA gives VMA1		
	"1" = DMA gives VMAX		

as can be seen MPU and DMA are completely independent of each other.

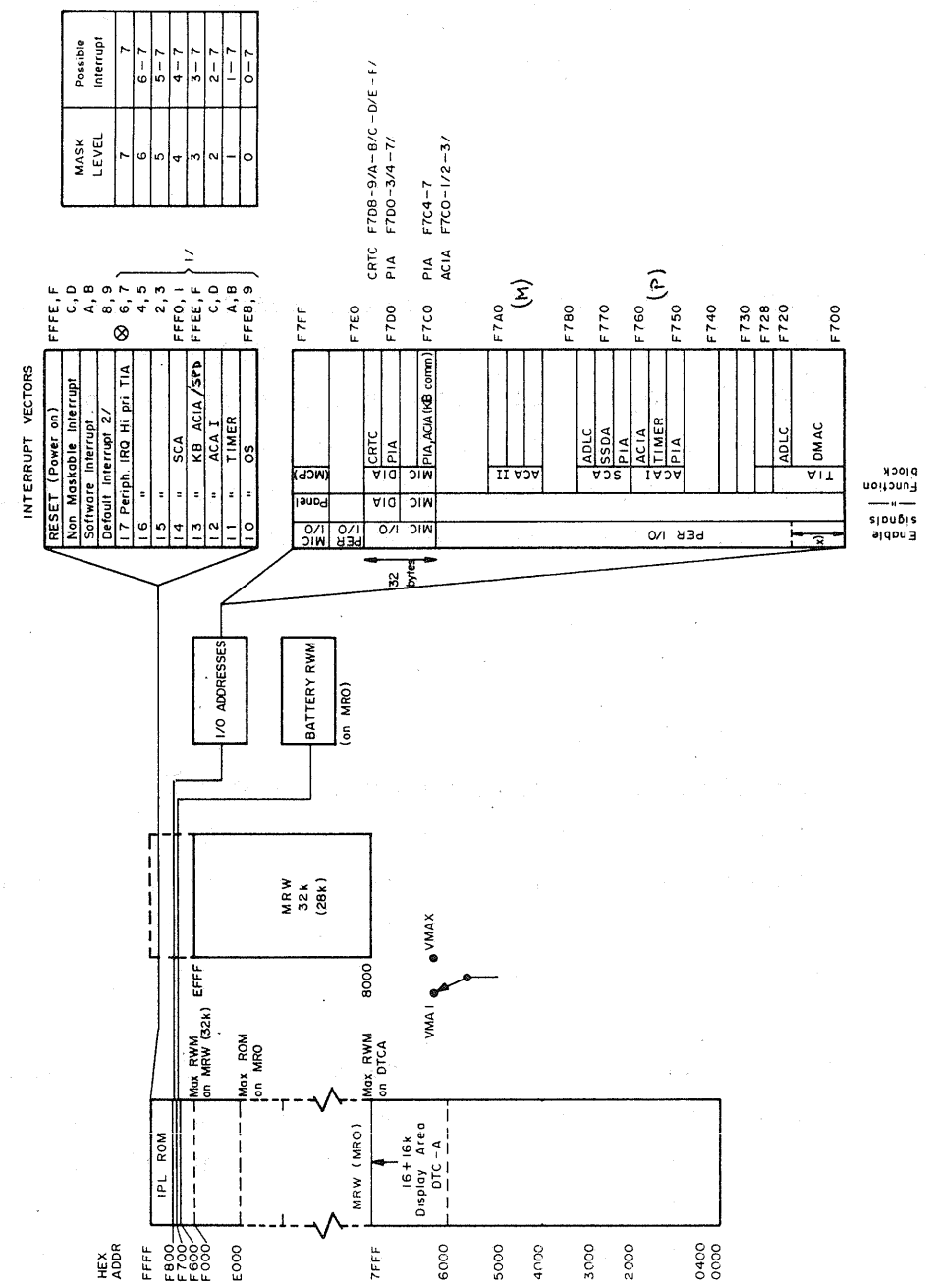


Fig. 10. Display terminal controller and subunits, memory organization (DTC-A).

EE360-810C

Parity Logic

The function of the PC-logic is to add an "extra bit" (a PC-bit) to the byte that is to be written into the memory. The value of this "extra bit" (PC-bit) is checked while the byte is read from the memory. The parity check function is odd (1000 1000 + 1 (PC)). The PC-error signal is connected to MIC-PIA.

Latch

LATCH is a double register, which is connected to the display memory area. Each DIA read cycle in the display memory area results in two bytes being stored in the LATCH. One byte at a time can be transferred to B-CG and/or by means of a buffer to LINE MEMORY/GEN. or ATB-GEN etc.

Line Memory/Generator

The Line Memory/Generator is a RWM with a capacity of 128 bytes. Depending on the Pres Mode, can either character code from display area I or II be written into the "line memory" during sweep 0.

These character codes are then used to access either the ATB-GEN or the Option Character Generator (see below). The Option Character Generator is located on the TIA-A circuit board. After sweep 0 (at the rest of the sweeps), the line memory function changes and it starts to work as a "Line Generator". The "Generator" either accesses the ATB-GEN or the Option Character Generator. When CRTC addresses one display area, the Line Generator is using its "prerecorded" character code from the other display area. Because of this it is possible to display two display areas simultaneously.

Attribute Generator

The ATTRIBUTE-GENERATOR (ATB-GEN) decodes data either from display - or line-memory. The ATB-GEN is also connected to some outputs at the DIA-PIA, through which the program is to control the attribute function.

Basic-Character Generator

The BASIC-CHARACTER GENERATOR (B-CG) is a (P)ROM and intended for standard character types. The generator can be extended in steps of 1 kbyte to 4 kbytes. The size and extension of the character generator depends on the number of different characters to be displayed and the number of sweeps within the character. For example, 2 kbytes for 128 character and each character 13 sweeps high, or 4 kbytes for 256 characters and each character 13 sweeps high.

Option-Character Generator

The OPTION-CHARACTER GENERATOR (O-CG) is normally a R/W-type and is a "soft" character generator in the system. The memory circuits are such a type that they easily can be exchanged by a (P)ROM and become the "firmware" character generator.

Like B-CG O-CG is able to be extended by steps of 1 kbyte up to 4 kbyte. O-CG is normally to be a substitute to B-CG, but can also be the complement. O-CG is located at the TIA-A circuit board.

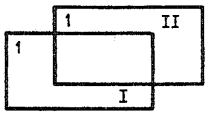
Display Memory Area

The display memory is a part of the RWM area. It uses a certain address area and the size of the area is either 4- or 8 kbyte. The size 4- or 8 kbyte, is selected through the MIC-PIA B-bit 3.

The display memory is from the presentations point of view divided into two areas I and II. Each of the areas contain 2- or 4 kbyte depending on the display memory size selection.

The presentation can be ordered to work in the following manner:

- PRES MODE 0:

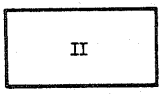


A mix of area I and II is displayed, which means that position 1 in area I and position 1 in area II are displayed at the same time. Area II is connected to the line memory and contains the attributes. The size of the display memory is either 2 x 2 kbytes or 2 x 4 kbytes. DIA-PIA B-bit 2 and 1 = 11.

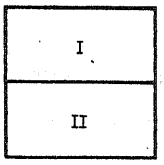
- PRES MODE 1: A mix of area I and II is displayed, but area I is connected to the line memory and contains the attributes. The size of the display memory is either 2 x 2 kbytes of 2 x 4 kbytes. DIA-PIA B-bit 2 and 1 = 10.

Note: If MPU uses ADDR.MODE 0 (see below) area I is at the lowest addresses (7000-77FF or 6000-6FFF) and area II at the highest addresses (7800-7FFF or 7000-7FFF)

- PRES MODE 2: Area II is only displayed. Character- and attribute codes are mixed. The size of the display memory is either 2- or 4 kbytes.
  DIA-PIA B-bit 2 and 1 = 01.



- PRES MODE 3: Area I and II are displayed in sequence. Character and attribute codes are mixed. The size of the memory is either 4- or 8 kbytes.
  DIA-PIA B-bit 2 and 1 = 00.



Thus: If PRES-MODE 0 or 1 is selected, the attribute character doesn't occupy any position on the screen which means that two display areas will always be presented simultaneously. This type of field function is mainly in accordance with the UTS 400 method.

If PRES-MODE 2 or 3 is selected, the attribute occupies one position on the screen and the FAC is (normally) presented as a space. This type of field function is mainly in accordance with the method used by IBM and 3500.

- The presentation addressing can be selected to "wrap around" by setting the most significant address bits as constants. This technique, where screen start can be set to an arbitrary value by letting the data string after the last address continue on the first address of the area, is advantageous at many memory editing functions as "roll" and "scroll".

  MPU (or DMA) has the possibility to address the display memory area in two different ways depending on the PRES-MODE ordered:

- ADDR.-MODE 0: Areas I and II are consecutively addressed from MPU, that is, first all positions in area I are addressed in sequence, then all positions in area II.

MIC-PIA B-bit 5 = 0.

- ADDR.-MODE 1: Areas I and II are addressed mixed, which means:

ADDRESS FROM MPU/DMAC	AREA I pos:	AREA II pos:
0	1	-
1	-	1
2	2	-
3	-	2
etc.		

MIC-PIA B-bit 5 = 1.

Selection of address mode can be useful at presentation of two areas for example when the presented characters are in area I and attributes in area II. The MPU normally works in ADDR.-MODE I, but if seeking of only attributes shall be done, the seek time is reduced if ADDR.-MODE 0 is selected.

Also when presentation of only area I is ordered both address modes can be advantageous. Normally MPU works in ADDR.-MODE 0 but if area II is defined as buffer area a fast copy to the buffer can be made if ADDR.-MODE 1 is selected at this work.

Please observe, that within the display memory area the same address mode rules are valid for DMA and MPU.

Field Attribute Characters

The Field Attribute Character (FAC) divides the display into different sections or fields.

The FAC is stored in the display memory in the same way as other characters and decides the special feature of the field together with DIA-PIA.

Three types of FAC:s are possible on DTC-A:

- Field attribute character - IBM (FAC-IBM)
- Field attribute character - UTS (FAC-UTS)
- Message Line Field Attribute Character MLFAC

In order to use FAC:s at all, DIA-PIA per reg A bit 7 must be = 1.

FAC-IBM

7 6 5 4 3 2 1 0	
1 0 - - - - - -	Code interpreted as FAC-IBM when DIA-PIA per reg B-bit 3 = 0, A bit 1 = 0. (For A and B bits see DIA-PIA A- and B reg).
1 0 - - - - 0 -	Code interpreted as FAC-IBM-APL, when DIA-PIA per reg B bit 3 = 0, A bit 1 = 1.
1 0 X X - - - X	Bits that are only treated by the software (e.g. numeric field)
1 0 - - X X - -	Bits controlling the presentation without being influenced by the DIA-PIA:
0 0	Normal light intensity (Soft: not Selector Pen (SP) detectable).
0 1	Normal light intensity (Soft: SP detectable).
1 0	High light intensity (Soft: SP detectable).
1 1	No presentation (Soft: not SP detectable).
1 0 - - - - X -	Bit that is either interpreted by the software or controls the presentation, depending on order bits in DIA PIA A.

FAC Bit 1	Bits 3 2 (DIA PIA A)	
0/1	0 0 Treated by software	
0	0 1 Doesn't influence the function	
1	0 1 Inverse presentation *)	Note! Not at APL
0	1 0 Doesn't influence the function	
1	1 0 Underline	
0	1 1 Doesn't influence the function	
1	1 1 Flashing field	

*) Inverse compared to presentation ordered by DIA PIA A-bit 0, see DIA PIA A- and B-reg.

FAC-UTS

7 6 5 4 3 2 1 0

1 - - - - - - - Code interpreted as FAC-UTS when DIA PIA B bit 3 = 1.

1 X X X X X - - Bits that are treated only by the software (e.g. numeric field).

1 - - - - - X X Bits controlling the presentation depending on order bits in DIA PIA per. reg. A:

Bits	3 2 (DIA PIA A)
0 0	- - Univac normal light intensity (high light intensity)
0 1	- - no presentation
1 0	- - Univac low light intensity (normal light intensity)
1 1	0 0 Univac normal light intensity
1 1	0 1 inverse light intensity*)
1 1	1 0 underline
1 1	1 1 flashing

*) inverse with respect to order by DIA PIA per. reg. A bit 0, see DIA-PIA A- and B-reg.

- For both FAC-IBM and FAC-UTS the following presentation functions are valid for by means of FAC:s defined fields:

  - underline begins on the position after a FAC and ends on the position before the next FAC. For underline resulting from code 5F, the underline stops at code 00, 20 or ATB, whichever occurs first. (See DIA-PIA peripheral register A, bit 5)

  - Other functions begin on the position of the FAC and end on the position before the next FAC.
    See also DIA-PIA peripheral register B bit 0.

- Message Line Field Attribute Character (MLFAC)
  This attribute has in distinction to the already described no connection to standard communication routines. It is only used as an internal attribute for the presentation logic.

MLFAC

7 6 5 4 3 2 1 0	
1 1 1 1 1 1 1 1	Start message line, DIA PIA per. reg. B bit 3 = 0 (IBM) (FAC-IBM)
0 1 1 1 1 1 1 1	Start message line, DIA PIA per. reg. B bit 3 = 1 (UTS) (FAC-UTS)

Display Adaptation Peripheral Interface Adapter (DIA-PIA)

DIA-PIA is connected to the internal bus system and is the circuit by which the program controls:

- Presentation type such as normal, inverse or inhibited presentation.
- Cursor presentation normal or with inversion of underlying character.
- Attribute functions.

DIA-PIA (F7D0)$_{16}$

DIA-PIA A-REG

7 6 5 4 3 2 1 0	
- - - - - - - x	0 = Normal video (dark background) 1 = Inverse video (light background for the whole screen)
x	0 = FAC-IBM 1 = FAC-IBM-APL/TEXT. For FAC control bits, see "Field Attribute Characters".
x x	Control bits for the FAC, see "Field Attribute Characters".
x	0 = Cursor is a covering video signal 1 = Inverted character visible "through" cursor. This is valid for dot matrix and underline.
x	0 = Code 5F (hex) produces underlined SPACE 1 = Code 5F (hex) produces underline from the following position until NUL, SPACE or FAC.
x	0 = CRTC "enable" (presentation on) 1 = CRTC "disable" (presentation blocked)
x	0 = No FACs interpreted 1 = FAC interpreted.

CRA at address $(F7D1)_{16}$
is set to

7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0

DIA-PIA B-REG $(F7D2)_{16}$

7	6	5	4	3	2	1	0	
-	-	-	-	-	-	-	x	1 = FAC valid until next FAC or end of line (5F(hex) same function) 0 = FAC valid until next FAC independent of end of line or end of page.
-	-	-	-	-	x	x	-	00 = PRES-MODE 3 01 = PRES-MODE 2 10 = PRES-MODE 1 11 = PRES-MODE 0
-	-	-	-	x	-	-	-	0 = attribute codes are interpreted as IBM (FAC-IBM) 1 = attribute codes are interpreted as UTS (FAC-UTS)
-	-	-	x	-	-	-	-	0 = video blocked 1 = video connected to CRU
-	x	x	-	-	-	-	-	00 = position box = 16 sweeps (=25 lines) 01 = position box = 12 sweeps (=33 lines) 10 = position box = 9 sweeps (=44 lines) 11 = position box = 22 sweeps (<25 lines) Note: If bits 6, 5 = 01 or 10 no field underline is generated.
x	-	-	-	-	-	-	-	= reserved

CRB at address $(F7D3)_{16}$ is set to

0	0	0	0	0	1	0	0

Note: If a FAC is visible as a space on the screen or "hidden", only depends on the PRES-MODE used.

TWO-WIRE INTERFACE ADAPTER, TIA

A general description of the TIA principles is found in the "Two-wire Communication" paragraph of the "Microcomputer" chapter. Thus the description of the TIA that follows is fairly detailed. The option character generator function of the TIA-A is described in the section "DTC-A Principles" above. The TIA is built up around the two LSI circuits; DMAC and ADLC.

- DMAC, Direct memory access controller, handles DMA transfer requests and provides memory address from an internal address counter, opening a data path between a memory cell and the transmit or receive data register of the ADLC.

- ADLC, Advanced data link controller, converts the data from parallel to serial form, for them to be modulated and transmitted on the two-wire (and vice versa for reception).

Transmission

To initiate a block transmission using DMA the program must prepare itself to answer an interrupt from the ADLC (IRQ7) by checking whether the whole block (frame) is transmitted and then switch over to reception. Also, the DMAC channel 0 address, byte count and control registers should be programmed with

1. Start address of DMA, i.e. memory address of first byte (except FLAG) to be transmitted on the two-wire connection.
2. Number of memory bytes to be transmitted.
3. Order to count up or down when addressing the memory.

4. Order to produce the read state (high level) on the R/W line when addressing the memory.

5. Transfer mode (TSC steal mode).

The Enable Tx RQ 0 bit of the DMAC priority control register is set to make DMAC react to TDSR (Transmit data service request) from ADLC.

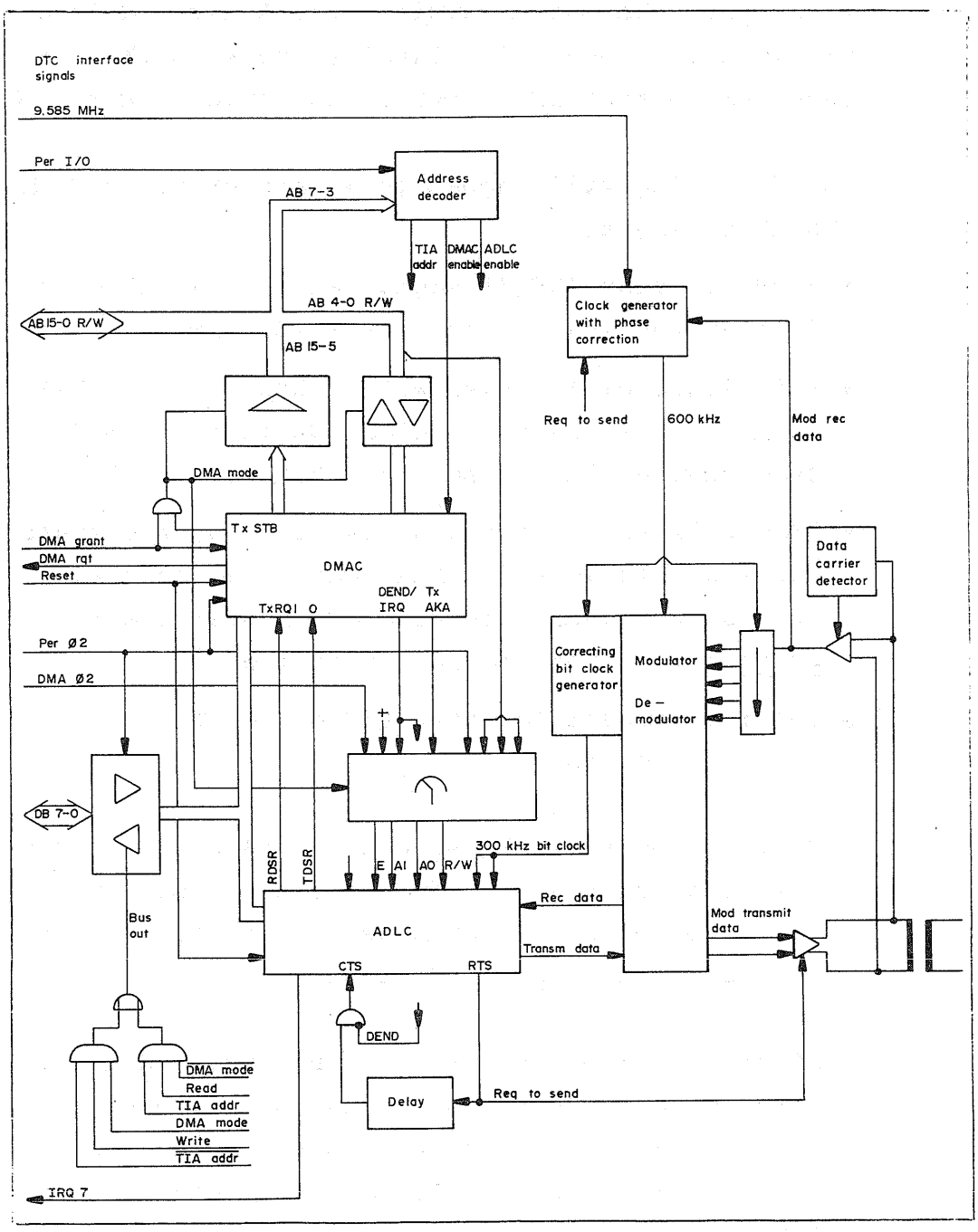


Fig. 11. Two-wire interface adapter

The ADLC must be programmed for

1. Correct word length (8 bits)
2. NRZ data format
3. Non-loop mode
4. Transmitter DMA mode etc.

The transmitting section of the ADLC is released and the RTS, Request to send, control bit is set to prepare the output amplifiers. After a delay, CTS (Clear to send) is signalled back to the ADLC, thus TDSR output from ADLC is no longer inhibited.

The following will happen as an answer to the TDSR (Tx RQ 0) signal.

- A DMA request (DRQT) signal will be sent to the display terminal controller (DTC) timing logic.
- This logic will answer with a DMA grant (DGRNT) signal, defining the cycles DMA T1 and DMA T2 (see chapter "Microcomputer" "DMA timing").
- DMAC will then produce a low level on the transfer acknowledge A (Tx AKA) output as channel 0 (transmit channel) is used. This will ensure a Write strobe for the ADLC when the
- DMA mode signal is activated by the Tx STB (Transfer strobe).

In the same time the ADLC is enabled, its transmit data register is addressed and a memory cell is addressed by the DMAC as the drivers for the address lines from DMAC are enabled.

After the direct memory access a byte is shifted out and modulated. Thus it takes a time before the transmit data register is again available. During this time the DTC is consequently undisturbed by any DMA request.

When the DMA transfer (consisting of the number of bytes defined in a DMAC register) is being completed, i.e. the last byte is being fetched from the memory, the DEND/IRQ output of ADLC is activated. This will result in

- a termination of the frame (write into the "Transmit FIFO terminate" register; ADLC adds CRCC and FLAG)
- a deactivation of the Clear to send (CTS) signal. The ADLC will react by signalling no more TDSR.

  The non-CTS signal causes an interrupt. As an answer, the program resets the RTS signal, clears status bits in ADLC and prepares DMAC and ADLC for reception.

Modulation

The modulation of the binary Transmit data from the ADLC is made in two steps. The Transmit data condition signal is generated in the ROM. It is either a high level (corresp. to Transmit data = low) or a copy of the Bit clock signal (corr. to Transmit data = high).

When this signal is fed to the modulation (JKMS) FF which is clocked by the 600 kHz JKMS clock from the clock generator, a frequency modulated signal is generated, as shown in Fig.12.

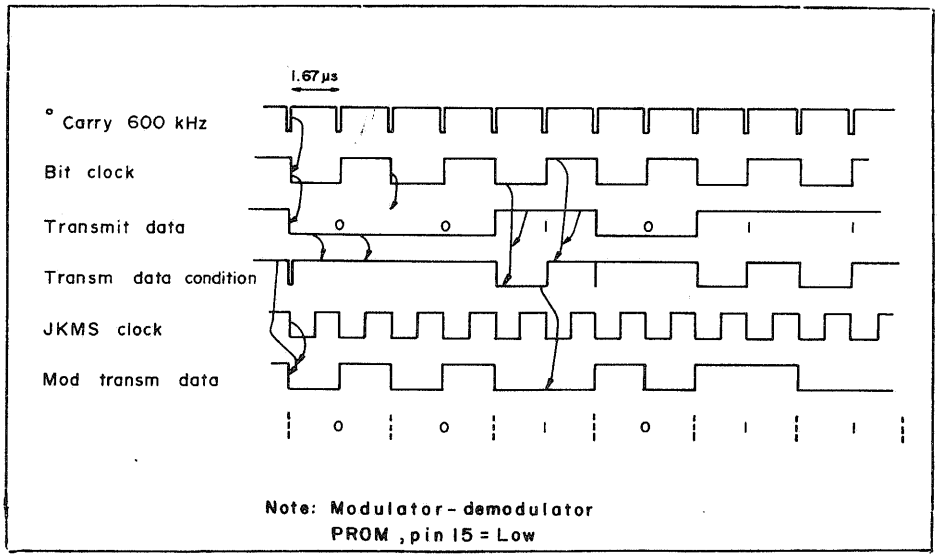


Fig. 12. Modulation

To be noted is that Mod. transmit data is fed back as Mod. rec data to the demodulator, is demodulated and fed into the ADLC. The ADLC must be properly programmed not to react to these returned data.

Reception

Preparatory programming of the DMAC and ADLC for receiving operations resemble to the one described in the Transmission paragraph. Furthermore, the program sees to it that an interrupt on level 7 (IRQ7) will be answered by program checking receiver status etc.

When bytes following the leading FLAG are received, ADLC sends RDSR (Receive data service request) to DMAC. These bytes will then be stored in the memory according to DMAC channel 1 programming. The DMA transfer is terminated as a response to an IRQ caused by error(framing error, ABORT received or receiver overrun) or by frame valid status. Then the program resets the ADLC and takes care of the received data on IRQ 0 level. Note that Request to send must of course be reset during reception.

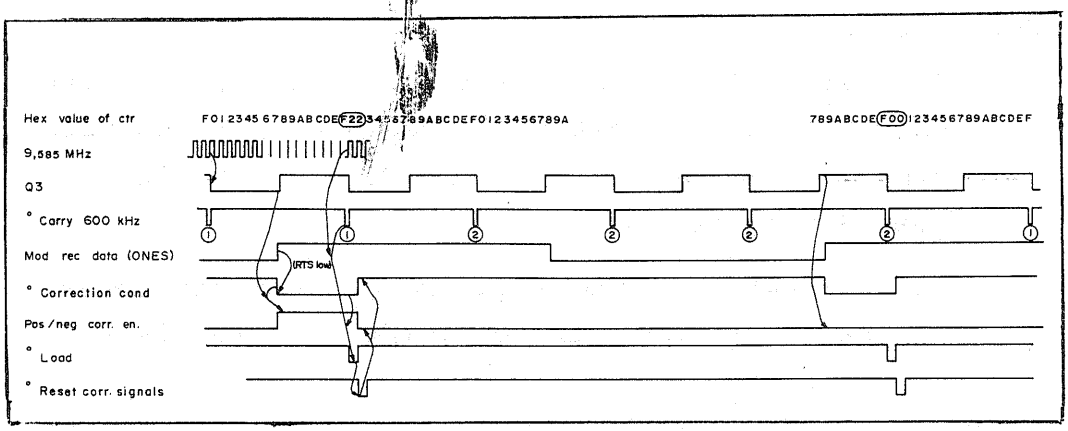


Fig. 13. Phase correction

Phase correction

The Carry 600 kHz signal is used to sample the Modulated rec. data. If there is no Request to send signal, phase correction is active, to make the Carry 600 kHz signal coincide with the centre of the shortest Modulated receive data pulses (corresponding to zeros).

The phase correction circuitry based on a divide by 16 counter. As can be seen from Fig. 13 it responds to every positive edge on the Mod. receive data line by compressing or stretching the nominal 1.67 µs (600 kHz) period. When a positive Mod. rec. data transition appears during the first "half" of the 600 kHz clock (Q3 low), the clock is accelerated, when the transition comes during Q3 high time, the clock is slowed down. Fig. 13 describes the phase correction on low frequency Mod. rec. data (logical ones). It only shows the two most correct phases ① and ② of the 600 kHz clock relative to the Mod. rec. data.

Demodulation

The Carry 600 kHz signal is used to shift Mod.rec. data into a five bit shift register. Its outputs are tied to the mod.-dem. PROM. The function of the PROM is to

1. Judge whether the sequence of ones and zeros that is present in the shift register is a valid one.
2. In that case produce a demodulated signal (Receive data).
3. Correct the 300 kHz bit clock, if needed, by 180 degrees, to produce a positive going shift-in-clock for the ADLC in the centre of the Receive data bits.

As one bit is represented by two values in the shift register (bit rate 300 kHz, shift clock rate 600 kHz), the five bits must either be divided into three bit groups as 1+2+2 values or 2+2+1 values where there is always a change of level from one bit group, equal to a data bit, to the next for correct data.

Fig. 14 shows that Rec. data indicates a zero only for valid data and 010 or 101 on A4, A3, A2. (High frequency = logical zero). Thus, both for correct logical ones and invalid data, Rec. data will indicate ones. Note that ADLC understands more than six consecutive "ones" as an error(ABORT).

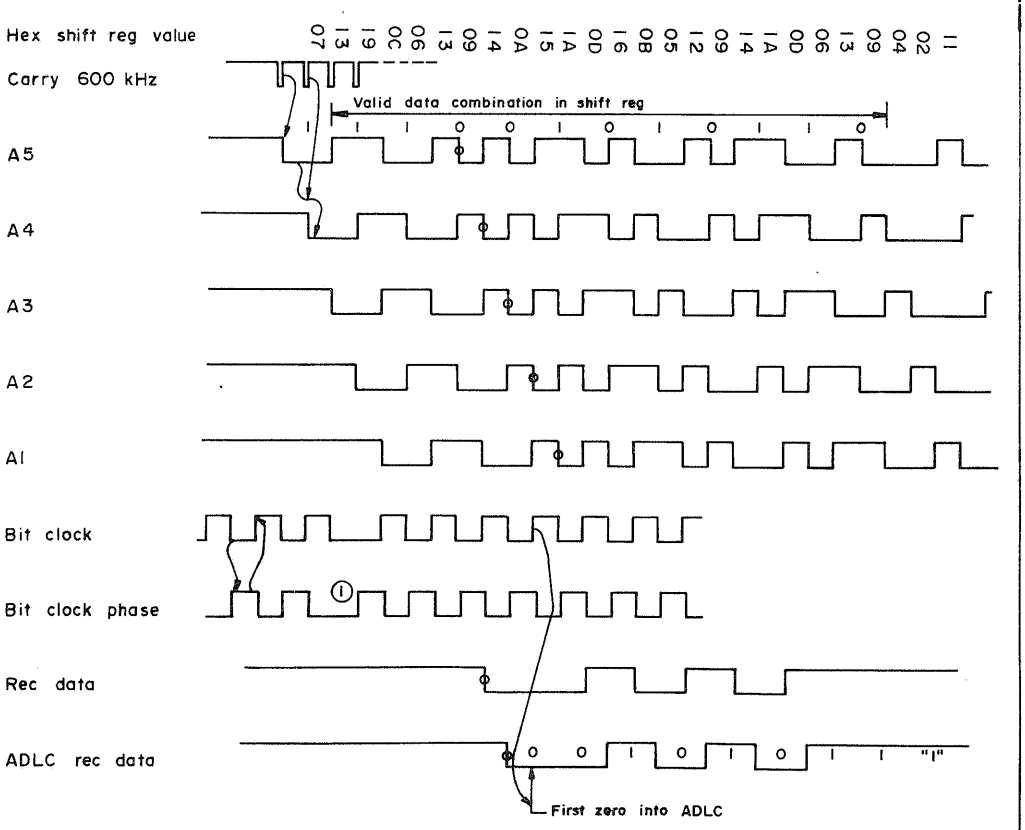
The Bit clock phase output is used to make the Bit clock positive transition coincide with the middle of an ADLC Rec. data bit. A data bit is clocked in when its first half is entering the "LS" position (A1) of the shift reg. This is shown in the timing diagram, Fig. 15.

Note that for the input combinations of Fig. 14 where Bit clock phase values are left out, this

output (Pin 12) merely presents the inverted value of Bit clock (Pin 5). Fig. 15 shows an example of some valid data bits being demodulated and Bit clock being corrected. Note that when no data are received, all ones (IDLES) are fed to the ADLC. Hexadecimal shift reg. values according to Fig.14 are shown.

Inputs								Outputs	
Valid	In-valid	Pin 2 A5	3 A4	4 A3	7 A2	6 A1	Hex	Pin 9 Rec.data	Pin 12 Bit clock phase
	X	0	0	0	0	0	0 0	1	
	X	0	0	0	0	1	0 1	1	
	X	0	0	0	1	0	0 2	1	
	X	0	0	0	1	1	0 3	1	
	X	0	0	1	0	0	0 4	1	
X		0	0.	1	0.	1	0 5	0	
X		0	0.	1	1.	0	0 6	1	1
	X	0	0	1	1	1	0 7	1	
	X	0	1	0	0	0	0 8	1	
X		0	1.	0	0.	1	0 9	1	1
X		0.	1.	0.	1.	0	0 A	0	
X		0.	1	0.	1	1	0 B	0	
X		0.	1	1.	0	0	0 C	1	0
X		0.	1	1.	0	1	0 D	1	0
	X	0	1	1	1	0	0 E	1	
	X	0	1	1	1	1	0 F	1	
	X	1	0	0	0	0	1 0	1	
	X	1	0	0	0	1	1	1	
X		1.	0	0.	1	0	2	1	0
X		1.	0	0.	1	1	3	1	0
X		1.	0	1.	0	0	4	0	
X		1.	0.	1.	0.	1	5	0	
X		1	0.	1	1.	0	6	1	1
	X	1	0	1	1	1	7	1	
	X	1	1	0	0	0	8	1	
X		1	1.	0	0.	1	9	1	1
X		1	1.	0	1.	0	A	0	
	X	1	1	0	1	1	B	1	
	X	1	1	1	0	0	C	1	
	X	1	1	1	0	1	D	1	
	X	1	1	1	1	0	E	1	
	X	1	1	1	1	1	1 F	1	

Fig. 14. Demodulation and Bit clock correction ROM



Notes (1) shows a bit clock phase correction (180 degrees)
The small circles mark the times when the first incoming zero may be detected on different lines.
PROM Pin 15 is low, thus Transmit data condition (Pin 10) is only affected by Transmit data (Pin 1), as described in Fig. 12.

Fig. 15. Demodulation. Timing diagram

TIA-S

The TIA or TIA-A board is in some applications substituted by a TIA-S board in order to enable connection of the terminal to an SS3 bus. The only functional difference is implemented by a dip-switch. (The electrical characteristics of the bus interface are also changed)

The IPL program can at start up read the terminals own poll/select address from the 8-bit dip-switch. The dip-switch is allocated to memory address $F740_{(16)}$ (to $F747_{(16)}$).

CATHODE RAY TUBE UNIT, CRU

Brief Outline

This description is valid for CRU E34112 0000, developed by Datasaab. The cathode ray tube unit generates pictures according to a raster scan method, i.e. an electron beam makes a number of consecutive horizontal sweeps over the screen.

The beam is modulated by the Video information, thus dots and horizontal lines forming the picture are generated on the phosphor coated screen. The CRU is supplied with +24 V from the display terminal power supply (DPS). Its interface towards the DTC (Display terminal controller) consists of four TTL level signals namely

- Video
- High intensity, active when text etc. with higher brightness is displayed.
- H-sync. Horizontal synchronization signal. This pulse initiates the retrace and the sweep from left to right of the beam over the screen (Horizontal deflection)
- V-sync. Vertical synchronization signal, initiating the vertical sweep of the beam (Vertical deflection) every 20 ms.

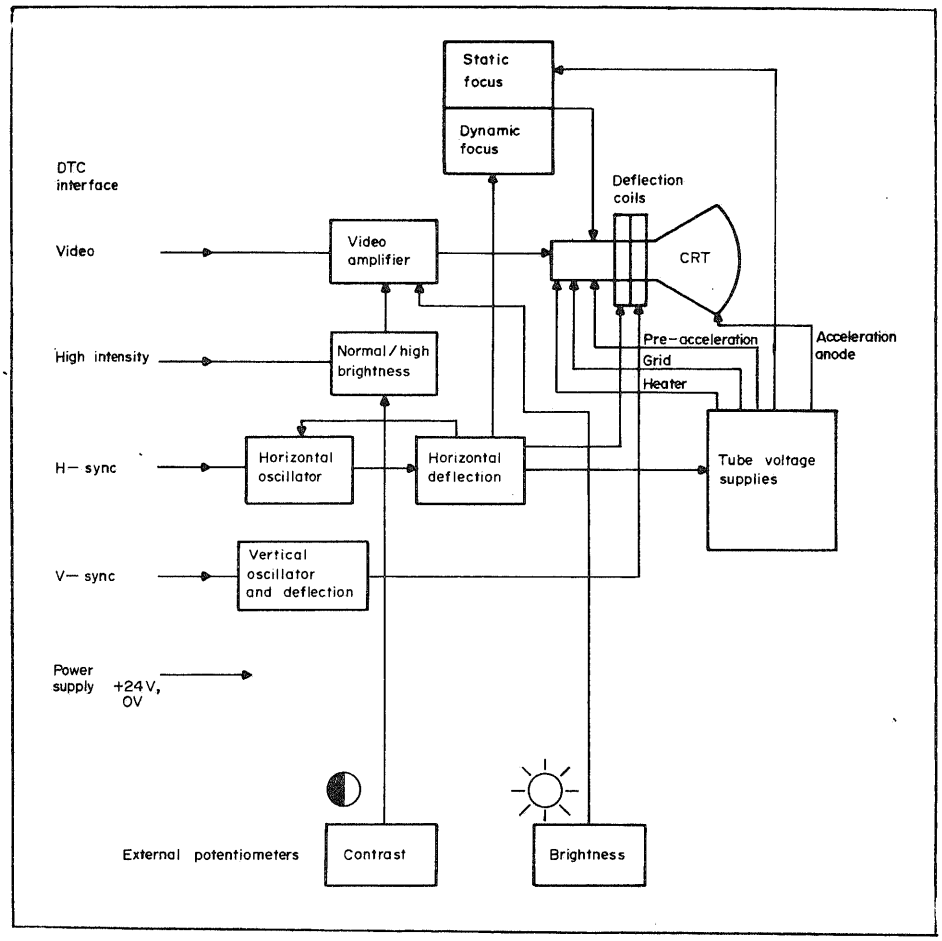


Fig. 16. CRU block diagram

The CRU consists of the picture tube (CRT) with deflection coils and a printed circuit board, CRB (Cathode ray tube unit board), separated into two parts that are permanently interconnected by a band cable.

- The CRT is a high resolution 110° tube. It is fed with high voltage for the anodes accelerating the electron beam, voltages for the electrode focusing the beam and for the control grid, current for heating the cathode and video current for the cathode. Furthermore signals affect the deflection coils that govern the vertical and horizontal deflection of the beam.

- The main part of CRB contains circuitry for vertical and horizontal deflection (oscillators,

output stages), circuitry for focusing of the beam and for generation of tube voltage supplies.

- The small part of the CRB is mounted behind the deflection unit on the CRT socket. It contains video amplifier, brightness control circuitry and circuits for flashover protection.

Detailed Description

See circuit diagram E34111 2000.
For trimming instructions, see "Installation and Maintenance" manual.

Deflection Circuits

The horizontal deflection is governed by IC 111, which contains an oscillator free-running at 21.3 kHZ and a phase locked loop for provision of a horizontal deflection signal in phase with the incoming H-sync signal. Furthermore, the horizontal deflection output stage generates the tube voltages by help of the transformer T153, e.g. the high voltage for the acceleration anode.

The vertical deflection is governed by IC 131. It contains an oscillator that should be free-running at a lower frequency (47 Hz) than the regeneration frequency of the incoming V-sync signal (50 Hz). It also contains an output stage providing the vertical deflection current.

Focusing

The voltage for the focus grid has two components; a static one, adjusted by help of a potentiometer, and a dynamic one, derived from the horizontal deflection signal. The parabolic dynamic focusing signal is needed as the path of the beam, from cathode to screen, is longer when directed towards the left or right edge of the screen than when

directed towards the centre.

Brightness Control

The brightness control circuitry and video amplifier located on the small part of CRB produces an amplified video signal for the cathode. The amplitude of this signal depends on the setting of the external Brightness potentiometer and, for fields with high intensity, on the setting of the external Contrast potentiometer. Furthermore, there is an internal potentiometer, BLACK LEVEL, that governs the amplitude of the beam by affecting the control grid.

Protective Circuitry

To prevent beam current on the acceleration anode from becoming too high, there is circuitry on the main part of CRB that will limit it by decreasing the voltage on the control grid.

The small part of CRB contains spark gaps that will direct flash-overs between the extra high tension on the acceleration anode and the other electrodes the shortest way to ground. Furthermore there are resistors in series with the electrodes that will diminish the flash-over current.

Adjustments

The deflection unit on the tube neck holds a pair of adjustable magnets for centering of the picture and a number of adjustable magnets for raster correction (correction of pin-cushion-formed picture). On top the the main part of CRB there are a number of potentiometers which are used for initial trimming of the CRU. There are also three more adjustment points lower on the CRB.

The adjustment points are:

- H: HOLD — for setting of the internal horizontal (sweep) frequency to 21.3 kHz.
- H: PHASE — for setting the phase of the horizontal deflection signal to fit the incoming video information.
- H: LIN* — Adjustable linarity control unit. For setting the linearity of the horizontal deflection signal.
- WIDTH* — Adjustable coil for setting the picture width.
- V: HOLD — for setting the free-running frequency of the vertical oscillator to 47 Hz.
- V: LIN — for setting the linearity of the vertical deflection (equal spacing between raster sweeps).
- HEIGHT — for setting the height of the picture.
- BLACK LEVEL — for initial adjustment of the brightness level.
- FOCUS — for focusing of the beam.
- J 101 VOLTAGE — to set the stabilized horizontal output supply voltage, measured between J 100 and J 101, to 21.0 V.

* accessed from solder side of PCB.

Display Units, DU 4111 and DU 4112

Contents

General

Display Units DU 4111 and DU 4112 are CRT (cathode ray tube) displays which can be used in a variety of configurations.

The Microcomputer chapter ought to be understood before this chapter is studied. A good orientation on the Communication chapter is also recommended; especially the Hardware part under Internal Communication via Two-wire. This chapter will in principle only treat circumstances not discussed in or solved in other ways than said in the two chapters mentioned above.

The DU 4111 is a one-colour-display and the DU 4112 is a four-colour-display (red, green, blue and white). They are similar in all other respects and are therefore both treated in this chapter. The expression "display unit" will refer to any one of them. Minor differences will be pointed out in their context and major distinctions assigned to one heading for DU 4111 and another for DU 4112.

Functions

The display unit can communicate with a host computer via a communication processor (cluster configuration). It can communicate with two host computers by means of two communication processors. The display unit can also communicate with flexible disk units. These communications are carried out via a two-wire or a coaxial cable. The display unit is also capable of communicating via a Standard serial interface of ß-type (SS3 bus). If an asynchronous communication adapter D is used it can communicate with a type of modem connection for personal computer applications.

Facilities are – or can be – provided for connections of a keyboard unit (KBU), a keyboard expansion unit (KXU), a magnetic identification device (MID) and a printer unit (PU).

The main tasks of the display unit are:

- Presentation of visual information on a screen. This information may be text entered via the keyboard, messages from a host computer or the operating system, data from a library on a flexible disk etc.
- Communication with host computers and other system units like a keyboard, printers, flexible disk units and communication processors.
- Editing and processing of data, e.g. documents defined by Alfaword, code conversions etc.

Subunits

The main parts of the display unit are:

- DBM-A (DU 4111) or DBM-B (DU 4112), display unit basic mechanics assembly, i.e. cabinet with internal racks but exclusive of the parts of the cathode ray tube unit and the power supply (see below).
- DPS, display power supply including display power board, DPB. See chapter Power Supplies.
- DTC-B (DU 4111) or DTC-C (DU 4112), display terminal controller containing a microcomputer, 64 kbytes of basic read/write memory, presentation logic, two-wire (or coaxial cable) and keyboard interfaces.
- CRU, cathode ray tube unit with circuitry for control of the beam.
- ICBD, interconnection board with connectors for keyboard, two-wire, coaxial cable and a DIP-switch for address setting and connecting or disconnecting of two-wire or coaxial cable terminating resistors.

One of the following optional boards can be mounted:

- ACA-B, asynchronous communication adapter which provides a serial data interface for printers. See chapter Asynchronous Communication Adapter.
- ACA-D, asynchronous communication adapter which provides a serial data interface for printers and a type of modem interface (the latter for personal computer applications).

Mechanics

The mechanical build-up of the display unit is shown in Fig. 1. The main parts of the display unit, from the mechanical aspect, are:

- DBM-A (DU 4111) or DBM-B (DU 4112), display unit basic mechanics assembly with connectors for mains and printer cables and control devices (Mains switch, Contrast control and Brightness control in DU 4111, Mains switch and Brightness control in DU 4112). Detailed information on the location of connectors and control devices etc. can be found in the Installation and Maintenance Manual. A fan, CTF, is included in DBM-B, i.e. in DU 4112.
- DPS, display terminal power supply.
- Housing (included in DBM-A or DBM-B) for DTC-B or DTC-C and one optional board.
- CRU, cathode ray tube unit including tube, mechanical supports and circuitry.

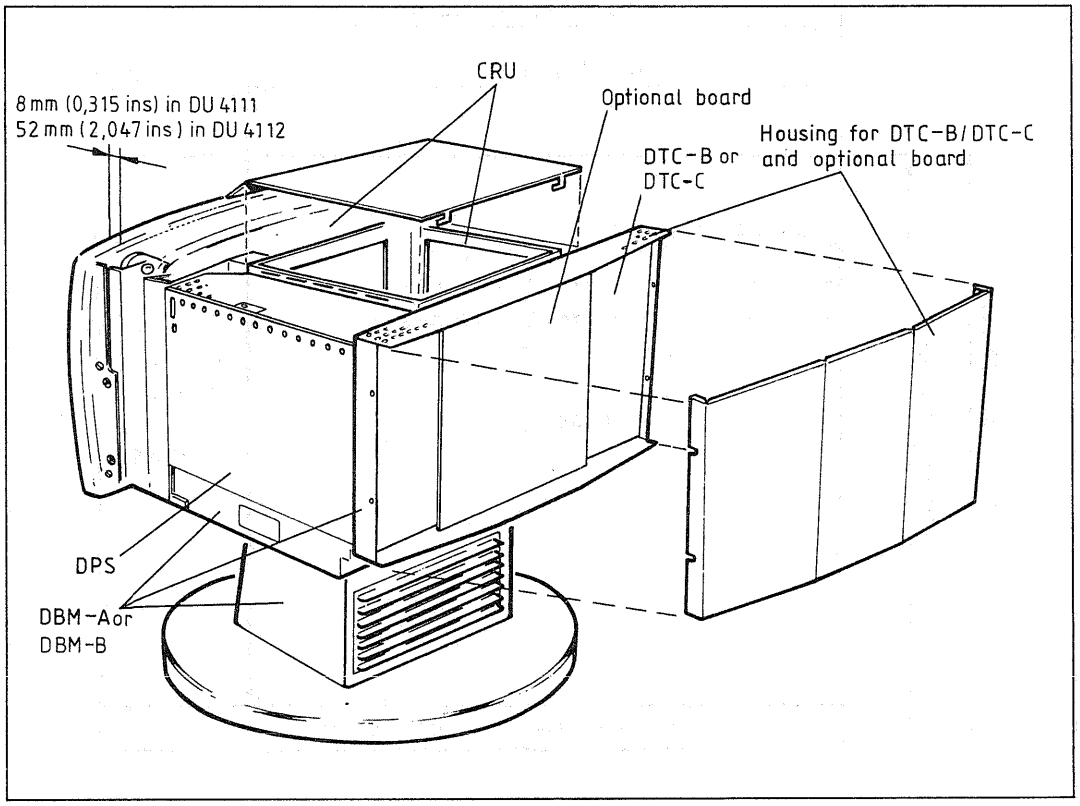


Fig. 1. Mechanical layout of DU 4111 and DU 4112

Brief Outline

Presentation

Screen Scanning

Presentation of characters on the screen is made by a raster scan method, i.e. an electron beam (the "beam" is actually composed of up to three different beams in DU 4112, see Cathode Ray Tube Unit, CRU in DU 4112) makes a number of consecutive horizontal sweeps over the screen. See Fig. 2. The beam is modulated by video information, thus generating dots that build up characters, a cursor, underlines etc.

The regeneration rate is 50 Hz, i.e. the electron beam builds up a whole picture on the screen every 20 milliseconds.

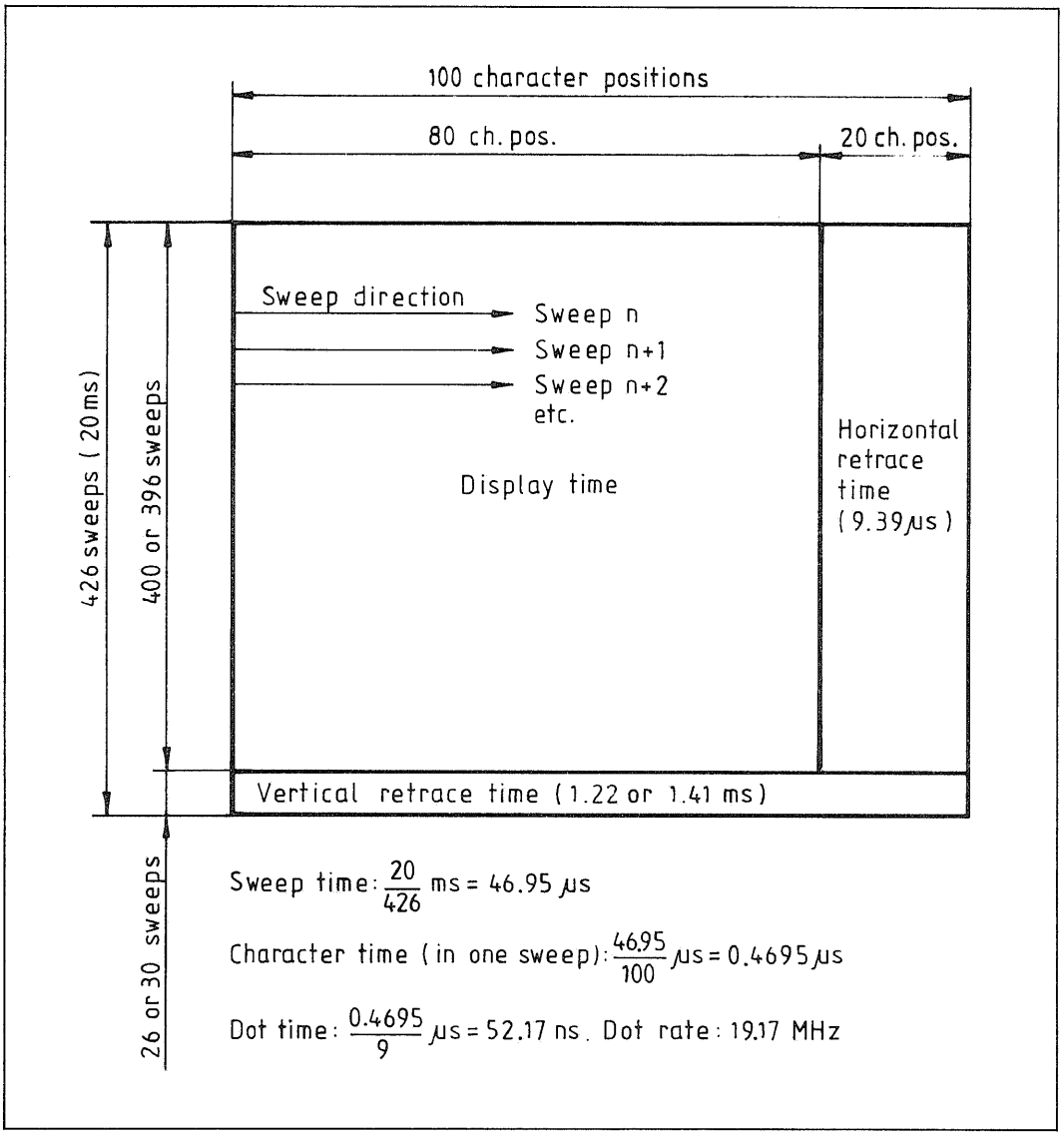


Fig. 2. Screen scanning times

Depending on software the number of text lines can be 17, 25, 33 or 44, including message line. The 25 and 33 lines formats are only used in DU 4112.

The total number of sweeps is 426. Depending on screen format, 400 (up to 25 text lines – including message line) or 396 sweeps (33 or 44 text lines – including message line) are used for presentation, spacing sweeps included. Thus, 26 or 30 sweep times are used for vertical retrace.

Each sweep is divided into 100 character positions out of which up to 80 (40, 56, 64 or 80) may be used for presentation. The remaining position times are used for horizontal retrace. Note that several sweeps are needed to build up the height of a character.

Each character position is divided into nine dot positions or columns (0 to 8). Seven columns (1 to 7) are normally used for character presentation.

The dot rate is 19.17 MHz (see Fig. 2), i.e. equal to the frequency of the basic crystal clock and 18 times the system clock rate (∅1, 1.065 MHz). Thus, in a specific sweep, the dots of two characters are presented during one system clock period.

(Horizontal lines are actually not, as vertical lines, formed by discrete dots but by continuous strokes of the electron beam.)

Presentation Geometry and Character Layout

The effective screen size of DU 4111 is 180 by 258 mm (7.09 × 10.16 ins). The horizontal distance between two adjacent dot positions is ≈ 0.36 mm. The vertical distance between two adjacent dot positions (sweeps) is ≈ 0.45 mm.

Thus, the ratio $\frac{\text{vertical dot spacing}}{\text{horizontal dot spacing}} \approx 1.25$

The effective screen size of DU 4112 is 170 by 240 mm (6.69 × 9.45 ins).

The spacing ratio is: $\frac{0.43\ \text{mm}}{0.33\ \text{mm}} \approx 1.3$

The number of vertical dot positions (sweeps) per text line is programmable and depends on the number of text lines on the screen:

Number of text lines[1)]	17	25	33	44
Number of sweeps/text line	22	16	12	9
Number of used sweeps/text line	13	13	10	8

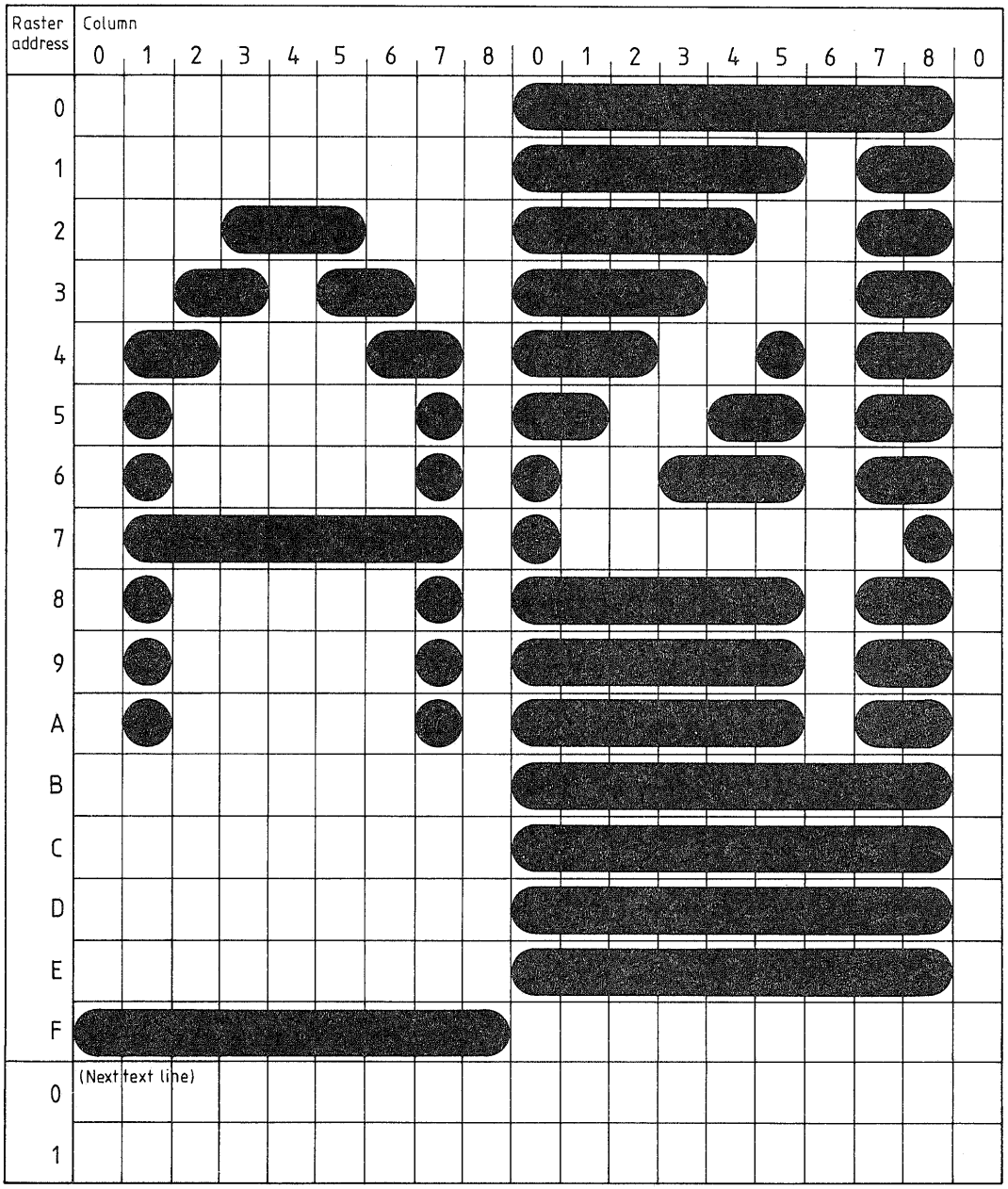
[1)] Including message line

Each sweep in a text line is identified by a raster address. Each text line starts at raster address 0 and ends at $15_{(16)}$, $F_{(16)}$, $B_{(16)}$ or $8_{(16)}$. There are thus four possible character cells: 9 (horizontal dot positions) × 22

(vertical dot positions), 9 × 16, 9 × 12 and 9 × 9. Raster address 0 is normally not used for presentation and raster addresses $>F_{(16)}$ cannot be used for presentation. The raster address is modified for the message line in order to obtain an "overline" instead of an underline of the message line.

Character layouts are shown in Fig. 3 and Appendix 4. Fig. 4 shows a 7 × 13 dot matrix (dots used for characters) in a 9 × 16 character cell (total number of dot positions).

The dot patterns of characters are generated in a character generator.



Note: Inversion of "4" caused by maximum height cursor with Cuform=1. Field underline of "A" and "4". Valid for formats with 25 textlines

Fig. 3. Character, underline and cursor layout example

Raster address	Column 0	1	2	3	4	5	6	7	8	Modified raster address at message line
0										"Over-line"
1										0
2										1
3										2
4										3
5										4
6										5
7										6
8										7
9										8
A										9
B										A
C										B
D										C
E										D
F										E

Appendix 3 version of the above figure:

D-bus	6[1)]	6	5	4	3	2	1	0	0[1)]
Col.	0	1	2	3	4	5	6	7	8
0									
1									
2									
3									
4									
5									
6									
7									
8									
9									
A									
B									
C									
D									
E									
F									

[1)] If bit 7 = 1

Fig. 4. 7 × 13 dot matrix in 9 × 16 character cell

Principles of Character Generation

Codes (8 bits) for characters to be displayed (as well as for field attribute characters, FACs, see Attributes) are stored in a part of the read/write memory that is defined as the display memory area. Note that the display memory code for a certain character may differ from the line code (used at communication with a host computer) and the code received from the keyboard for the same character.

The presentation is supervised by a cathode ray tube controller (CRTC, an LSI circuit) which addresses two display memory cells at the same time in one system clock cycle. One cell is allocated in a 64 kbytes memory block and the other in a 16 kbytes memory block. See Fig. 5. Thus, two character codes are fed from the memory. They are latched and then in proper order fed to a character generator for half a system clock cycle time each.

The character generator is a memory (2 × 2048 bytes RAM) that produces dot values when addressed by the character code and the raster address originating from the cathode ray tube controller. The character code tells which character and the raster address tells which sweep in the character that is to be displayed. The character generator can contain up to 256 character patterns and is loaded together with the operating system. (The MPU addressing, see Fig. 5, is then activated, the raster addressing and the outputs of latch C are taken to high impedance state and the data bus buffer is enabled.)

The output from the character generator is loaded into a video register and shifted out as video information to the cathode ray tube unit at a rate of 19.17 MHz.

The video register output is in DU 4112 fed to the cathode ray tube unit as Red, Green or/and Blue video information as shown in Fig. 6. If all three colours are activated the resulting colour is white.

Example of Character Generation

Assume that "A" and "4" are just displayed and the character codes have been entered for the fifth time into the A and B latches. See Figs 4 and 5. Thus, the binary value of the raster address is $00100_{(2)}$. First the 8-bit code for "A" ($41_{(16)}$) is applied to the character generator. The outputs of the character generator will present 1001 $1100_{(2)}$ where the zeroes correspond to the four dots of that sweep. (Due to inversion in the data bus buffer the character generator will be loaded with 1001 $1100_{(2)}$ when the external data bus contains 0110 $0011_{(2)}$ or $63_{(16)}$. Thus a 1 on the external data bus [and on the diskette] corresponds to a lit dot.) The video register will be loaded with nine bits where the bits for column 0 and 8 are functions of bits 0, 6 and 7. (Bit 7 high [low on the external data bus] will keep column 0 and 8 turned off. See Fig. 5.)

While the dots of the "A" are shifted out, the character generator will produce 1110 $0101_{(2)}$ (corresponding to 0001 $1010_{(2)}$ or $1A_{(16)}$ on the external data bus) i.e. the dots of the fifth sweep of the number "4". Next, the two 8-bit codes for the following two characters of the textline are addressed and entered into the A and B latches, etc.

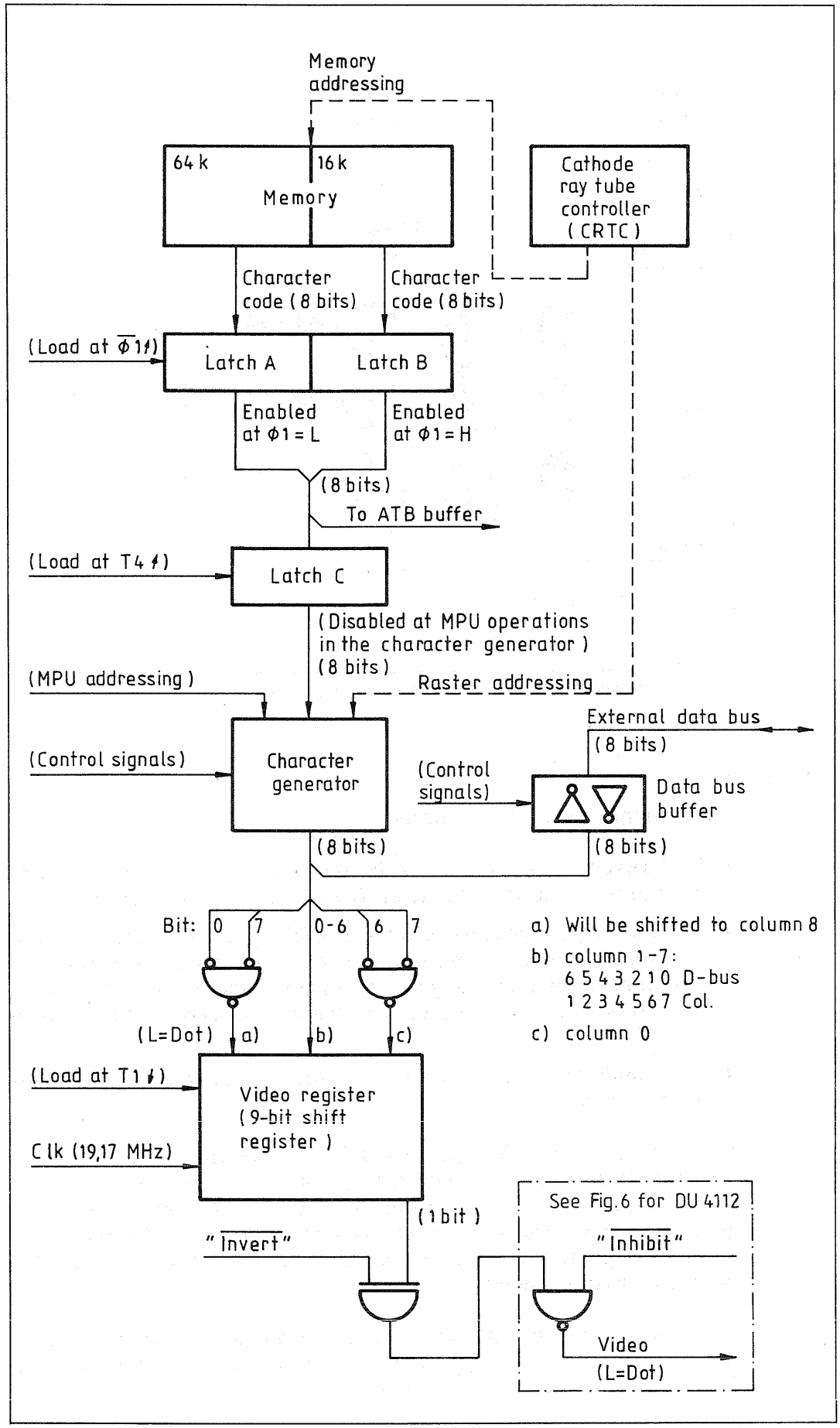


Fig. 5. Principles of character generation

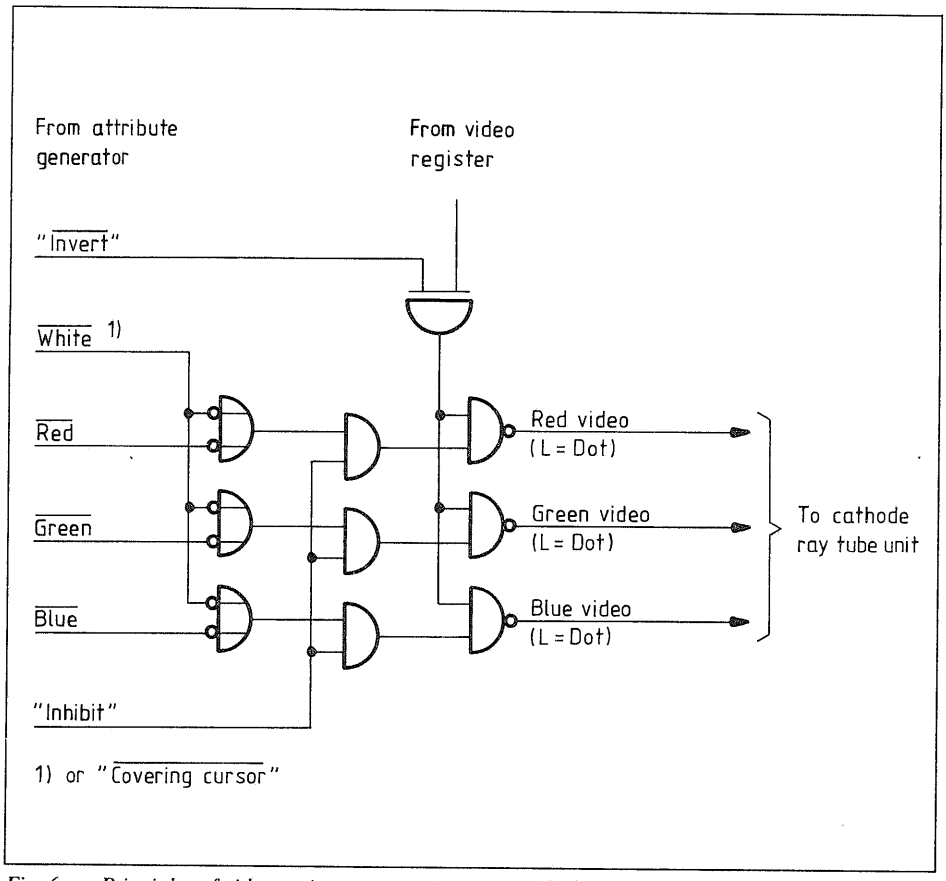


Fig. 6. Principles of video register output circuitry in DU 4112

Underlining

Underlining exists in three forms:

- Underlined space, generated (by the code $5F_{(16)}$) by the character generator as a normal character. (Columns 1 – 7 are underlined.)
- Underlined space, generated (by the code $5F_{(16)}$) by an attribute generator. (Columns 0 – 8 are underlined.) The character generator is in this case often programmed to generate a space in the position of the $5F_{(16)}$ code. If programmed to generate an underlined space both lines might be visible.
- Field underline, generated by the attribute generator. (Columns 0 – 8 are underlined.)

Sweep $F_{(16)}$ of a text line (not valid for the message line) is used for underline from the attribute generator (or spacing). The preceding sweep is only used for spacing. See Fig. 4. Underlining from the attribute generator is not possible for 33 and 44 text line formats since the sweep $F_{(16)}$ is not included then. Neither is underlined space from the character generator recommended.

Underlining from the attribute generator is not possible at the message line. An "overline" (the uppermost sweep) separating the message line from the rest of the screen is produced instead.

The underline is of the same colour as the text in DU 4112.

Cursor

The cursor is generated separately. Its height and vertical location in the character cell is programmable. The cursor can also be blanked, steady or blinking at 1.6 or 3.1 Hz. Two types of cursors may be selected by the program:

Cuform = 0: Width 9 dots. A character at the position of a steady cursor is not visible. The cursor is always white in DU 4112.

Cuform = 1: Width 9 dots. A character at the cursor position is always visible due to text inversion (i.e. normally bright character dots are dark) at the cursor position. The cursor is normally programmed to fill a whole box and is of the same colour as the field in DU 4112.

Attributes

Colour Presentation of DU 4112

The colour presentation of DU 4112 in IBM emulations is summarized here. (If a DU 4112 is used in an UTS emulation, normal light intensity fields [see Dynamic Attributes] are green, low light intensity fields are white – with normal intensity – and the message line is blue. In Alfaword applications the presentation will work as in IBM monochrome mode. DU 4112 is not suitable for VT 100 applications.)

The DU 4112 can work in base colour mode or monochrome mode. The mode can be selected by a base colour switch key, BCS-key, on the keyboard.

In base colour mode a field (see Dynamic Attributes) can be presented in either red, green, blue or white colour. In monochrome mode a field can be presented in either green or white colour. The message line is blue in both modes.

Light inversion (normally bright dots are dark and dark dots bright) will not change the colour of a field, e.g. a red R on dark background will at inversion be a dark R on red background.

Dynamic Attributes

The 8-bit words of the display memory may be character codes or field attribute character (FAC) codes. Underlined space ($5F_{(16)}$), Nul ($00_{(16)}$) and Space ($20_{(16)}$) may sometimes function in a similar way as the FACs. The FACs are used together with static control bits to define various properties, other than the text pattern, of different fields (a number of character positions) on the screen. Some of the FAC information is taken care of by the software and some by the hardware.

Note that one FAC may result in more than one new attribute of the associated field. The field attributes that may be caused by the field attribute characters hardware control are:

- Field underline
- Flashing field (1 Hz). (The cursor has an independent blink function.)
- Inverted field (referring to bright or dark dots).
- Extinguished field.
- Light intensity (two levels). Only in DU 4111.
- Field colour. Only in DU 4112.
- Message line (denoted by an MLFAC; no other attributes are permitted in the message line unless those interpreted by the software. This attribute has in distinction to the others no connection to standard communication routines. It is only used as in internal attribute for the presentation logic).

Generally (underline is sometimes an exception) the function of a FAC starts at the position of the FAC and ends just before the next FAC or stop at end of line or stop at end of last line (but the message line). (The "stops" can be programmed by means of the display adapter peripheral interface adapter. See under that heading in the Detailed Description.)

The logic can be programmed to generate field underline at the code for Underlined space ($5F_{(16)}$). The underline will then start in the position after the $5F_{(16)}$ code and end just before Nul ($00_{(16)}$), Space ($20_{(16)}$) or the same condition as an ordinary FAC (see above).

Static Attributes

Apart from the control registers of the cathode ray tube controller, defining the display area, vertical cursor format and cursor blink etc. there are two 8-bit registers of a display adapter peripheral interface adapter, DIA PIA, that either directly control some video attributes or define how to interpret the FACs.

Note that the DIA PIA definitions are static, i.e. cannot be made for specific fields on the screen as there is no simple connection between display timing and program but by the vertical synchronization signal.

The DIA PIA functions are specified in the Detailed Description and summarized below:

- Video control. The video signal can be blocked and the cathode ray tube controller (kept) reset. Either action will stop presentation. The whole screen can be inverted, i.e. all normally bright dots are dark against a bright background. Inverted fields at inverted screen will result in not inverted fields.
- Cursor control. Cursor format 0 or 1 can be selected (see Cursor).
- FAC interpretation control. FAC interpretation can be enabled or disabled, FAC interpretations defined, functions of $5F_{(16)}$, $00_{(16)}$ and $20_{(16)}$ defined etc.
- Stop at end of line or end of last line (but the message line) control. The functions of FACs and field underlines by code $5F_{(16)}$ can be ended by programmable stops at end of line or end of last line (but the message line).

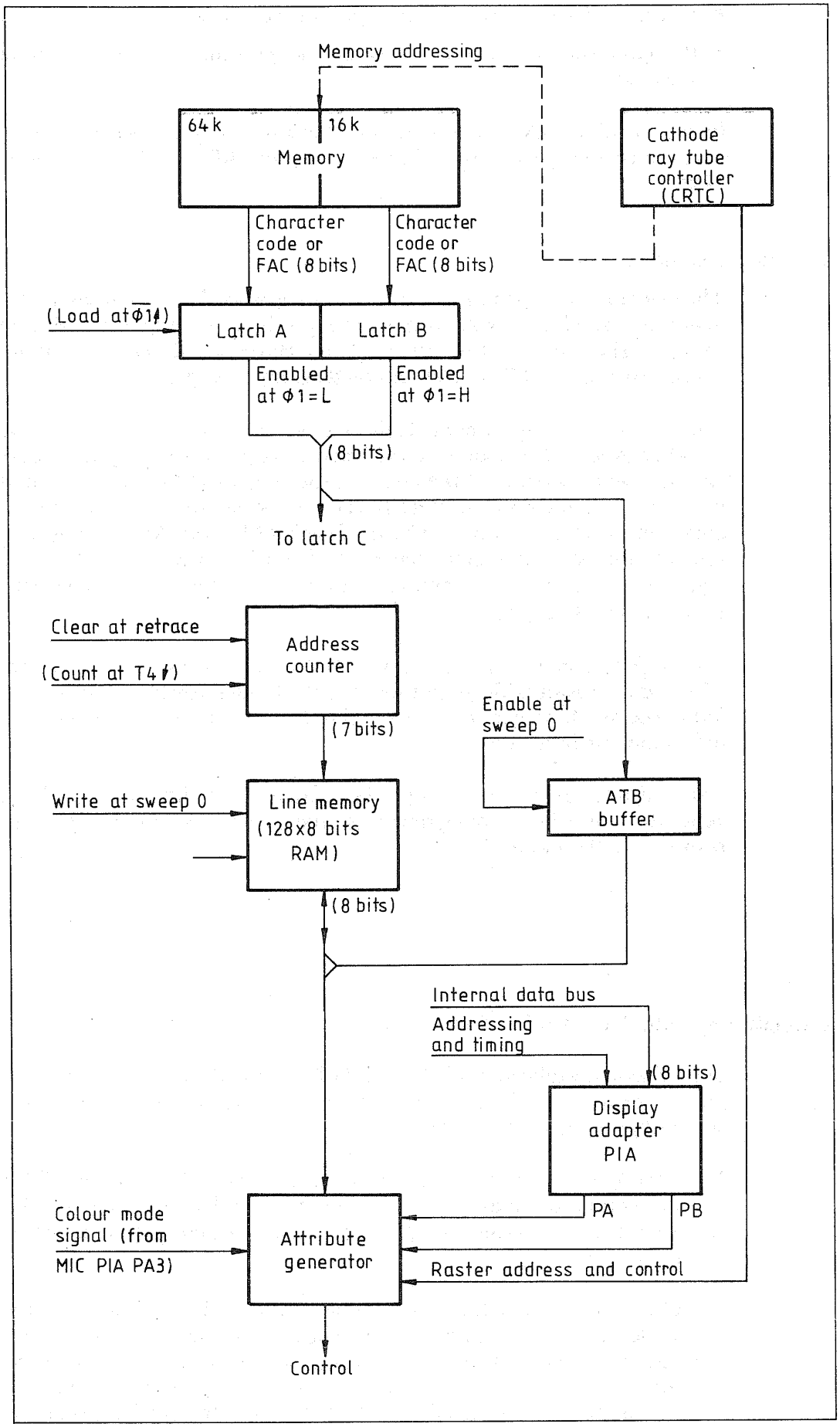


Fig. 7. Principles of attribute generation

- Sweeps per line control. Sweeps per line can be set to 9, 12, 16 or 22.
- Presentation mode control. One of four presentation modes can be selected.

In DU 4112 an additional static attribute is set by the software in a microcomputer peripheral interface adapter, MIC PIA, i.e. colour mode.

Attribute Generation

The principles of attribute generation are shown in Fig. 7. The character generator (see Fig. 5) will normally not produce any bright dots during sweep 0. The contents of the latches A and B will be fed to a line memory via an attribute (ATB) buffer, enabled during sweep 0.

The line memory is in write mode during sweep 0 and in read mode during all other sweeps. The address to the line memory is supplied by an address counter, which is reset at the end of each sweep and then incremented at each character position time in the following sweep. All character codes and FACs (or only FACs, see Display Memory Addressing) of a line will thus be written in the line memory during sweep 0 and in proper order fed from the line memory at each one of the following sweeps until the next sweep 0.

The outputs from the line memory as well as most outputs from the DIA PIA and the raster address and some control signals from the CRTC are fed to the attribute generator. In DU 4112 a colour mode signal is also fed to the attribute generator.

The attribute generator consists of latching and decoding circuits, which generate a set of control signals (e.g. the "Invert" signal in Fig. 5) in response to the inputs.

Cathode Ray Tube Unit, CRU, of DU 4111

The cathode ray tube unit (CRU) of DU 4111 generates pictures according to a raster scan method, i.e. an electron beam makes a number of consecutive horizontal sweeps over a screen.

The CRU (see Fig. 8) consists of mechanics, a cathode ray tube (CRT) with deflection coils and a printed circuit board, CRB (cathode ray tube unit board), separated into two parts that are permanently interconnected by a band cable.

The CRT is a high resolution 110° tube. It is fed with high voltages for the anodes accelerating of the electron beam, voltages for the electrode focusing the beam and for the control grid, current for heating of the cathode and video current for the cathode. It is furthermore fed with signals that affect the deflection coils that govern the vertical and horizontal deflection of the beam.

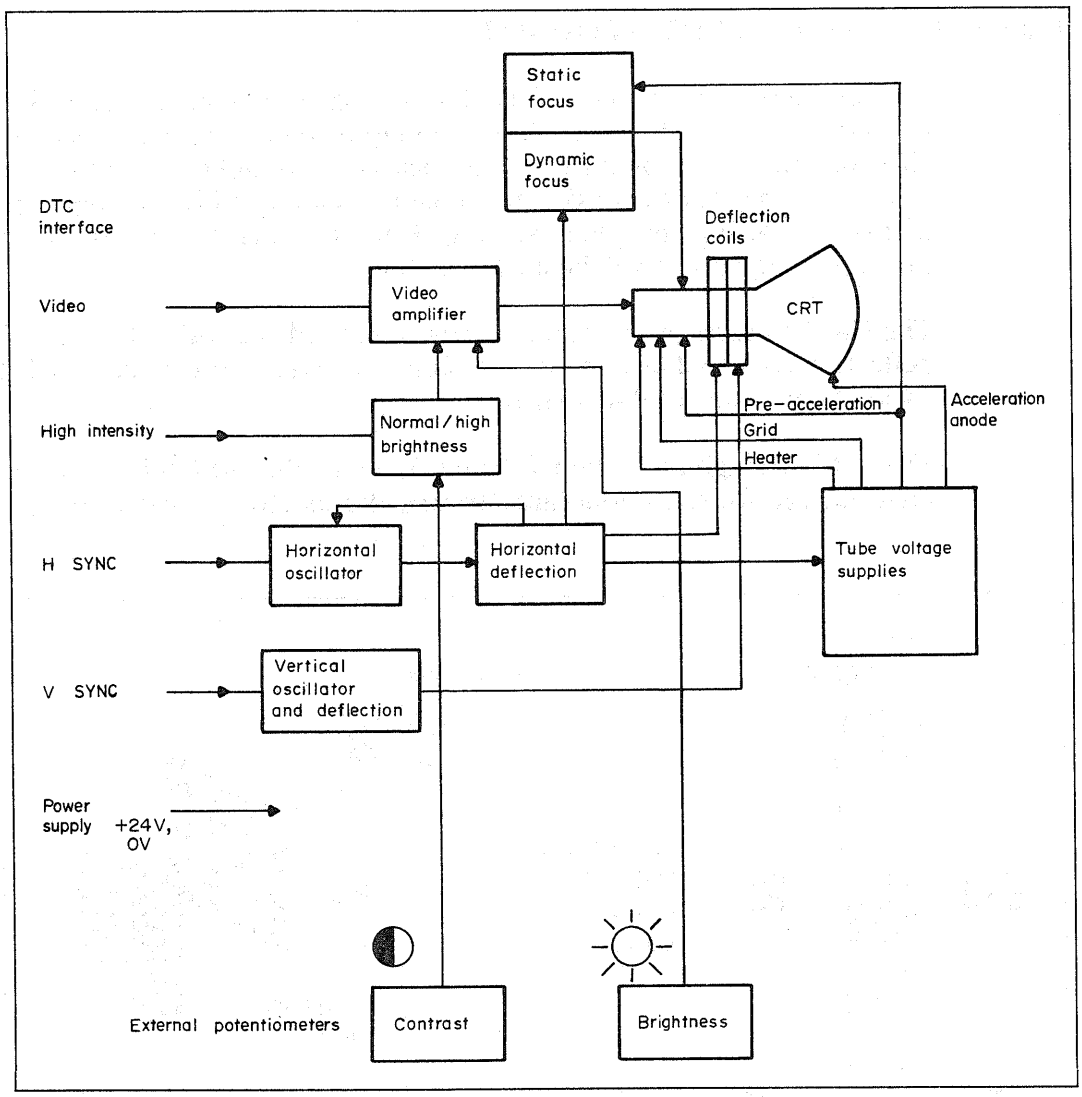


Fig. 8. CRU of DU 4111, block diagram

The main part of the CRB contains circuitry for vertical and horizontal deflection (oscillators, output stages), circuitry for focusing of the beam and for generation of tube voltages. The small part of the CRB is mounted behind the deflection unit on the CRT socket. It contains video amplifier, brightness control circuitry and circuits for flashover protection. The beam is modulated by the video information, thus dots and horizontal lines forming the picture are generated on the phosphor coated screen.

The CRU is supplied with +24 V from the display terminal power supply. Its interface towards the display terminal controller consists of four TTL level signals namely:

- Video.
- High intensity, active when text etc. shall be displayed with higher brightness.
- H SYNC, horizontal synchronization signal. This pulse initiates the horizontal retrace sweep of the beam.
- V SYNC, vertical synchronization signal, initiating the vertical retrace sweep of the beam every 20 ms.

Cathode Ray Tube Unit, CRU, of DU 4112

The cathode ray tube unit (CRU) of DU 4112 generates pictures according to a raster scan method, i.e. three converging electron beams make (if all is on) together a number of consecutive horizontal sweeps over a screen. By means of a mask arrangement each beam hits just one type of colour emitting phosphor. See Fig. 9. If all beams are on at the same time the mixed effect will be a white colour.

The CRU consists of mechanics, a colour cathode ray tube (CRT) with deflection coils and three printed circuit boards (Interface board, CRT socket board and Analog board).

The CRT is a high resolution 90° tube. See Fig. 10. The CRT is fed with high voltage for the anode accelerating the electron beams, voltages

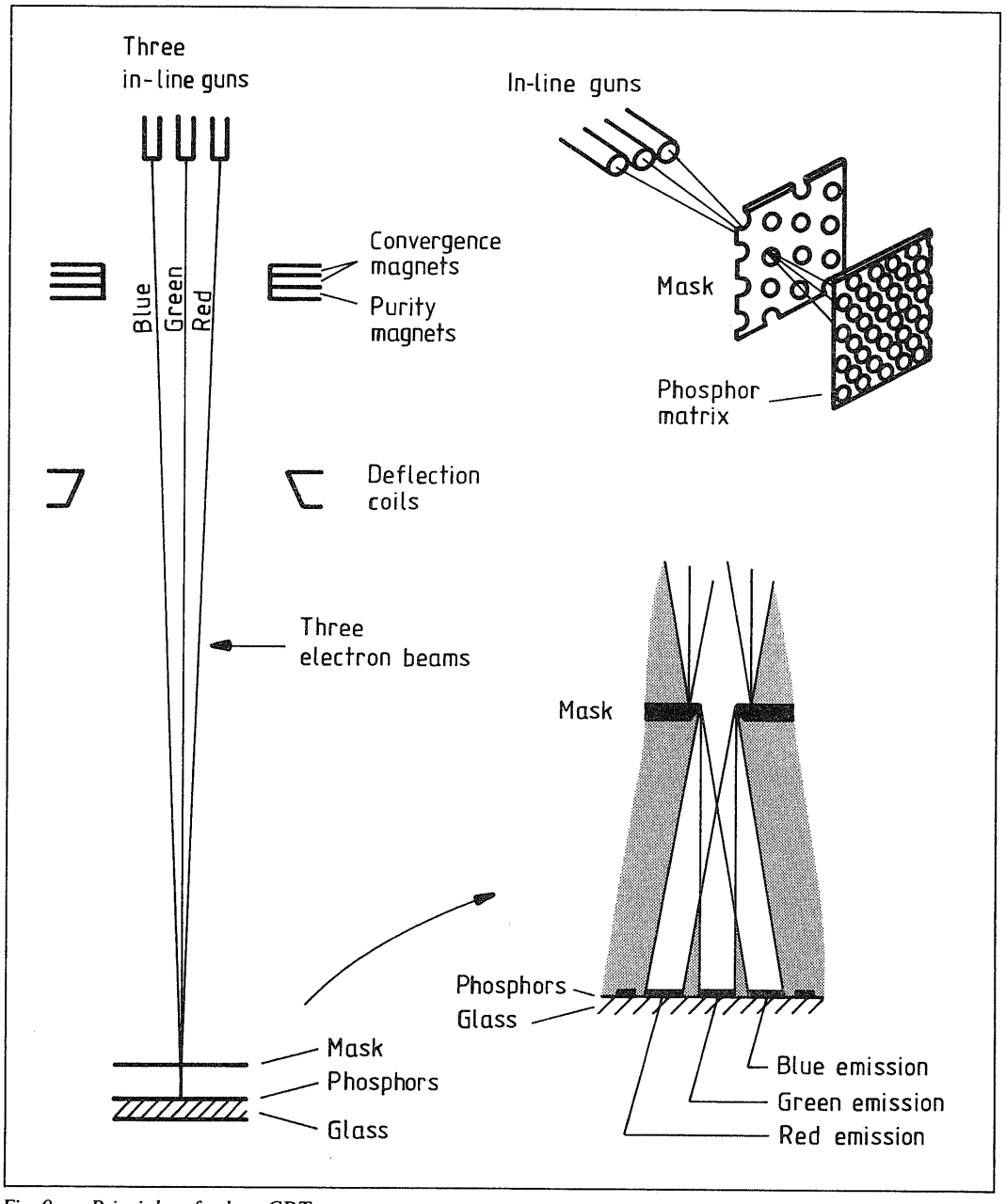


Fig. 9. Principles of colour CRT.

for the electrode focusing the beams and the control grids, current for heating of the cathodes and video currents for the cathodes. It is furthermore fed with signals that affect the deflection coils that govern the vertical and horizontal deflection of the beams.

The three logic boards (see Fig. 10) contain circuitry for vertical and horizontal deflection (oscillators, output stages), pincushion correction control and X-ray supervision that will shut down the horizontal deflection (and thereby also inhibit flyback transformer outputs) if the voltages increase for any reason. The boards also contain video amplifiers, brightness control circuitry and circuits for flashover protection. The beams are modulated by the video information; dots and horizontal lines forming the picture are thus generated on the phosphor coated screen.

The CRU is supplied with 85 V DC from the display terminal power supply. A degaussing coil is activated for a short time at each power on (from circuits in the terminal power supply). The interface to the DTC-C board consist of five TTL level signals namely:

- Three video signals for red, green and blue video.
- H SYNC, horizontal synchronization signal. This pulse initiates the horizontal retrace sweep of the beam.
- V SYNC, vertical synchronization signal, initiating the vertical retrace sweep of the beam every 20 ms.

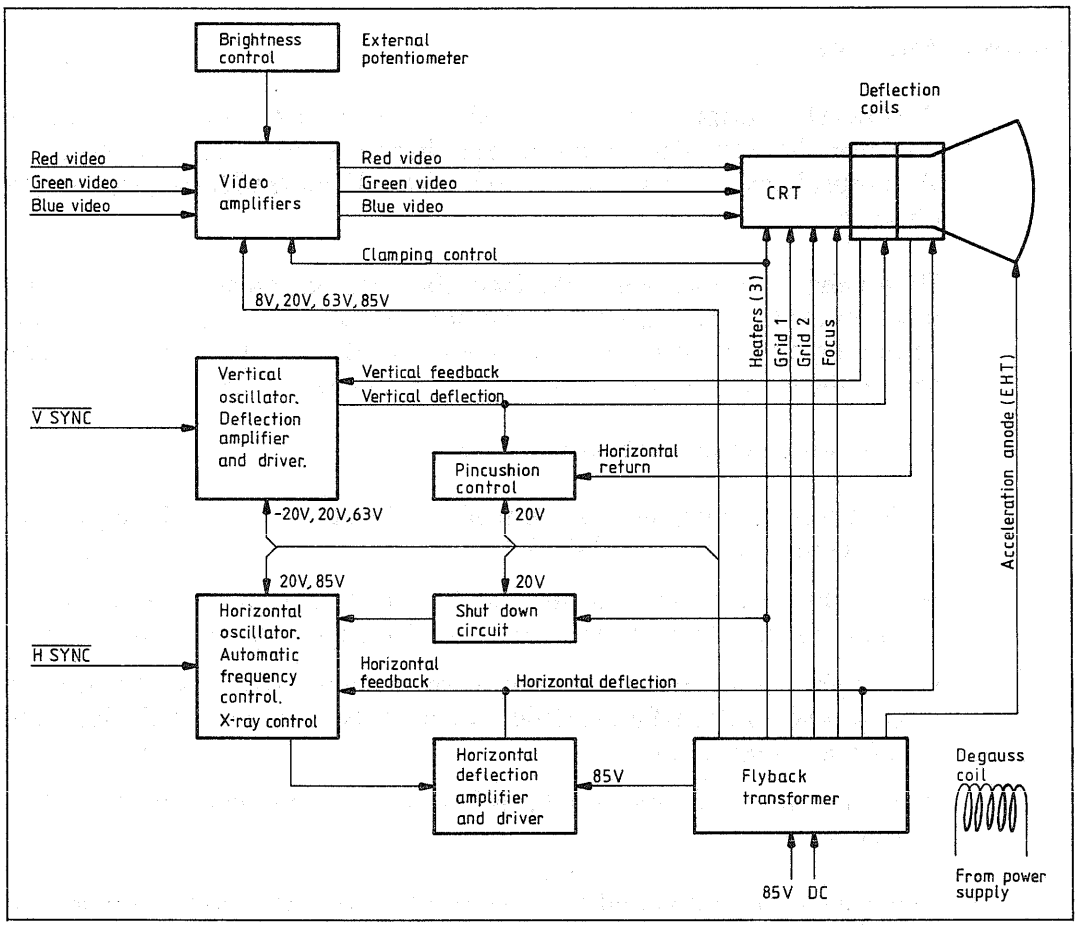


Fig. 10. CRU of DU 4112, block diagram

Keyboard Interface

Data communication with the keyboard unit (and connected devices; KXU, MID) is performed via a keyboard interface on the DTC-B or DTC-C board. The line interface consists of:

- Two power lines (+5 V, 0 V) to the keyboard.
- A reset keyboard line, used to initialize the keyboard. Reset keyboard is activated via the microcomputer peripheral interface adapter (MIC PIA PA1).
- A bidirectional data line used for asynchronous transfers of orders (e.g. alarm on), data (e.g. lamp data) status (e.g. shift key depressed) or key numbers/identity characters. All transmissions are initiated by the microcomputer on the DTC-B or DTC-C board.

The communication procedure is described in the keyboard chapter.

The interface logic consists mainly of an LSI chip, an asynchronous communication adapter, ACIA, which is connected to the internal data bus. The MPU can thus transmit or receive 8-bit words (programmable) via the ACIA. The ACIA converts the parallel data on the data bus to serial data on the keyboard data line and vice versa.

Two-wire Interface

A general description of the two-wire interface is found in the Communication chapter. The functions are therefore only summarized here. A comprehensive treatment is found in the Detailed Description in this chapter.

By means of the two-wire interface, data can be transferred to or from a buffer in the basic read/write memory from or to equipment external to the display unit via a two-wire (or coaxial) line. The transmission on the two-wire is in serial format with a bit rate of 300 kHz.

The two-wire interface consists mainly of:

- A direct memory access controller, DMAC, circuit which handles direct memory access, DMA.
- An advanced data link controller, ADLC, which converts 8-bit parallel data to serial data and vice versa. It also supervises the transmission.
- A high speed modem which converts serial data from NRZ format to frequency shifted data and vice versa. It also generates a bit clock and a Clear to send signal.
- A line interface which contains line drivers, line receivers and a line transformer.

The DMAC is connected to work in two channel mode. Channel 0 is used for transmitting and channel 1 for receiving.

Memory Map

The memory map (see Fig. 11) differs somewhat from the description in the Microcomputer chapter.

The total map is divided into an ordinary read/write memory (RWM) area, an input/output (I/O) area and an initial program load (IPL) area.

The ordinary RWM area is located at the addresses $0000_{(16)}$ – $F6FF_{(16)}$. The physical memory is actually 64 kbytes ($0000_{(16)}$ – $FFFF_{(16)}$) with a duplicate memory of 16 kbytes. (It must be possible for the presentation logic to read two words at the same time from the display memory area.) The normal display memory area is located at the addresses $6000_{(16)}$ – $7FFF_{(16)}$ (8 kbytes) or $7000_{(16)}$ – $7FFF_{(16)}$ (4 kbytes). The display memory area can alternatively be located at the addresses $E000_{(16)}$ – $EFFF_{(16)}$ (4 kbytes, high end location). No parity checking will occur in the RWM and battery backup is not available.

The following circuits and units can be addressed within the I/O area:

- MIC PIA and interrupt register for some central microcomputer functions and MCP for test purposes.
- CRTC and DIA PIA for presentation control.
- KB ACIA for keyboard transfers and control.
- ACA ACIA and PIA for printer transfers and control. A PTM on the ACA-B board can also be addressed for timing purposes. An additional ACA ACIA and PTM on the ACA-D board can also be addressed.
- SS3 address register, ADLC and DMAC for two-wire transfers and control.

Detailed I/O addresses are listed in Appendix 1.

The IPL area consists of 2 kbytes of read only memory and contains among other things four interrupt vectors at the addresses $FFF8_{(16)}$ – $FFFF_{(16)}$.

Interrupt Handling

The principles of the interrupt system on the DTC-B or DTC-C board are quite different from those described in the Microcomputer chapter. They are therefore described below except for the internal MPU functions which of course are the same. The following general microcomputer functions are not used:

- Hardware interrupt prioritizing. (The interrupt register, mask register and address modifier are thus left out.)
- Interrupts out.

The locations of the interrupt vectors (i.e. the addresses to the interrupt routines) are found in the memory map.

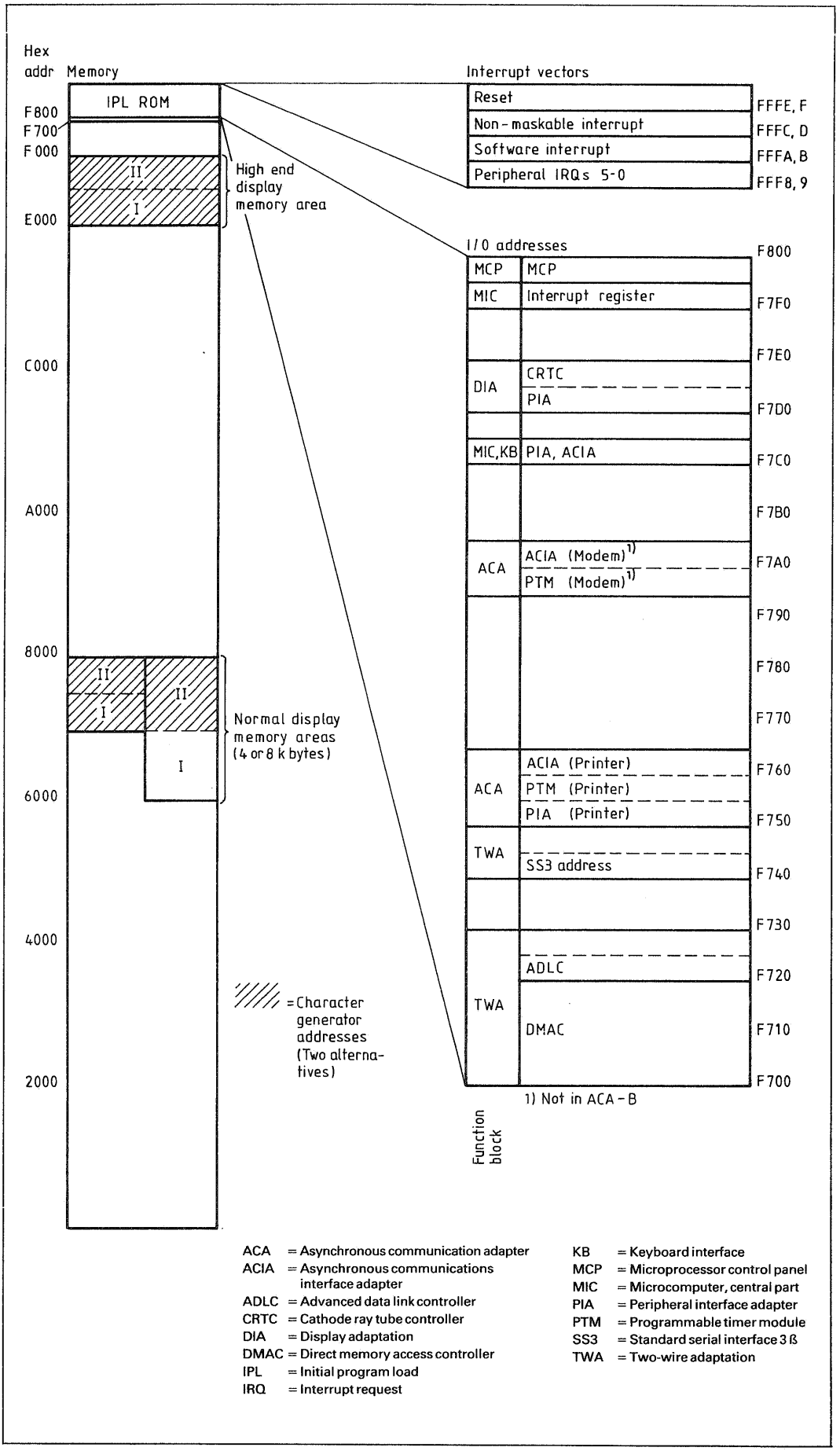


Fig. 11. DU 4111 and DU 4112 memory map

Reset, Non-maskable and Software Interrupts

The MPU and the microcomputer peripheral interface adapter, MIC PIA, are reset at power on. A general reset is also activated.

The MPU is also reset at a depression of the Reset button. The signal from the Reset button resets only the MPU, but is also fed to the control line CB1 of the MIC PIA. The program can thus check the type of reset (by reading the MIC PIA control register B and testing bit 7, the IRQB 1 flag) and then, if suitable, generate a general reset (by setting and re-setting bit 6 of MIC PIA peripheral register B). Note that the general reset signal does not reset the MPU or the MIC PIA.

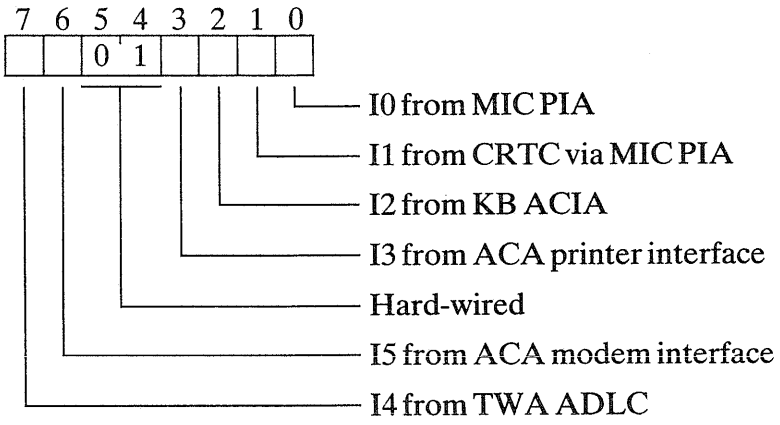
The MPU will after a reset fetch the contents of the memory cells $FFFE_{(16)}$ and $FFFF_{(16)}$ to the program counter and execute a reset program.

NMI, non-maskable interrupt can only be activated by the microprocessor control panel, MCP, and causes the MPU to fetch the address of the interrupt routine from the cells $FFFC_{(16)}$ and $FFFD_{(16)}$.

SWI, software interrupt is generated by an MPU instruction and causes the MPU to fetch the address of the interrupt routine from the cells $FFFA_{(16)}$ and $FFFB_{(16)}$.

Interrupt Requests

The interrupt request, IRQ, input of the MPU can be kept deactivated by the test unit MCP (Microprocessor control panel) or be activated by six interrupts, I0 – I5. The six interrupts are or-ed together. The MPU can check the six interrupts by reading an interrupt register at address $F7FO_{(16)}$:



The interrupts are not latched in the interrupt register and must thus be stable until they are handled by the MPU. The interrupts are further only serviced if the I-bit of the MPU condition code register is reset. The address to the interrupt routine is located at $FFF8_{(16)}$ and $FFF9_{(16)}$.

The interrupts are further explained below:

I5 originates from the ACIA (asynchronous communication interface adapter) or the PTM (programmable timer module) of the modem interface on the ACA-D board.

I4 originates from the ADLC (advanced data link controller) of the two-wire interface.

I3 originates from the ACIA (asynchronous communication interface adapter), the PIA (peripheral interface adapter) or the PTM (programmable timer module) of the printer interface on the ACA-B or ACA-D board.

I2 originates from the ACIA of the keyboard interface.

I1 originates from the CRTC (cathode ray tube controller) of the presentation logic. It is actually the vertical synchronization (V SYNC) signal fed via the MIC PIA and can during presentation be used as a 20 ms timer.

I0 originates from the CA2 output of the MIC PIA. The program can thus make an interrupt by writing $110_{(2)}$ in bits 5 – 3 of the MIC PIA control register A.

Detailed Description

This section will in principle only cover more complicated circuits or special solutions.

Microcomputer

The microcomputer in DU 4111 or DU 4112 differs mainly from the one described in the Microcomputer chapter regarding the address decoder, the use of the MIC PIA and the interrupt logic. The MPU is also connected in a somewhat different way.

Basic Timing

A basic 19.17 MHz clock signal is generated by a crystal clock. See Fig. 12. From this clock all timing signals, e.g. dynamic memory timing, two-wire transfer bit clock etc. are derived. Five 2.13 MHz clocks are generated (T1 – T5). Combinations of these are used to define different points of time during each half of the system clock period. The system clock, ∅, (microprocessor instruction clock) has a frequency of 1.065 MHz. It appears in several phases and with different pulse-pause ratios to compensate for propagation delays etc.

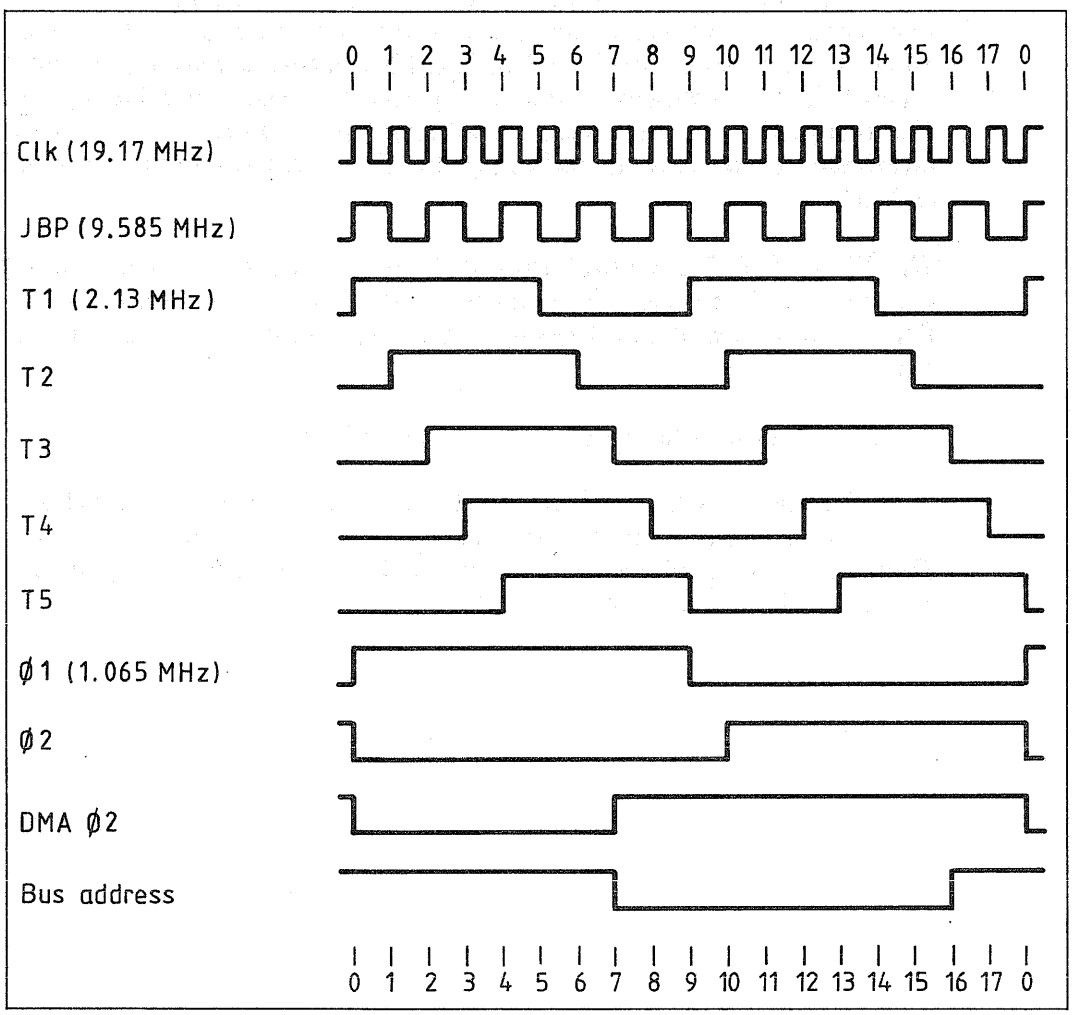


Fig. 12. Basic timing signals

The MPU demands two non-overlapping complementary clocks. Circuitry is added to provide these; MPU ∅1 and MPU ∅2. (Observe that they are only fed to the MPU.) They differ from the other ∅ signals in that they may be frozen in the ∅1 state by a stretch signal (STR). The stretch signal is generated during a direct memory access.

MPU

The NMI (non-maskable interrupt) and HALT inputs of the MPU are only connected to the MCP and the TSC (three-state control) input is hard-wired inactive. The address bus and the R/W line are taken to high impedance state by external buffers at direct memory access and the data bus by deactivation of the DBE (data bus enable) input. The DBE input is controlled by the DMA signal from the address decoder.

Address Decoding and Direct Memory Access

The principles of the address decoder is the same as said in the Microcomputer chapter but the build-up and the signals to and from the address decoder are different. The decoder consists of one FPLA (field programmable logic array) and one PAL (programmable array of logic gates).

The FPLA functions are summarized on the opposite page. The D1 (= DMA grant) and D2 signals are generated as a response to a DMA request and makes it possible for the FPLA to inhibit output signals during DMA cycles 1 – 3 or 2 as well as to generate the DMA and stretch (STR) signals. The rest of the output signals has addressing functions or controls the internal/external data bus buffer (INTERN) or the read/write memory (RWM).

The TWA, F7CD and DMA outputs from the FPLA are then together with ∅2, R/W and AI6 – AI3 (address bits 6 – 3) further decoded in the PAL to more specific addressing signals, summarized on the opposite page.

See also Memory Map (Fig. 11) and Appendix 1

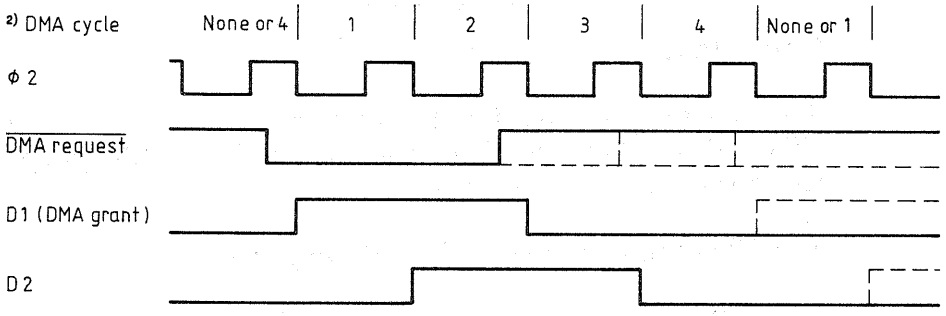
Note that the separation of internal and external address and data buses is not as stringent as in the Microcomputer chapter. See Appendix 2. Thus direct memory access from the DMAC affects both the internal address and internal data buses.

Address decoder, FPLA functions

Activating input combinations					Activated output		Comments
A[1] 15–4 Hex	VMA	D1[2]	D2[2]	R/W	Signal	Active level	
000–FFF	X	High	High	X	DMA	Low	DMA cycle 2
000–FFF F70–F72	Low Low	High One high	High	X X	STR	Low	←DMA cycle 2 ←DMAC, ADLC addressed
F70–F7F	Low	One low		X	PER I/O	Low	Peripheral I/Os enabled
F70–F72 F74 F7C–F7D F7F F80–FFF	Low Low Low Low Low	Low One low One low One low One low	Low	X X X X High	INTERN[3]	Low	←DMAC, ADLC addressed
F70–F72 F74 F7F	Low Low Low	Low One low One low	Low	X X X	TWA[4]	High	←DMAC, ADLC addressed
F7C–F7D	Low	One low		X	F7CD[4]	High	
F80–FFF	Low	One low		High	F8XXR	Low	IPL ROM Read
000–FFF F70–F7F F80–FFF	High X X	One low One low One low		X X High	RWM[5]	High	←VMA inactive ←I/Os addressed ←IPL ROM addressed

[1] A = Address from MPU.

[2] DMA cycle



Thus D1, D2: Both high = DMA cycle 2 Both low = Not DMA cycle 1–3
One high = DMA cycle 1–3 One low = Not DMA cycle 2

[3] Sets the internal/external data bus buffer in (MPU) read position.

[4] See below.

[5] Disables the read/write memory.

Address decoder, PAL functions

Activating input condition									Activated output		Output activated by address
Ø2	DMA	TWA	F7CD	R/W	AI6	AI5	AI4	AI3	Signal	Active level	
X	X	High	X	X	Low	Low	X	X	DMACS	Low	F700-1F
X or X	Low X	X High	X X	X X	X Low	X High	X Low	X Low	ADLCCS	Low	F720-7
Low	X	High	X	High	High	Low	X	Low	SS3	Low	F740-7
X	X	X	High	X	X	X	High	X	DIA PIA	High	F7D0-F
X	High	X	High	X	High	Low	High	High	CRTCS	Low	F7D8-F
Low	X	High	X	High	High	High	High	Low	IRQR	Low	F7F0-7
X	X	High	X	X	High	Low	High	High	MCP	Low	F7F8-F
X	X	X	X	X	X	X	Low	Low	A3 · A4	Low	—

Microcomputer PIA, MIC PIA

The MIC PIA controls several central functions. See chapter Microcomputer; Peripheral Interface Adapter, PIA for definitions and a general survey. PA6 – PA4, IRQB, CB2 and PB2 – PB0 are, however, not used.

The vertical synchronization signal is fed to the CA1 input and will thus set an interrupt flag (CRA bit 7) and activate the IRQA output (if CRA bits 1, 0 = $11_{(2)}$) which results in interrupt 1 to the MPU.

The CA2 output is used to generate interrupt 0 if CRA bit 3 is zero-set (and CRA bits 5, 4 = $11_{(2)}$).

The PA outputs/inputs are used in the following way:

PA7	Input	Hard-wired low.
PA3	Output	Not used in DU 4111. In DU 4112: 0 = Monochrome mode. 1 = Base colour mode. (Four colours.)
PA2	Input	0 = MCP connected and in test mode. 1 = MCP not connected.
PA1	Output	0 = Keyboard not reset. 1 = Keyboard reset.
PA0	Input	Hard-wired low.

An interrupt flag (CRB bit 7) is set if the Reset button is depressed (as CB1 is then taken low). This must be checked by the program as the IRQB output is not connected.

The PB outputs are used in the following way:

PB7 0 = High end display memory position ($E000_{(16)}$ – $EFFF_{(16)}$). PB3 must be = 1.
1 = Normal display memory position ($6000_{(16)}$ – $7FFF_{(16)}$ or $7000_{(16)}$ – $7FFF_{(16)}$).

PB6 1 = Activate general Reset. The output must be set low after power on. The MPU and the MIC PIA are not reset by general Reset.

PB5 0 = Address mode 0. See Display Memory Addressing.
1 = Address mode 1.

PB4 0 = Disable character generator to bus.
1 = Enable character generator to bus.

PB3 0 = 8 kbytes display memory.
1 = 4 kbytes display memory.

Read/Write Memory

Data Flow

The read/write memory consists of one 65536 × 8 bits and one 16384 × 8 bits block of dynamic random access memory. See Fig. 13. They are denoted "64 k block" and "16 k block".

Input data is fed to both memory blocks from the 8-bit external data bus. Output data is fed to:

- The latches A and B (see Fig. 5). The data is here latched each time Ø1 goes low and is then fed further to the presentation logic.
- The "64 k" and "16 k" latches (see Fig. 13). The outputs of these latches are in high impedance state when the output control inputs, tied to D64 and D16 respectively, are inactive. One of the signals D64 and D16 may be activated (by the address decoding logic) during Ø2 of a read cycle from the MPU or the direct memory access controller, DMAC, in the two-wire interface. The contents of the applicable latch is then fed to the external data bus.

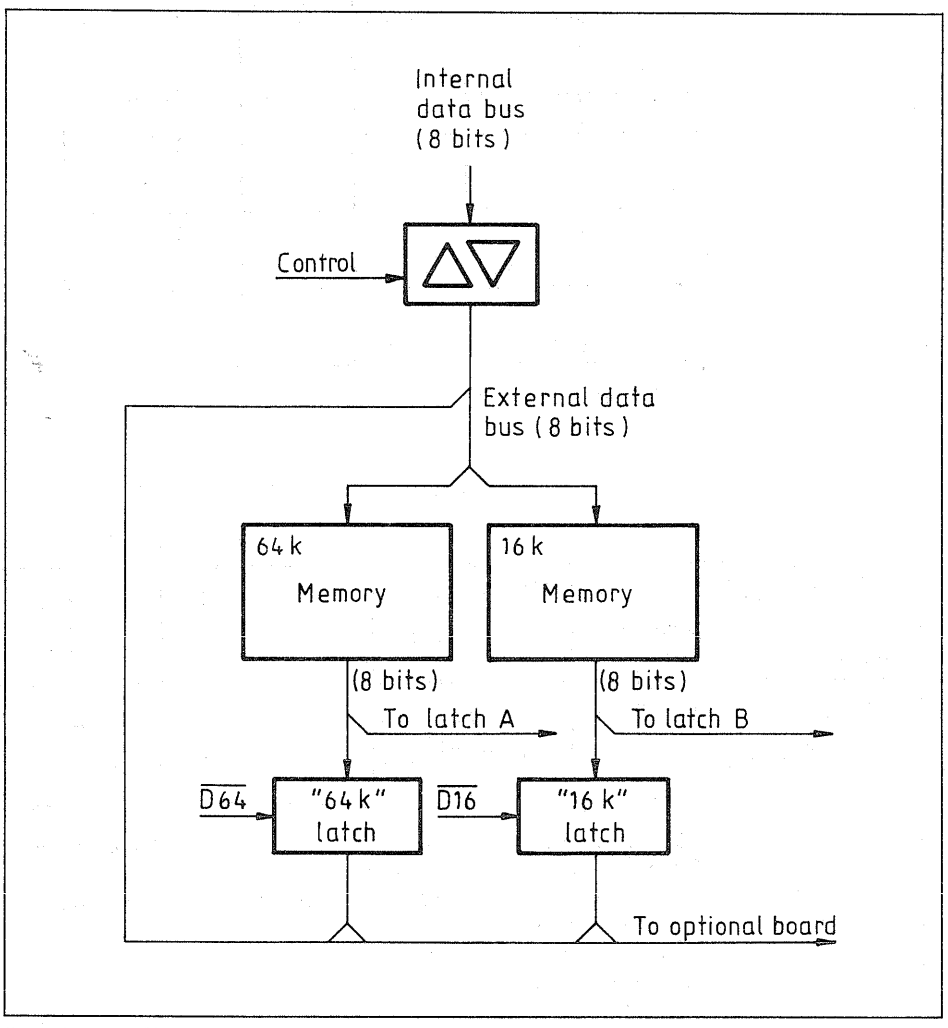


Fig. 13. Read/Write memory data flow

Addressing and Timing

Each memory block is organized in a number of rows and 8-bit columns. See Fig. 14. At a memory access a row address is first strobed into both memory blocks by a Row address strobe and then a column address by a Column address strobe. At a write operation a Write enable signal to the applicable block is also activated. There are, as can be seen from Fig. 15, two complete addressings during a system clock cycle (Ø1 high and low time).

A 16-bit address is fed from a bus/CRTC address selector to an address selector and refresh generator chip, which splits the address into a row address and a column address. The row address is propagated to both memory blocks when a timing signal (Address selecting) is high and the column address when the timing signal is low unless when a Refresh signal is active. In the latter case a row (refresh) address from an internal refresh counter is inverted and fed to the memory. (Only row addressing is necessary in order to obtain the refresh function.) The refresh counter will be incremented at each refresh addressing. A refresh cycle will take place in each system clock cycle when no presentation (CRTC cycle) occurs.

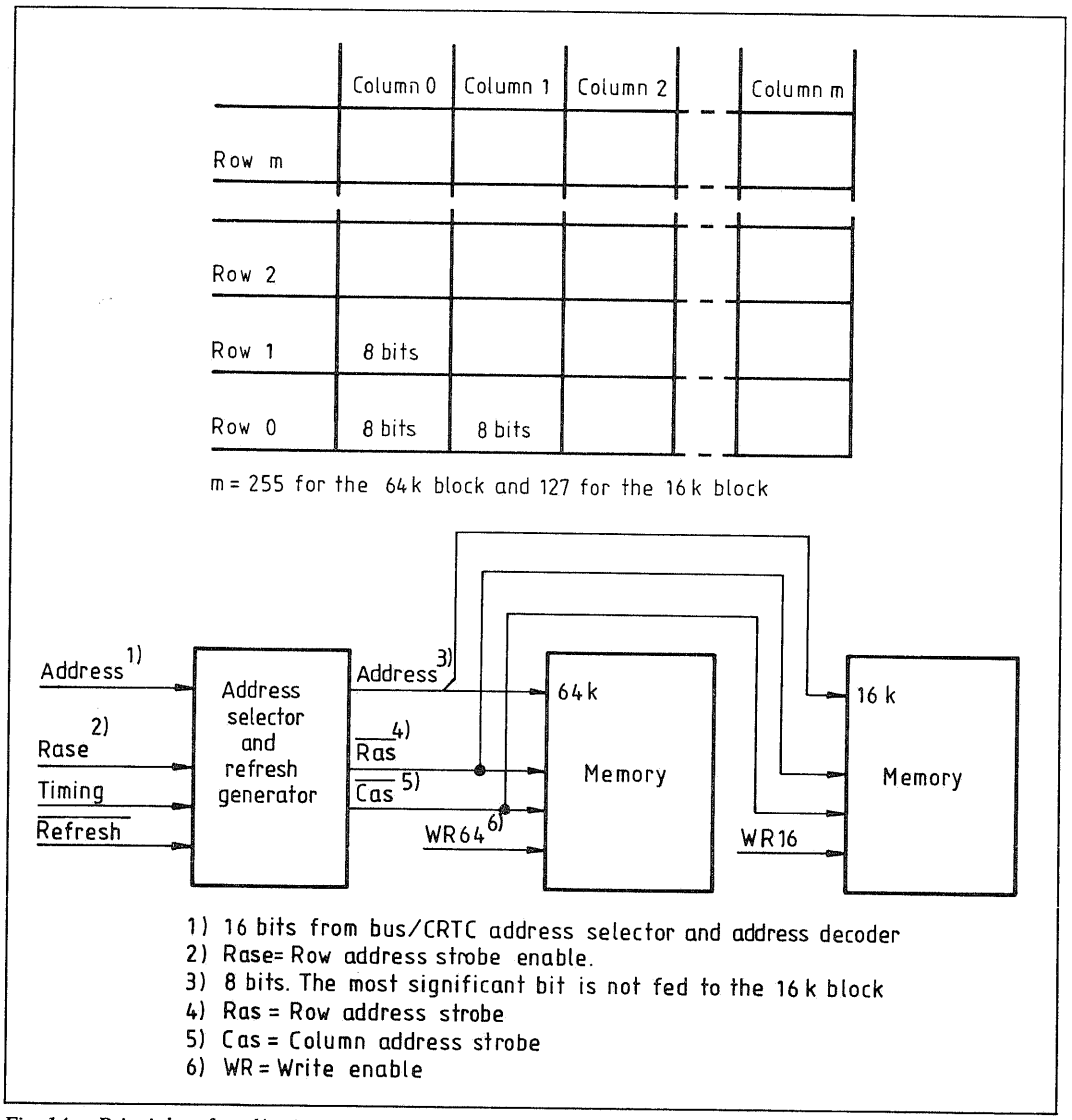
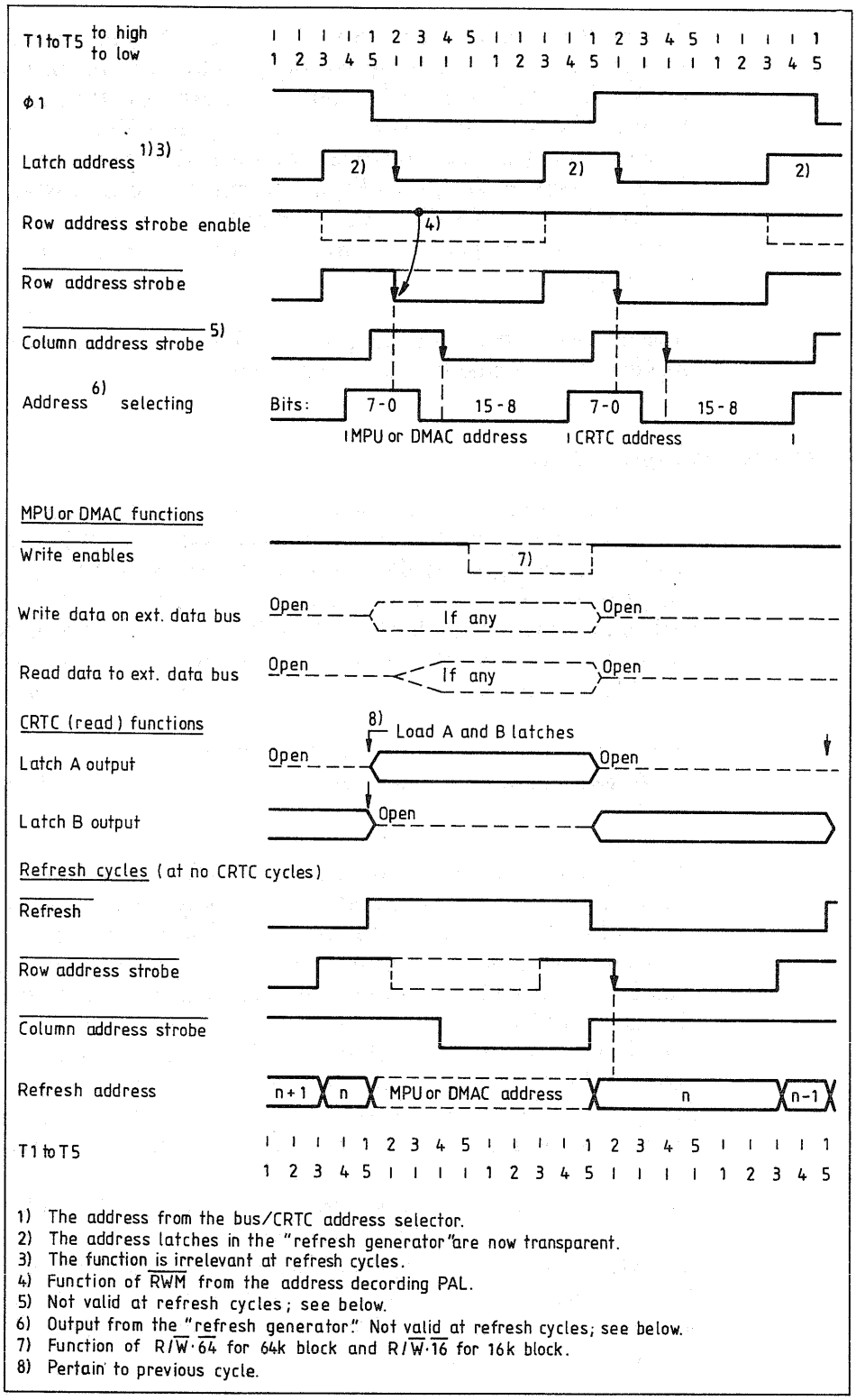


Fig. 14. Principles of read/write memory addressing

To sum up; during ∅1 high time the address may originate from the cathode ray tube controller (CRTC) or the refresh counter in the address selector and refresh generator.

During ∅1 low time the address may originate from the MPU or the direct memory access controller, DMAC. The Row address strobe may, however, be inhibited by a high RWM signal from the address decoding FPLA which will delete a memory operation.



1) The address from the bus/CRTC address selector.
2) The address latches in the "refresh generator" are now transparent.
3) The function is irrelevant at refresh cycles.
4) Function of $\overline{RWM}$ from the address decording PAL.
5) Not valid at refresh cycles; see below.
6) Output from the "refresh generator". Not valid at refresh cycles; see below.
7) Function of $R/\overline{W} \cdot \overline{64}$ for 64k block and $R/\overline{W} \cdot \overline{16}$ for 16k block.
8) Pertain to previous cycle.

Fig. 15. Read/Write memory timing

Memory Address Selection

Four PALs (programmable arrays of logic gates) and an or-gate, forming an address decoder and a bus/CRTC address selector, are used to generate the necessary control signals and address rearrangement for the functions described below. See also Memory Map.

Read/Write Memory Addressing

Except from what is said under the next heading about the display memory area all read/write memory addresses are located in the 64 k block. The MPU can read from the addresses $0000_{(16)}$ – $F6FF_{(16)}$ and write at the addresses $0000_{(16)}$ – $F6FF_{(16)}$ and $F800_{(16)}$ – $FFFF_{(16)}$. The direct memory access controller can (at DMA) perform read or write at the addresses $0000_{(16)}$ – $FFFF_{(16)}$. Memory operations are inhibited in the $7XXX_{(16)}$ or $EXXX_{(16)}$ areas (depending on display memory allocation) during character generator manipulations (see Character Generator Loading).

(The address selector rearranges the address bits, but this has no functional implications outside the display memory area.)

Display Memory Addressing

The display memory size can be 4 or 8 kbytes. (MIC PIA PB3 = 1 or 0.) It is normally located at the addresses $7000_{(16)}$ – $7FFF_{(16)}$ or $6000_{(16)}$ – $7FFF_{(16)}$ but a high end allocation can be ordered (MIC PIA PB7 = 0) if 4 kbytes are selected and it is then allocated to the addresses $E000_{(16)}$ – $EFFF_{(16)}$. Both the 64 k block and the 16 k block are used for display memory functions. This leads to byte allocation rearrangements and might lead to inaccessible areas when switching memory size.

The display memory can be addressed in two ways from the MPU (or DMA logic) denoted address mode 0 and 1 (selected by MIC PIA PB5 = 0 or 1).

Two logical areas are defined; area I and II. See Fig. 16. In address mode 0, 2 or 4 kbytes (depending on display memory size) are assigned to area I and 2 or 4 kbytes to area II. The bytes and areas are arranged consecutively as seen from the MPU or DMA logic. In address mode 1 area I consists of 2 or 4 kbytes at even addresses and area II of 2 or 4 kbytes at odd addresses as seen from the MPU or DMA logic. The selection of address mode does not affect the addressing of other areas than the display memory nor the addressing from the presentation logic. The picture on the screen will thus not change if the address mode is changed without a change of the display memory contents.

Fig. 17 shows how the bytes belonging to area I and area II are allocated in the 64 k and 16 k blocks. In address mode 0 bit 11 or 12 from the MPU or DMA logic is fed to the least significant memory address bit (after the address selector) and bit 0 selects the memory block. At consecutive addresses from the MPU or DMA logic all bytes of area I are first accessed and then all bytes of area II.

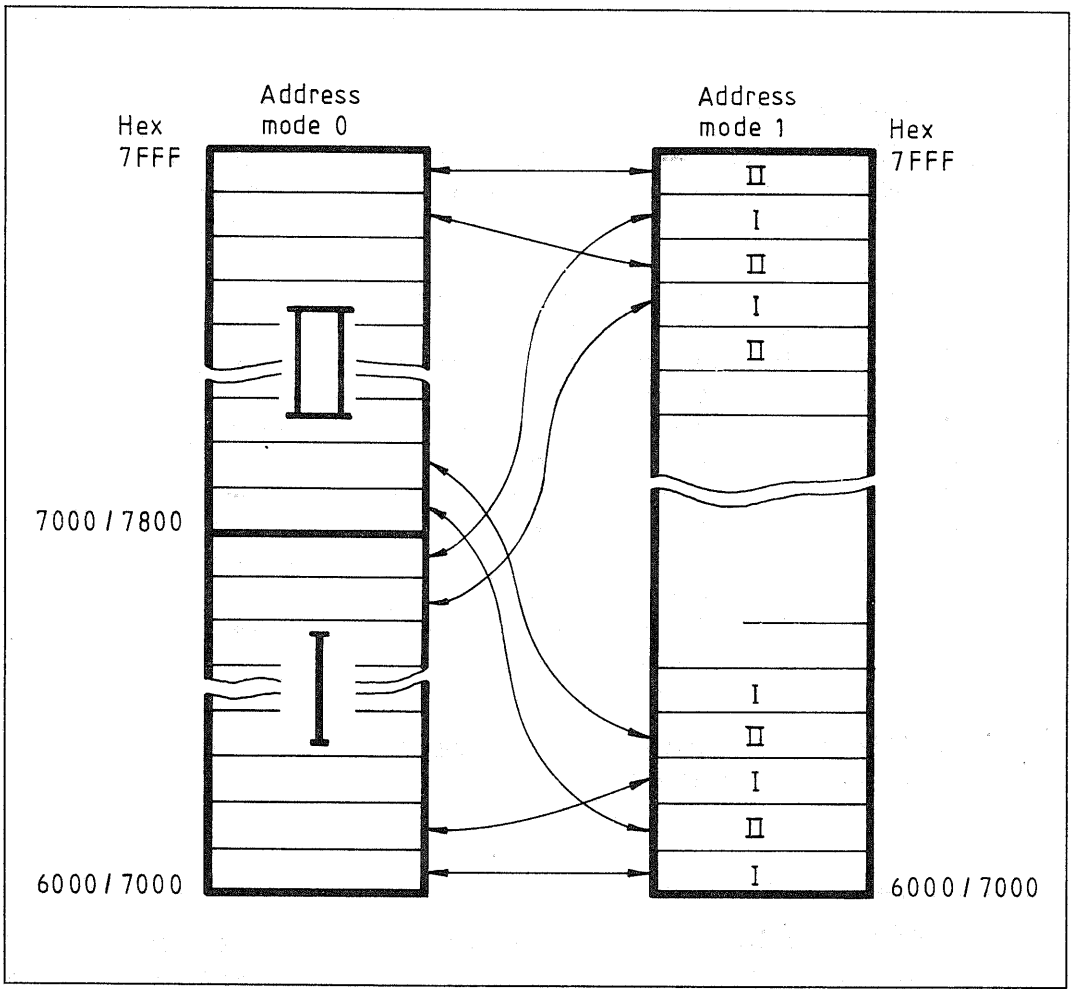


Fig. 16. Principles of address mode changes

The presentation logic can only access the display memory area. As said earlier one display memory access from the presentation logic (the CRTC) supplies one byte from the 64 k block and one byte from the 16 k block which then in order are fed to the character generator and, during sweep 0, also to the line memory. (The character generator will, however, during sweep 0 normally not generate any bright dots independent of the character code input.) During the rest of the sweeps (1, 2, 3 etc.) of a line, the line memory contents will be fed to the attribute generator synchronously with the character code stream from the display memory to the character generator.

The presentation can, depending on the display adapter peripheral interface adapter, DIA PIA, outputs PB2 and PB1, work in four different modes:

DIA PIA		Presentation mode	Presents simultaneously via	
PB2	PB1		character generator	attribute generator
1	1	0	Area I	Area II
1	0	1	Area II	Area I
0	1	2	Area II[1)]	Area II[2)]
0	0	3	Area I + II[3)]	Area I + II[4)]

[1) 2)] Same byte. [3) 4)] Same byte.

Address mode 0

MPU or DMA address bus bit				64 kbytes memory block	16 kbytes memory block
2	1	11[1]	0		
0	0	0	0	I_a	
0	0	0	1		I_a
0	0	1	0	II_d	
0	0	1	1		II_d
0	1	0	0	I_b	
0	1	0	1		I_b
0	1	1	0	II_e	
0	1	1	1		II_e
1	0	0	0	I_c	
1	0	0	1		I_c
1	0	1	0	II_f	
:	:	:	:	:	:

[1] At 8 kbytes display memory bit 12 denotes the display memory area I and II instead of bit 11.

Address mode 1

MPU or DMA address bus bit				64 kbytes memory block	16 kbytes memory block
3	2	1[2]	0[3]		
0	0	0	0	I_g	
0	0	0	1	II_h	
0	0	1	0		I_g
0	0	1	1		II_h
0	1	0	0	I_i	
0	1	0	1	II_j	
0	1	1	0		I_i
0	1	1	1		II_j
1	0	0	0	I_k	
:	:	:	:	:	:

[2] Denotes memory block [3] Denotes memory area I or II.

Presentation

Pres. mode	Sweep	Red bytes	(Contents)
0	0	II_h, II_j ...	(FACs)
0	not 0	I_g, I_i ...	(Char.)
1	0	I_g, I_i ...	(FACs)
1	not 0	II_h, II_j ...	(Char.)
2	all	II_d, II_e ...	(FACs/Char.)
3	all	I_a, I_b ... II_d, II_e ...	(FACs/Char.)

Fig. 17. Principles of byte allocations to display memory areas and blocks

The presentation mode can be selected independently of the address mode.

The utilization of address and presentation modes enables the use of "visible" or "invisible" field attribute characters. It will also make some memory manipulations easier and less time consuming. The presentation modes function in the following way (the references I_a, I_b, I_c, II_d, II_e etc. are made to Fig. 17):

- Presentation mode 0: The contents of odd (as seen from the MPU or DMA address bus) display memory addresses (belonging to area II) are during sweep 0 fed to the line memory (and the character generator, but the generator is normally programmed to present dark dots during sweep 0), i.e. II_h, II_j etc. Even display memory positions (belonging to area I) are during all other sweeps of a line fed to the character generator, i.e. I_g, I_i etc. Area II shall thus contain the field attribute characters and area I the text character codes. This also implies that there exist two memory positions for each screen position, out of which one may be used for a field attribute character where the other contains a character code.
- Presentation mode 1: The contents of even addresses are fed to the line memory (and the character generator) during sweep 0, i.e. I_g, I_i etc. and the contents of odd addresses to the character generator during the rest of the sweeps, i.e. II_h, II_j etc. Area I shall thus contain the attributes and area II the characters.
- Presentation mode 2: The contents of even and odd display memory addresses of area II (a high level is applied instead of address bits 11 or 12) are during sweep 0 fed to the line memory, i.e. II_d, II_e etc. The same display memory positions are during all sweeps of a line fed to the character generator, i.e. II_d, II_e etc. The field attribute characters, if used, must thus be mixed with the character codes of the text. The field attribute characters activate the attribute generator but may not activate the character generator (present a space). The character codes activate the character generator but may not (except for $5F_{(16)}$, $00_{(16)}$ and $20_{(16)}$ with special functions) activate the attribute generator. The field attribute characters and character codes must thus be different (with the exception mentioned above).
- Presentation mode 3: The contents of even and odd display memory addresses of area I (the area selection is controlled by bit 11 from the cathode ray tube controller for 4 kbytes of display memory and by bit 12 for 8 kbytes) are during sweep 0 fed to the line memory, i.e. I_a, I_b etc. The same display memory positions are during all sweeps of a line fed to the character generator. When area I is finished area II will be handled in the same way as area I, i.e. II_d, II_e etc. will be taken care of. The same as said under presentation mode 2 is valid for mixing of field attribute characters and character codes.

In order to get a natural connection to the address modes the references above are made to Fig. 17 and the MPU or DMA addresses. Note, however, that the presentation modes are dependent upon the addressing from the cathode ray tube controller, which addressing also is rearranged in the bus/CRTC address selector. The extremely interested reader can find details in Appendix 3.

The presentation addressing can wrap around i.e. the first text line of the screen can be fetched from an arbitrary start address in the display memory and when the last position of the display memory is reached the addressing will go to the first position in the display memory. See Fig. 18. This function is, however, controlled by the start address programming of the cathode ray tube controller.

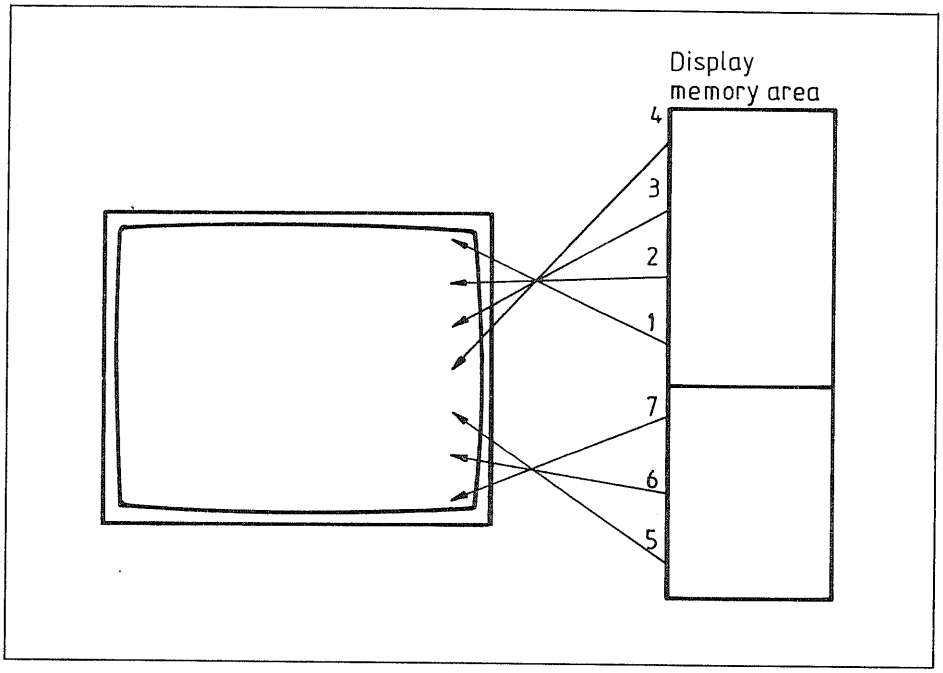


Fig. 18. Principles of wrap around

Character Generator Loading

Memory operations are inhibited at the addresses $7XXX_{(16)}$ (if MIC PIA PB7 = 1) or $EXXX_{(16)}$ (if MIC PIA PB7 = 0) when Enable character generator to bus is set (MIC PIA PB4 = 1) and a flip-flop change will generate $\overline{CHG}$ low and CHG high. This will switch over the character generator control to the microcomputer. See Fig. 19. The bus buffers will then enable addressing of the character generator by bits 11 – 0 on the address bus and bidirectional data transfers between the character generator and the external data bus. Control signals are also provided for write and read operations in the character generator.

The address bus bits 11 – 4 are used to address 256 character "files" and the address bus bits 3 – 0 to address 16 sweep dot patterns (8-bit locations) in each character "file". For example, assume that the A in Fig. 4 has the character code $41_{(16)}$. The character generator must then be loaded as follows in order to generate the dot pattern in the figure:

Address	Data bus contents	Character generator contents
$X^{1)}41^{2)}0^{3)}{}_{(16)}$	$00_{(16)} = 0000\,0000_{(2)}$	$1111\,1111_{(2)}$
$X411_{(16)}$	$00_{(16)} = 0000\,0000_{(2)}$	$1111\,1111_{(2)}$
$X412_{(16)}$	$1C_{(16)} = 0001\,1100_{(2)}$	$1110\,0011_{(2)}$
$X413_{(16)}$	$36_{(16)} = 0011\,0110_{(2)}$	$1100\,1001_{(2)}$
$X414_{(16)}$	$63_{(16)} = 0110\,0011_{(2)}$	$1001\,1100_{(2)}$
etc.		

[1] X = 7 or E. [2] 41 = character code. [3] 0 = raster address.

In practical applications the character generator is normally loaded directly from a diskette via the two-wire and the direct memory access controller.

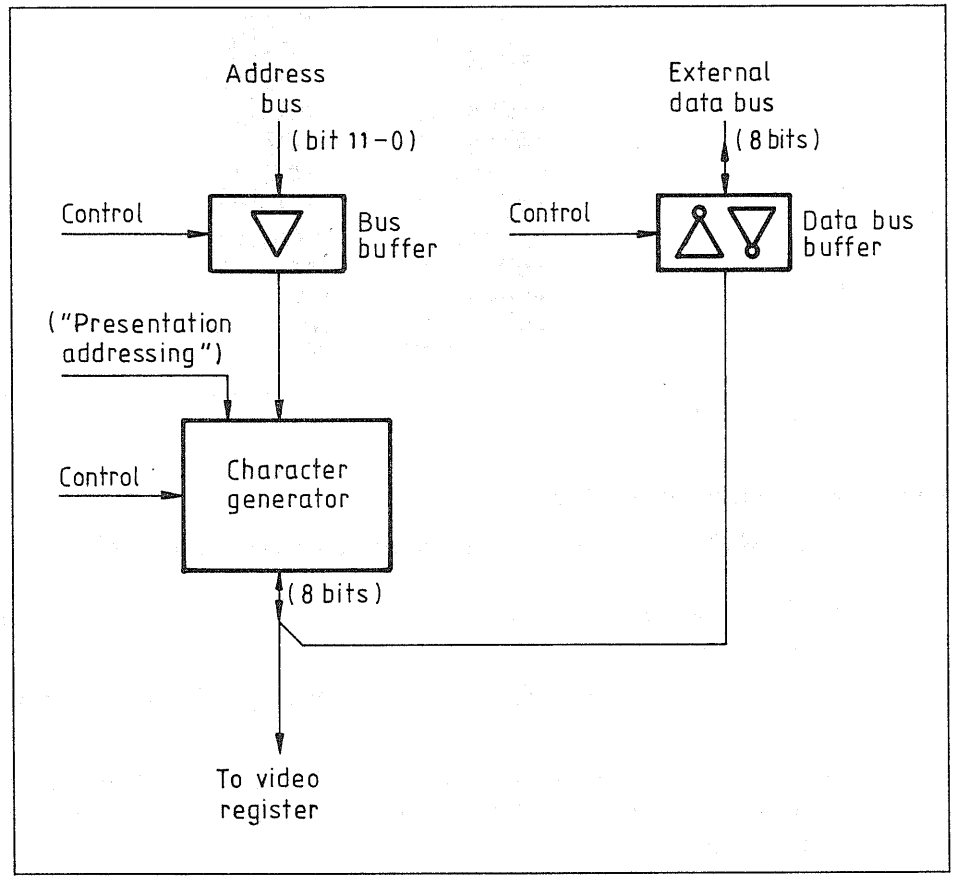


Fig. 19. Principles of character generator loading

Cathode Ray Tube Controller, CRTC

The presentation on the screen is supervised by the cathode ray tube controller (CRTC) shown in Fig. 20.

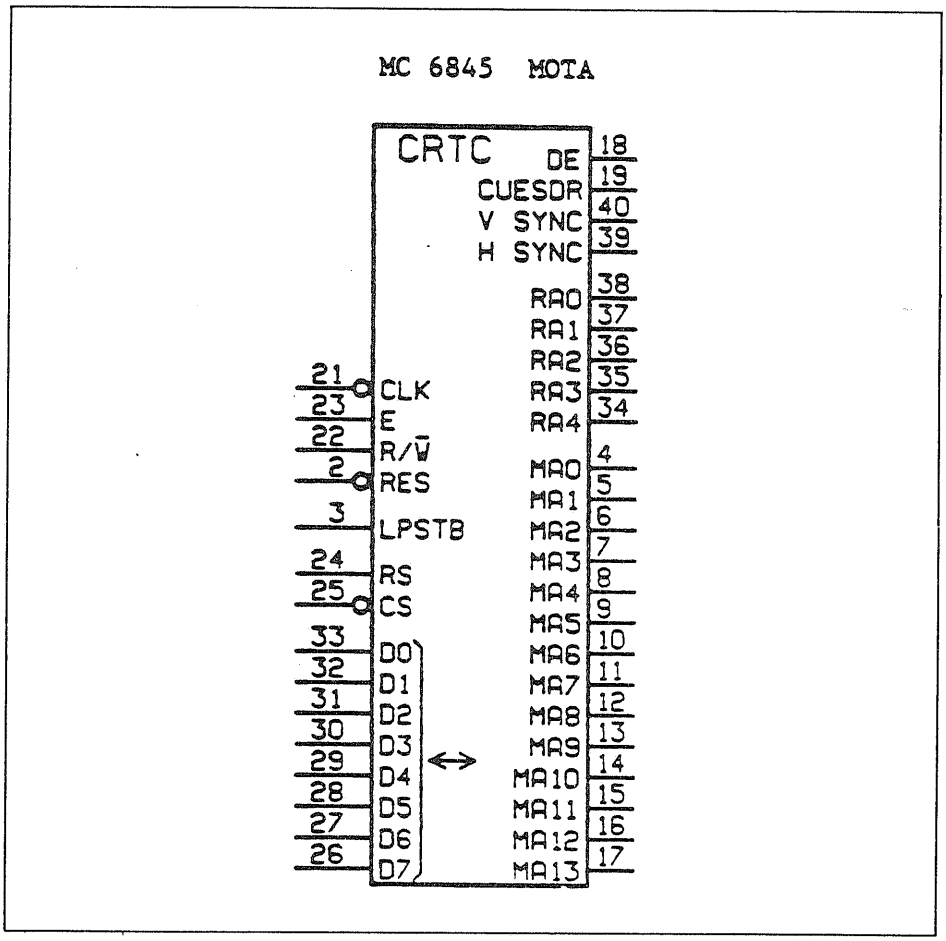


Fig. 20. CRTC drawing symbol

CRTC – MPU Interface and Reset

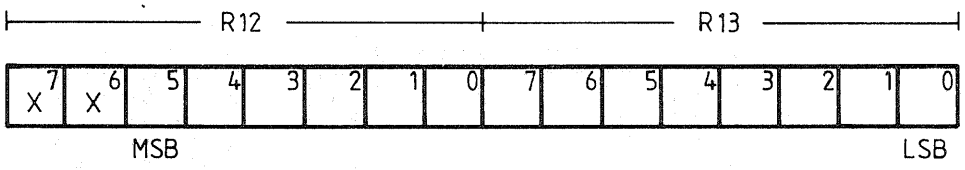
The CRTC interfaces the microcomputer via the following lines:

- Eight bidirectional data lines (D7 – D0) connected to the internal data bus.
- One read/write line (R/W, Low = Write) fed from the R/W output of the MPU.
- One enable line (E) which enables the inputs/outputs of the MPU interface and clocks data to and from the CRTC. It is driven by the Ø2 clock.
- One chip select line (CS) activated when the CRTCS signal from the address decoder is active (CS is thus activated by the internal addresses $F7D8_{(16)}$ – $F7DF_{(16)}$, see Memory Map).
- One register select line (RS) connected to address bit 0 (from the MPU). The CRTC contains one 5-bit write-only address register and 15 other registers of variable length that are accessible from the address bus. The address register is accessed when CS is active and RS = 0, i.e. at the addresses $F7D8_{(16)}$, $F7DA_{(16)}$, $F7DC_{(16)}$ or $F7DE_{(16)}$. One of the 15 registers is addressed when CS is active and RS = 1, i.e. at the addresses $F7D9_{(16)}$, $F7DB_{(16)}$, $F7DD_{(16)}$ or $F7DF_{(16)}$. The contents of the address register decides which one of the 15 registers that is accessed, e.g. register 5 (R5) is accessed when the address register contents is $XXX00101_{(2)}$.

The CRTC can be reset by a signal from the display adapter peripheral interface adapter (DIA PIA) fed to the reset (RES) input of the CRTC. All outputs go to low level and all internal counters are cleared at reset but the internal registers are unaffected. The CRTC must be kept reset during or be reset after initial programming.

CRTC Registers

The above mentioned registers are summarized in the table below. The most significant bits are not used when a register contains less than eight bits. When a double register contains more than eight bits it is formatted as in the following example:



Register	Number of bits	Name	Function
R0	8	Character times per sweep	Determines the total number of horizontal character (displayed + non-displayed) position times per sweep minus one.
R1	8	Characters per line	Determines how many characters should be displayed on a line.
R2	8	H SYNC position	Determines at which horizontal character position the H SYNC signal should start.
R3	4	H SYNC width	Determines the number of character times the H SYNC signal should be active.
R4	7	Lines total	Determines the total number of text lines (displayed + non-displayed) minus one.
R5	5	Vertical total adjust	Determines how many sweep times should be added to the "Lines total" time to form a full frame time.
R6	7	Lines displayed	Determines how many lines should be displayed.
R7	7	V SYNC position	Determines after which line the V SYNC signal should be generated (V SYNC is active during the time of one text line).
R8	2	Interlace control	Should be 00 or 10 as interlace is not used.
R9	5	Maximum raster address	Determines the number of sweeps per text line minus one. (Must match with DIA PIA programming.)
R10	B+P+5	Cursor start and blink	The five bits determine at which sweep of a text line the cursor top is located. The B and P bits, cursor blink and blink rate. See also Fig. 21.
R11	5	Cursor end	Determines at which sweep of a text line the cursor base is located. See Fig. 21.
R12 R13	14	Display start address	Determines at which linear address in the display memory (with which character) the display should start.
R14 R15	14	Cursor location	Points out at which linear address (character position in memory) the cursor should be displayed.
R16 R17	14	Light pen hit address	Not used.

R0 – R13 are write-only registers, R14 – R15 are read/write registers and R16 – R17 are read-only registers.

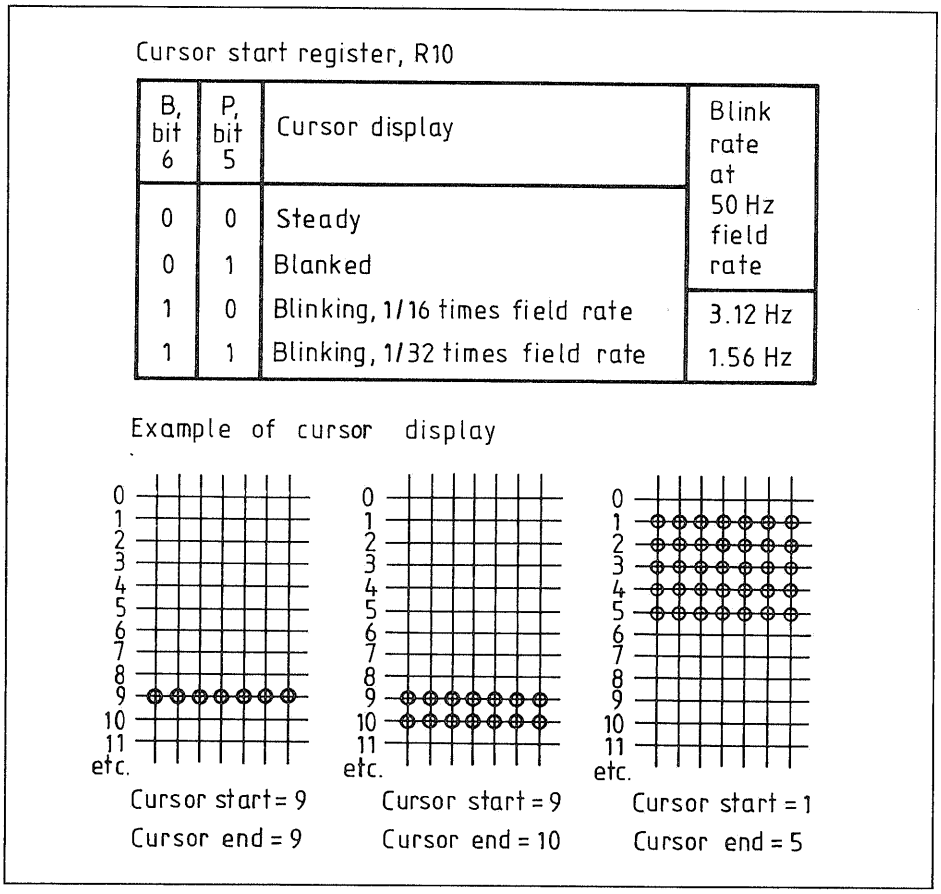


Fig. 21. Cursor control functions of R10 and R11

CRTC Control and Address Outputs

The timing of the CRTC control and address outputs, as well as the internal counters, is controlled by a clock signal (T1) fed to the clock (CLK) input of the CRTC. The clock provides two periods for each system clock cycle, i.e. one clock period for each horizontal character position. See Fig. 22.

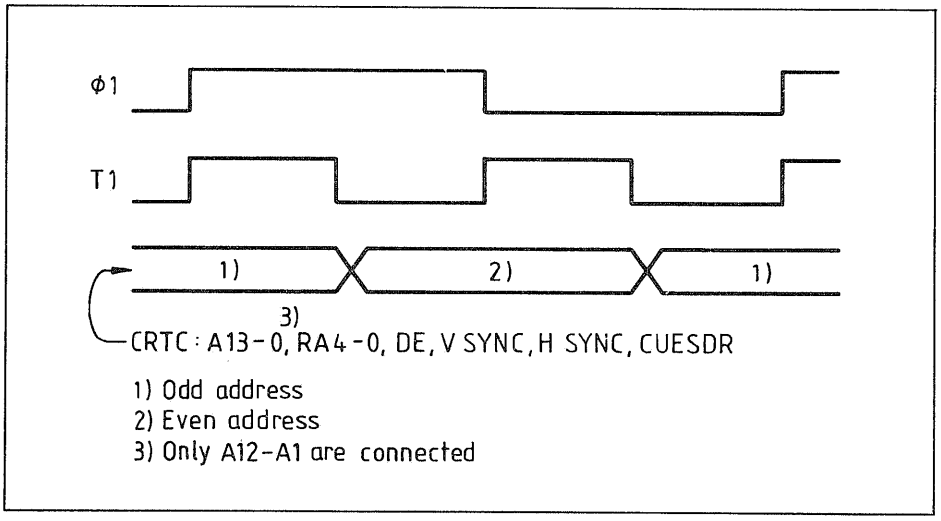


Fig. 22. CRTC timing

The following control outputs are provided:

- DE, device enable, is the master enable signal for the video generating circuitry. This signal is active during the display time of the text lines, i.e. during the part of the frame time decided by R1 and R6 but not during horizontal and vertical retrace time or when the CRTC is reset.
- CUESDR, cursor display which is active during the time when cursor dots should be produced by the video generating circuits, i.e. at applicable sweeps, decided by R10 and R11, through a specific character position, decided by R14 and R15.
- H SYNC, horizontal synchronization signal which tells the logic that the cathode ray should retrace and start the next sweep over the screen. The start and duration of the H SYNC is programmed in R2 and R3.
- V SYNC, vertical synchronization signal governing the vertical retrace of the ray that precedes the regeneration of an image on the screen. The start of the V SYNC is programmed in R7.

The CRTC provides a 5-bit raster address (RA4 – RA0) out of which four bits eventually are fed to the character generator simultaneously with the code for the actual character. For each text line the 5-bit raster address tells the actual sweep number, i.e. provides values from 0 up to the programmed value of R9. The raster address is fed to the character generator via an FPLA (field programmable logic array), where it is decoded for control signal generations and modified at the message line presentation.

The CRTC also provides a 14-bit linear address (MA13 – MA0) to the display memory, i.e. the address of the actual character to be displayed. This address is supplied by an internal linear address counter. This counter is started at a value decided by R12 and R13 and is stepped as many times as decided by R1 for each sweep. For each text line it counts through the *same* address sequence as many times as decided by the value in R9.

Then it goes on after the next H SYNC and makes a similar sequence for each displayed line i.e. as many times as decided by R6. This whole sequence starts over again after the next V SYNC, unless the programming of the CRTC has been changed.

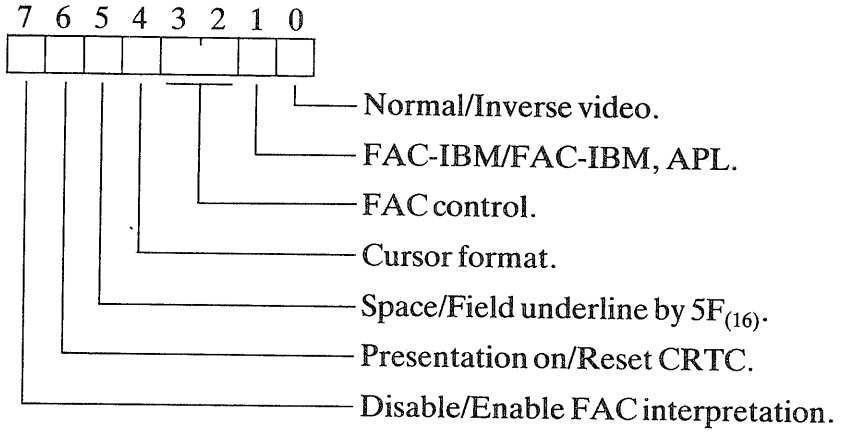
The MA0 and MA13 bits are not connected in this application. Bit 0 is logically generated in the bus/CRTC address selector as well as bit 13 – and more significant bits – since the display memory never exceeds eight kbytes.

Display Adapter Peripheral Interface Adapter, DIA PIA

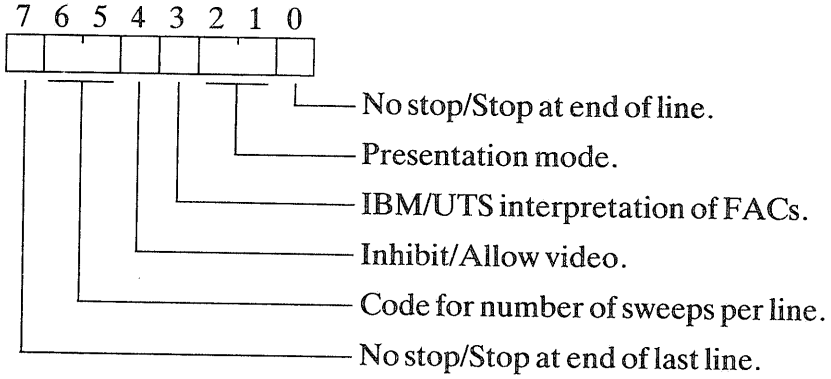
The display adapter peripheral interface adapter, DIA PIA is connected to the bus system of the microcomputer and controls several functions in the presentation logic. For definitions and a general survey, see chapter Microcomputer; Peripheral Interface Adapter, PIA.

The IRQ lines are not used and neither are the control lines on the peripheral side. The control register (CRA and CRB) contents shall be $04_{(16)}$ unless when the DIA PIA is initiated. All peripheral register bits (PA7 – 0 and PB7 – 0) are used as outputs:

DIA PIA PA7 – PA0



DIA PIA PB7 – PB0



DIA PIA Peripheral Register A

The functions of the peripheral register A outputs are further described below:

PA7 0 = Disable FAC interpretation. (Does not affect interpretation of $5F_{(16)}$, $00_{(16)}$, $20_{(16)}$ or MLFAC.)
1 = Enable FAC interpretation.

PA6 0 = Enable CRTC.
1 = Reset CRTC. PA6 must be set during or after initial programming of the CRTC.

PA5 0 = Field underline by a FAC starts in the position of the FAC at presentation mode 0 and 1 and in the following position at presentation mode 2 and 3. Character code $5F_{(16)}$ generates, if PB3 = 0, underlined space from the attribute generator in the position of the $5F_{(16)}$.
1 = Field underline by a FAC always starts in the position after the FAC. Character code $5F_{(16)}$ generates, if PB3 = 0, field underline from the following position until $00_{(16)}$, $20_{(16)}$, FAC or stop at end of line or stop at end of last line (see PB7 and PB0).

PA4 0 = Cuform 0 or covering cursor. (See Cursor.)
1 = Cuform 1 or transparent cursor. (See Cursor.)

PA3,2 FAC control. See Field Attribute Interpretation.

PA1 0 = Character codes $80_{(16)}$ – $BF_{(16)}$ interpreted as FACs (FAC-IBM). PB3 must be 0.
1 = Character codes $10XX\ XX0X_{(2)}$ i.e. $80_{(16)}$, $81_{(16)}$, $84_{(16)}$, $85_{(16)}$, $88_{(16)}$ ... $BC_{(16)}$ and $BD_{(16)}$ interpreted as FACs (FAC-IBM, APL). PB3 must be 0.

PA0 0 = Normal video i.e. bright text on dark background. Only text in fields inverted by a field attribute character or a character at the cursor position (Cuform 1) is dark on a bright background.
1 = Inverse video. The whole screen light is inverted.

DIA PIA Peripheral Register B

The functions of the peripheral register B outputs are further described below:

PB7 0 = No stop at end of last line (message line excluded) if PB0 = 0. Exception: Field underline due to $5F_{(16)}$ will wrap around only if at least one screen position contains a FAC.
1 = Stop at end of last line, i.e. attributes do not wrap around from the last line (message line excluded) to the first line of the screen.

PB6,5 The attribute generator must be informed about the CRTC programming:
00 = 16 sweeps in character cell.
01 = 12 sweeps in character cell.
10 = 9 sweeps in character cell.
11 = 22 sweeps in character cell.
(If 01 or 10 field underline is not generated.)

PB4 0 = Inhibit video. The video signal(s) to the cathode ray tube unit is (are) blocked.
1 = Do not inhibit video.

PB3 0 = IBM interpretation of FACs. See also PA1.
1 = UTS interpretation of FACs. Character codes $80_{(16)}$–$FF_{(16)}$ interpreted as FACs (PA1 has no effect).

PB2,1 00 = Presentation mode 3.
01 = Presentation mode 2.
10 = Presentation mode 1.
11 = Presentation mode 0.
See Display Memory Addressing.

PB0 0 = No stop at end of line. A FAC is valid until the next FAC or only to the stop at end of last line (message line excluded) if PB7 = 1 or 40 ms if no more FACs are encountered. The same is valid for field underline by $5F_{(16)}$ but it will also end at the codes $00_{(16)}$ and $20_{(16)}$ and wrap around only if any screen position contains a FAC even if PB7 = 0.
1 = Stop at end of line. A FAC is valid until the next FAC or until end of line. The same is valid for field underline by $5F_{(16)}$ but it will also stop at the codes $00_{(16)}$ and $20_{(16)}$.

Field Attribute Interpretation

Three types of field attribute interpretations are possible:

- IBM interpretation. (FAC-IBM and IBM/APL.)
- UTS interpretation. (FAC-UTS.)
- Message line field attribute character, MLFAC, interpretation.

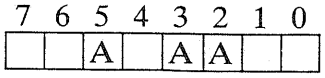
The effect of a FAC starts in the position of the FAC and ends just before the next FAC or stop at end of line or end of last line (but the message line) or after 40 ms if no more FACs or stops are encountered. There are, however, exceptions for underline functions. These exceptions are pointed out below.

In order to get the interpretations listed below PA7, PA6 and PB4 must be 1, 0 and 1 respectively. (PAn and PBm denotes outputs from DIA PIA peripheral register A and B.) The CRTC and DIA PIA must also be programmed for 16 or 22 sweeps in order to get field underline.

FAC-IBM and IBM/APL

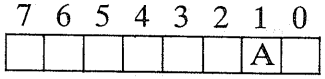
FAC	Comments
7 6 5 4 3 2 1 0 1 0	Interpreted as FAC-IBM when PA1 = 0 and PB3 = 0.
7 6 5 4 3 2 1 0 1 0 0	Interpreted as FAC-IBM/APL when PA1 = 1 and PB3 = 0.
7 6 5 4 3 2 1 0 A A A A	Bits 5, 4, 1 and 0 may be interpreted by the software, e.g. numeric field.
7 6 5 4 3 2 1 0 A A	Bits 3 and 2 are interpreted in the following way by the hardware in DU 4111:

Bit: 3 2	Interpretation
0 X	Normal light intensity.
1 0	High light intensity.
1 1	No presentation.



Bits 5, 3 and 2 are interpreted in the following way by the hardware in DU 4112:

Bit: 5 3 2	Base colour switch: Monochrome	Base colour
0 0 X	Green	Green
0 1 0	White	Red
1 0 X	Green	Blue
1 1 0	White	White
X 1 1	No presentation	No presentation



Bit 1 may be interpreted by the software and controls the presentation hardware if any of PA3 or PA2 is set. If bit 1 is set the following functions will be obtained depending on PA3 and PA2:

PA3	PA2	Function
0	1	Field (light) inversion
1	0	Field underline
1	1	Flashing field (1 Hz)

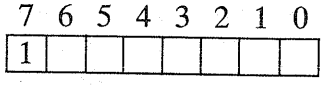
Note that these functions are not available at FAC-IBM/APL. (Bit 1 is always 0.) The field underline starts in the position after the FAC and ends before the next FAC or stop at end of line or stop at end of last line. If presentation mode 0 or 1 and PA5 = 0 is selected the underline, however, starts in the same position as the FAC.

Field underline is also generated by $5F_{(16)}$ if PA5 = 1. The field starts at the position after $5F_{(16)}$ and ends before next $00_{(16)}$ or $20_{(16)}$ codes, FAC or stop at end of line or stop at end of last line. (Regarding stops, see Display Adapter Peripheral Interface Adapter, DIA PIA.)

$FF_{(16)}$ functions as message line field attribute character, MLFAC. The message line is always blue in DU 4112.

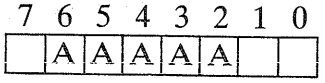
FAC-UTS

FAC



Comments

Interpreted as FAC-UTS when PB3 = 1.



Bits 6 – 2 are interpreted by software only, e.g. numeric field.

7	6	5	4	3	2	1	0
						A	A

Interpretation of bits 1–0:

0 0 Normal light intensity (= IBM high light intensity) in DU 4111. Green presentation in DU 4112.

0 1 No presentation

1 0 Low light intensity (= IBM normal light intensity) in DU 4111. White presentation in DU 4112.

1 1 The light intensity will be normal in DU 4111 and the presentation green in DU 4112 and if PA3 or PA2 or both are set the following will be obtained:

PA3	PA2	Function
0	1	Field (light) inversion
1	0	Field underline
1	1	Flashing field (1 Hz)

The field underline starts in the position after the FAC and ends before the next FAC or stop at end of line or stop at end of last line. If presentation mode 0 or 1 and PA5 = 0 is selected the underline, however, starts in the same position as the FAC.

Note that the codes $5F_{(16)}$, $00_{(16)}$ and $20_{(16)}$ function as ordinary characters and do not generate field underline or end of field underline.

$7F_{(16)}$ functions as message line field attribute character, MLFAC. The message line is always blue in DU 4112.

Cathode Ray Tube Unit, CRU, of DU 4111

Please refer to circuit diagram E34111 00X0 which shows the circuitry of the cathode ray tube board (CRB), the cathode ray tube (CRT) and interconnections.

Deflection Circuits

The horizontal deflection is governed by IC 111, which contains an oscillator, free-running at 21.3 kHz, and a phase locked loop circuit for provision of a horizontal deflection signal in phase with the incoming H SYNC signal. Furthermore, the horizontal deflection output stage generates the tube voltages by help of the transformer T153, i.e. the high voltages for the acceleration anodes (17 kV, 930 V) and the heating current for the cathode.

The vertical deflection is governed by IC 131. It contains an oscillator that should be free-running at a lower frequency (47 Hz) than the incoming V SYNC signal (50 Hz, regeneration frequency). It also contains an output stage providing the vertical deflection current.

Focusing

The voltage for the focus grid has two components; a static one, adjustable by help of a potentiometer, and a dynamic one, derived from the horizontal deflection signal. The parabolic dynamic focusing signal is needed as the path of the beam, from cathode to screen, is longer when directed towards the left or right edge of the screen than when directed towards the centre.

Brightness and Contrast Control

The brightness and contrast control circuitry and the video amplifier located on the small part of the CRB produce an amplified video signal for the cathode. The amplitude of this signal depends on the setting of the external Brightness potentiometer and, for fields with high intensity, on the setting of the external Contrast potentiometer. Furthermore, there is an internal potentiometer, Black level, that governs the amplitude of the beam by affecting the control grid.

Protective Circuitry

To prevent beam current on the acceleration anode from becoming too high, there is circuitry on the main part of the CRB that will limit it by decreasing the voltage on the control grid. The small part of CRB contains spark gaps that will direct flash-overs between the extra high tension anode and the other electrodes the shortest way to ground and resistors that will diminish the flash-over currents.

Adjustments

The deflection unit on the tube neck holds a pair of turnable magnets for centering of the picture and a number of adjustable magnets for raster correction (correction of pincushion-formed picture). On top of the main part of the CRB there are a number of potentiometers which are used for initial trimming of the CRU. There are also three more adjustment points lower on the CRB. Specific trimming information is found in the Installation and Maintenance Manual.

Cathode Ray Tube Unit, CRU, of DU 4112

Please refer to circuit diagram E34112 2020. The diagram shows the circuitry, the cathode ray tube and the interconnections.

Deflection Circuits

The horizontal deflection is governed by an integrated circuit (IC 401), which contains an automatic frequency control circuit (AFC) a horizontal oscillator and an X-ray protection circuit. The H SYNC signal is fed to pin 17 and a feed-back pulse from a flyback transformer (T551) to pin 15. The horizontal drive signal is output from pin 12 unless inhibited by the X-ray protection circuit, which is controlled via a shut down circuit by

the heater voltage (from T551). The horizontal drive signal is amplified by a transistor (Q502) and a transformer (T501) in order to generate a strong drive current for the switching horizontal output transistor, which drives the primary winding of the flyback transformer and the horizontal deflection coil. The secondary windings of the flyback transformer supplies the internal voltages to the CRU.

The V SYNC signal is inverted (in Q401) and fed to (pin 8 of) the vertical oscillator in the integrated circuit (IC 401). This circuit governs the vertical deflection and the output (pin 6) is amplified (in Q402, Q403, Q404, Q406, Q407, Q408) and fed to the vertical deflection coil.

The vertical deflection voltage is tapped for a pincushion correction signal, which via a transistor (Q751) and a saturable reactor (T751) affects the horizontal deflection.

Video Amplifiers and Brightness Control

The video circuitry has three identical circuits for red, green and blue video. Each one of the three video signals is received by a nand gate (IC 1301) and is fed further to an open-collector nand gate (IC 1302), which output is fed with a variable voltage via a transistor (Q1311). The voltage level is a function of the external Brightness potentiometer position. The signal is then inverted (by Q301/Q302/Q303) and amplified (by Q304/Q305/Q306 and, in later versions, by Q308, Q311/Q309, Q312/Q310, Q313) before it is fed to the cathode of the tube. Furthermore there is a clamping circuit (around Q307) for adjustments of the cut-off levels of the three different cathodes.

Adjustments

There are six magnets on the neck of the tube and a number of trimming potentiometers for adjustments. These are described in the Installation and Maintenance Manual under Service Manual CRU E34112 0020, FTD 82100246.

Keyboard Asynchronous Communication Adapter, KB ACIA

ACIA Functions

For definitions and general survey, see Asynchronous Communication Adapter, ACIA under Internal Communication with Printers in the Communication chapter.

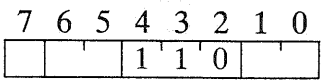
The MPU interfaces the ACIA by the 8-bit internal data bus, the Read/Write line, the address bus and one interrupt line. Address bit 0 is directly fed to the register select input of the ACIA and the inversion of bit 2 to one chip select input. The two other chip select inputs of the ACIA are supplied by the address decoder. The timing on the enable input is supplied by Ø2. The IRQ line is fed to the interrupt register as interrupt 2.

Both the transmitter and receiver clocks are derived from Ø2 in a divide-by-14 counter and the clock rate is thus 76.07 kHz. (The RTS output is not used nor are the CTS and DCD inputs which are hard-wired low.)

The received and transmitted data lines are connected to each other on the outside of the line buffers, thus creating a bidirectional line. (The ACIA can thus receive the same data as it transmitts.)

ACIA Programming

The ACIA control register is usually programmed in the following way:



10 during operation. The external clock (76.07 kHz) is divided by 64 and the bit transfer rate is thus 1188 Hz.
11 during reset periods (Master reset).

The word length is eight bits of data, one parity bit and one stop bit.

01 during transmitting. Interrupts at transmitted data register empty are enabled.
X0 during reception from the keyboard or when not communicating with the keyboard.

0 when not communicating with the keyboard. Receiver interrupts are disabled.
1 during communication with the keyboard. Interrupts at received data register full and overrun are enabled. (The receiver interrupt may be used to check when the last bit has been shifted out at transmission.)

Two-wire Interface

The two-wire interface can work in MPU mode or DMA (direct memory access) mode. The MPU mode is used when the direct memory access controller, DMAC, and the advanced data link controller, ADLC, circuits are initialized or supervised by the MPU. DMA mode is used at DMA transfers.

The main circuits and signals of the two-wire interface are shown in Fig. 23. Please refer to the figure when reading the following text.

Direct Memory Access Controller, DMAC

The DMAC handles DMA transfers between the basic read/write memory and the ADLC (see below). For definitions and a general survey of the DMAC, see the Microcomputer chapter.

The DMAC is in MPU mode addressed by the DMACS signal from the address decoder and five address bits. The R/W line is also fed to the DMAC. The bus buffers will allow the five address bits and R/W to pass in direction to the DMAC in MPU mode.

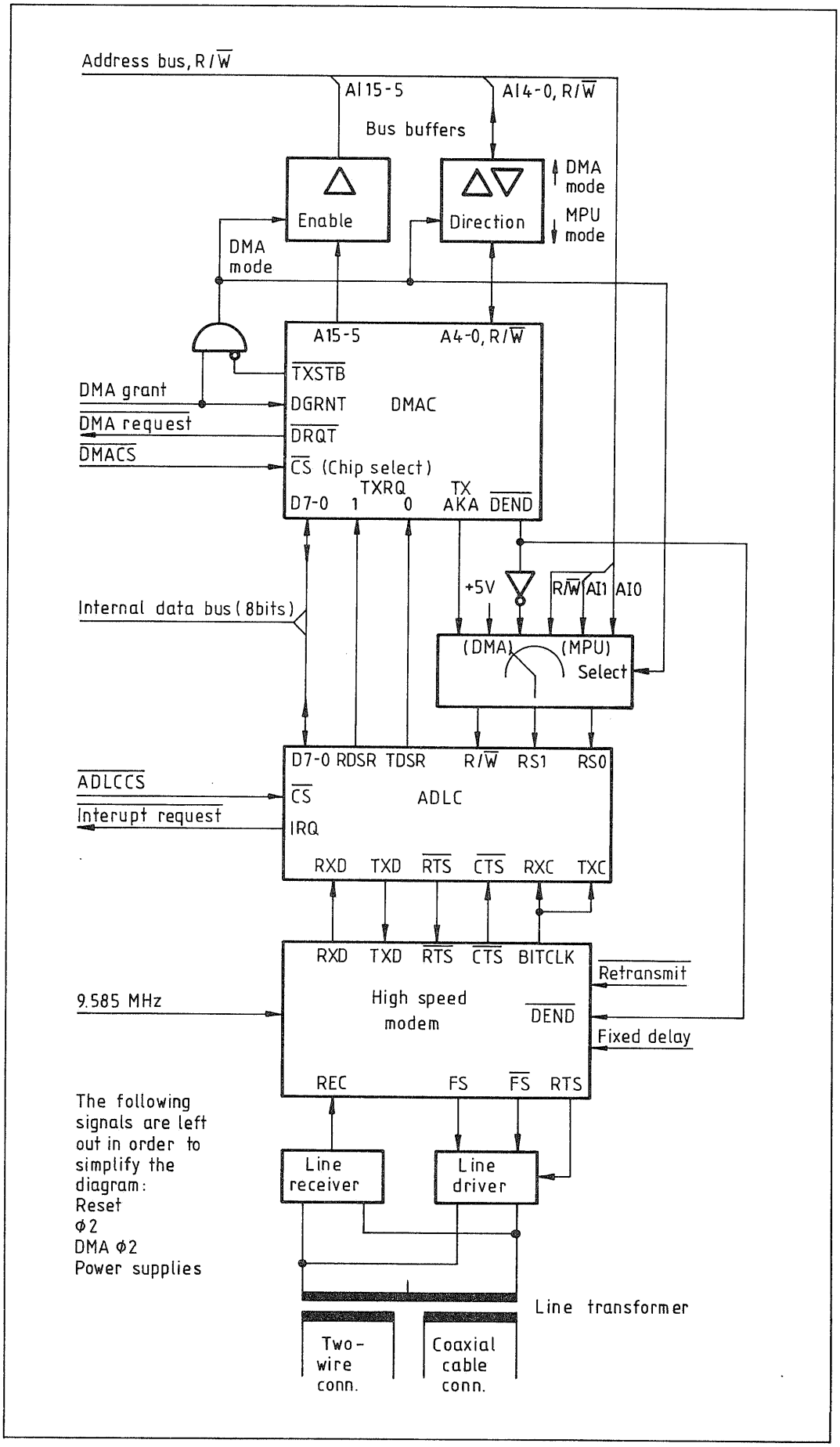


Fig. 23. Principles of two-wire interface

The DMAC will at a service request from the ADLC (RDSR or TDSR active) make a DMA request and when this is accepted (DMA grant active) go to DMA mode (with TXSTB active at DMA cycle 2). The DMAC then provides a memory address from an internal address counter, a R/W signal (to the ordinary R/W line) and control signals to the ADLC (TXAKA, producing a read/write signal to the ADLC and DEND at the last byte of a transferred block). The bus buffers will allow all fifteen address bits and R/W to pass in direction from the DMAC in DMA mode.

The DMAC is connected for TSC steal mode and two channel mode. Channel 0 is used for transmitting and channel 1 for receiving.

Advanced Data Link Controller, ADLC

The ADLC converts parallel data from the internal data bus to serial data (TXD) for the high speed modem or serial data from the high speed modem (RXD) to parallel data for the data bus. For definitions and a general survey of the ADLC, see the Communication chapter.

The ADLC is programmed to generate a Request to send (RTS) signal to the high speed modem and to interrupt the MPU. The two-wire interrupt thus originate from the ADLC – not the DMAC. (The data carrier detect input, DCD, is hard-wired active. The DTR/LOC and FLAGDET outputs are not used.)

It is possible to make transfers of two-wire data via the MPU but up to now only DMA transfers are used.

The ADLC is in MPU mode addressed by the ADLCCS signal from the address decoder and two address bits, AI1 – 0. The R/W line is also fed to the ADLC. The ADLCCS signal from the address decoder is also activated at DMA cycle 2 but the R/W and RS0 inputs are in DMA mode fed from the DMAC and the RS1 is then fed from a hard-wired high. The addressing is summarized below:

Address conditions							Register addressed and read/write function
MPU mode			DMA mode			Address control bit[1]	
R/W (R/W)	A1 (RS1)	A0 (RS0)	TXAKA (R/W)	+5 V (RS1)	DEND (RS0)		
0	0	0		[2]		X	Write control register 1
0	0	1		[2]		0	Write control register 2
0	0	1		[2]		1	Write control register 3
0	1	0	0	1	0	X	Write transmitter FIFO[3]
0	1	1	0	1	1	0	Write transmitter FIFO[4]
0	1	1	0	1	1	1	Write control register 4
1	0	0		[2]		X	Read status register 1
1	0	1		[2]		X	Read status register 2
1	1	X	1	1	X	X	Read receiver FIFO

[1] Control register 1 bit 0. [2] 0 is not possible. [3] Frame continue. [4] Frame terminate.

High Speed Modem

The high speed digital modem consists of one chip, which contains a modulator, a demodulator, a frequency divide and phase correction logic, and a delay circuit for Clear to send. See Fig. 24.

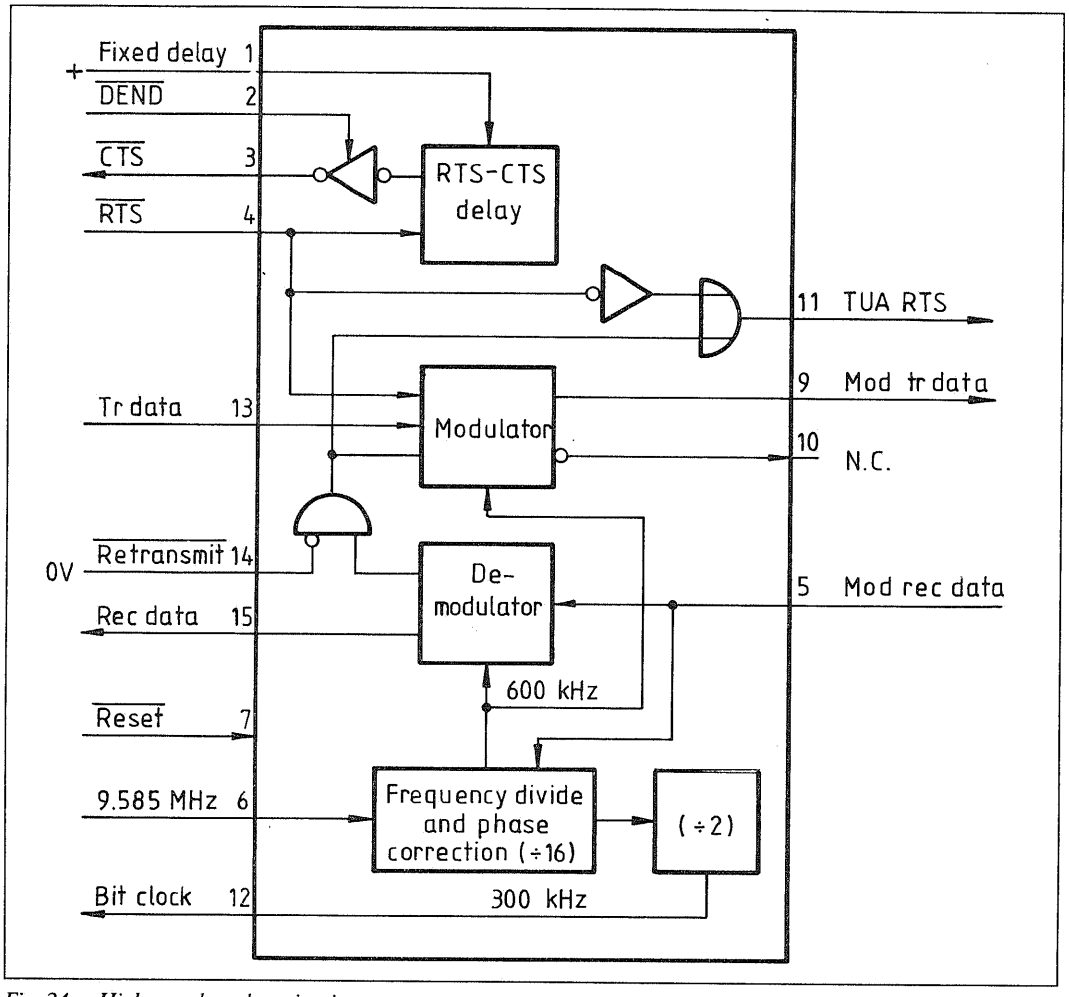


Fig. 24. High speed modem circuit

Data to be transmitted (TXD, from the ADLC) is presented in serial format to the modulator. The modulator converts the signal to a frequency shifted signal (FS) which then is fed to the two-wire if Request to send (RTS) is active.

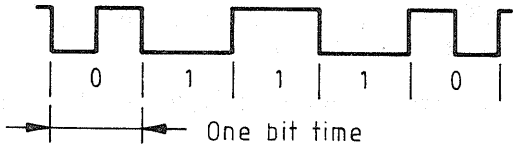
The frequency shifted data transferred from the two-wire (REC) is fed to the demodulator and to the frequency divide and phase correction logic. An internal strobe from the latter logic shifts the frequency shifted received data into the demodulator and generates a bit clock (BITCLK), fed to the ADLC. The demodulator converts the received data to de-modulated valid data (RXD, fed to the ADLC). (Received data is not, as Retransmit is hard-wired high, fed to the modulator and retransmitted.)

The modulation method, clock generation and phase correction are described in the Communication chapter.

The Request to send output (RTS) follows (but inverted) the Request to send input (RTS). (It is also a function of received data in the not used retransmit mode.)

Clear to send (CTS) follows the Request to send input (RTS) with a delay of 32 (controlled by strapping of the fixed delay input) bit clock periods. This delay will enable the modem in the receiving unit (e.g. in a CPR) to synchronize before the transmission is started. A frequency corresponding to the state of the transmitted data input (TXD) will be sent on the FS lines during the delay. An active DEND signal (at the end of a transmission sequence) from the DMAC will deactivate Clear to send.

Frequency shifted received data is validated, i.e. each bit must consist of an LH, HL, LL or HH and there must be a shift of polarity between the bits in order to be accepted:



Received data (RXD) is always high when received data is invalid or before the modem has synchronized.

Line Interface

The high speed modem is connected to a line transformer via a line driver and a line receiver. The line transformer has separate windings for a two-wire and a coaxial cable connection. (See Fig. 23.)

Transmitting

To initiate a block transmission, using DMA, the program must prepare itself to answer an interrupt from the ADLC (interrupt 4) when the whole block (frame) is transmitted. The DMAC channel 0 address register, byte count register and control register should also be programmed with:

- Start address of DMA, i.e. the memory address of the first byte (except flag) to be transmitted on the two-wire connection.
- Number of memory bytes to be transmitted.
- Order to count up or down when addressing the memory.
- Order to produce the read state on the R/W line when addressing the memory.
- TSC steal mode.
- Set Enable TXRQ 0 in the DMAC priority control register in order to make the DMAC react to TDSR (Transmit data service request) from the ADLC.

The ADLC must be programmed for:

- Correct word length (8 bits).
- NRZ data format.
- Non-loop mode.
- Transmitter DMA mode etc.

The transmitting section of the ADLC is released and the Request to send (RTS) control bit is set by the program. After a delay Clear to send (CTS) is signalled back to the ADLC, and thus the Transmitter data service request (TDSR) output from the ADLC is no longer inhibited.

The following will happen as an answer to a TDSR signal:

- A DMA request signal (DRQT) will be sent to the timing logic.
- This logic will answer with a DMA grant signal (DGRNT), defining the DMA cycles 1 and 2.
- The DMAC will then produce a low level on the transfer acknowledge A (TXAKA) output as channel 0 is used. This will ensure a write strobe for the ADLC when;
- The DMA mode signal is activated by the Transfer strobe (TXSTB).

In the same time the ADLC is enabled, its transmit data register is addressed and a memory cell is addressed by the DMAC as the drivers for the address lines from the DMAC are enabled.

After a direct memory access the byte is shifted to the high speed modem. Thus it takes some time before the transmit data register is again available. The microcomputer is consequently undisturbed by any DMA request during this time.

When the DMA transfers (consisting of the number of bytes defined in the DMAC byte count register) are being completed, i.e. when the last byte is being fetched from the memory, the DEND output of the DMAC is activated. This will result in:

- A termination of the frame (write into the "transmit terminate FIFO" register; the ADLC adds CRCC and flag).
- A deactivation of the Clear to send (CTS) signal. The ADLC will react by not signalling any more TDSR.

  The non-CTS signal causes an interrupt. As an answer, the program resets the RTS signal, clears status bits in the ADLC and prepares the DMAC and the ADLC for reception.

Receiving

Preparatory programming of the DMAC and the ADLC for receiving operations resembles the one described in the Transmitting paragraph. Furthermore, the program sees to it that an interrupt will be answered by a program check of receiver status etc.

When bytes following a leading flag are received, the ADLC sends receiver data service request (RDSR) to the DMAC. These bytes will then be stored in the memory according to DMAC channel 1 programming. The DMA transfer is terminated as a response to an interrupt caused by error (framing error, abort received or receiver overrun) or by frame valid status. The program then resets the ADLC and takes care of the received data on interrupt 0 level. Note that Request to send must of course be reset during reception.

Appendix 1

I/O Addresses, $F7FF_{(16)} - F700_{(16)}$

(The first two figures, $F7_{(16)}$, are omitted in the address column below. R/W = W means write, R/W = R means read and R/W = R/W means read or write as seen from the MPU.)

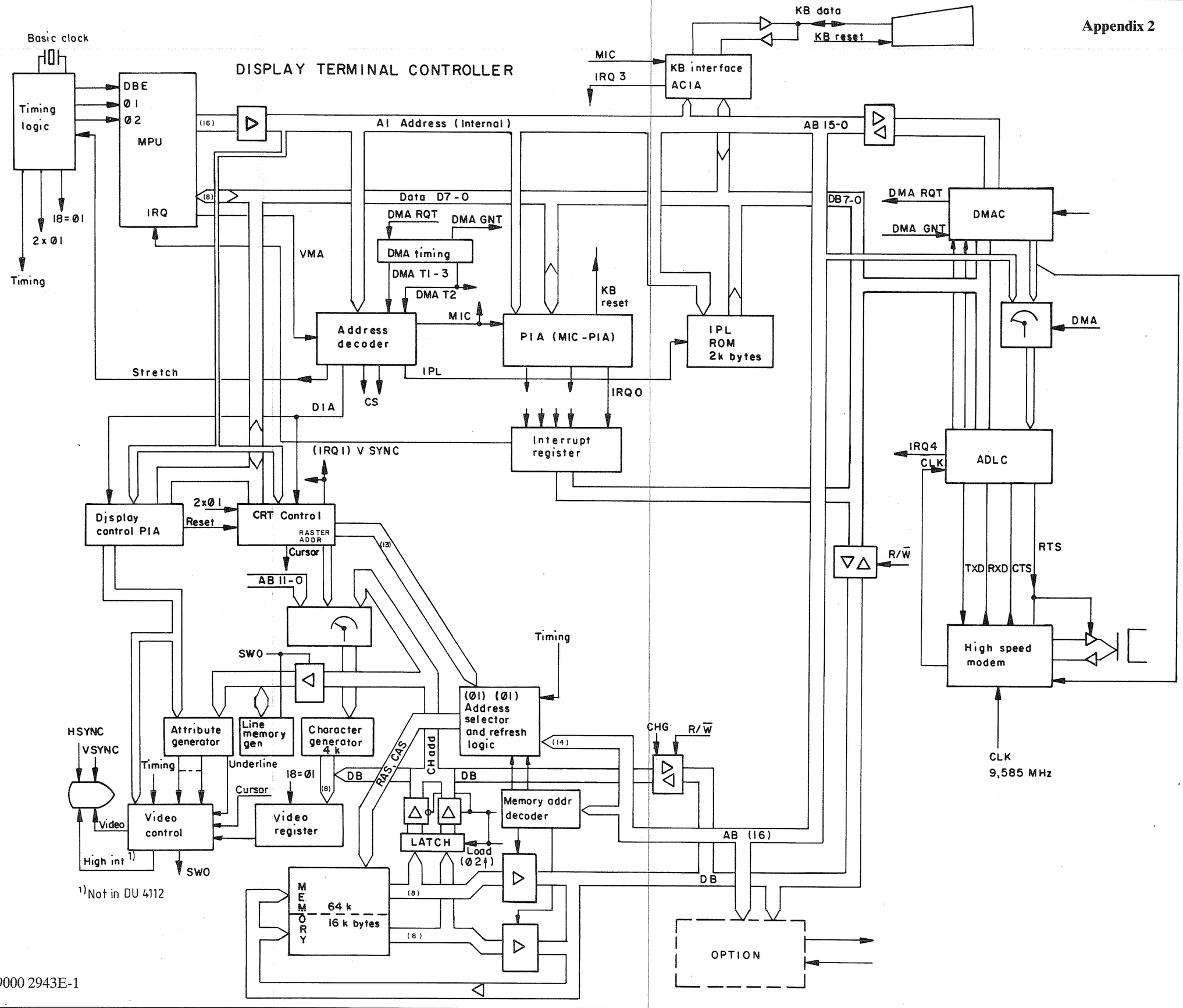
Address (Hex)	R/W	Other condition	Addressed unit and register
			MCP
FF	W		STORE (bit 7), CCR, ACCB, ACCA, XHI, XLO, PCHI and PCLO switches
FE	W		Data 7–0 switches
FD	W		Address 15–8 switches
FC	W		Address 7–0 switches
FB	R		A, B, C, D, NMI, IRQ, VMA and R/W LEDs
FA	R		Data 7–0 LEDs
F9	R		Address 15–8 LEDs
F8	R		Address 7–0 LEDs
F7–F0	R		**Interrupt register**
EF–E0			**Not used**
			CRTC
DF	→	→	= D9
DE	W		= D8
DD	→	→	= D9
DC	W		= D8
DB	→	→	= D9
DA	W		= D8
D9	W	AR = $00_{(16)}$	Character times per sweep
D9	W	AR = $01_{(16)}$	Characters per line
D9	W	AR = $02_{(16)}$	H SYNC position
D9	W	AR = $03_{(16)}$	H SYNC width
D9	W	AR = $04_{(16)}$	Lines total
D9	W	AR = $05_{(16)}$	Vertical total adjust
D9	W	AR = $06_{(16)}$	Lines displayed
D9	W	AR = $07_{(16)}$	V SYNC position
D9	W	AR = $08_{(16)}$	Interlace control ($00_{(16)}$ used)
D9	W	AR = $09_{(16)}$	Maximum raster address
D9	W	AR = $0A_{(16)}$	Cursor start and blink
D9	W	AR = $0B_{(16)}$	Cursor end
D9	W	AR = $0C_{(16)}$	Display start address (MSB)
D9	W	AR = $0D_{(16)}$	Display start address (LSB)
D9	R/W	AR = $0E_{(16)}$	Cursor location (MSB)
D9	R/W	AR = $0F_{(16)}$	Cursor location (LSB)
D9	R	AR = $10_{(16)}$	Light pen hit address (MSB, not used)
D9	R	AR = $11_{(16)}$	Light pen hit address (LSB, not used)
D8	W		Address register (called AR above)
			DIA PIA
D7	R/W		= D3
D6	R/W	→	= D2
D5	R/W		= D1
D4	R/W	→	= D0
D3	R/W		Control register B (CRB)
D2	R/W	CRB2 = 0	Data direction register B
D2	R/W	CRB2 = 1	Peripheral register B
D1	R/W		Control register A (CRA)
D0	R/W	CRA2 = 0	Data direction register A
D0	R/W	CRA2 = 1	Peripheral register A

Address (Hex)	R/W	Other condition	Addressed unit and register
CF – C8			**Not used**
			MIC PIA
C7	R/W		Control register B (CRB)
C6	R/W	CRB2 = 0	Data direction register B
C6	R/W	CRB2 = 1	Peripheral register B
C5	R/W		Control register A (CRA)
C4	R/W	CRA2 = 0	Data direction register A
C4	R/W	CRA2 = 1	Peripheral register A
			KB ACIA
C3	→		= C1
C2	→		= C0
C1	W		Transmitted data register
C1	R		Received data register
C0	W		Control register
C0	R		Status register
BF – A8			**Not used**
			ACA ACIA (Modem interface)
A7	→		= A1
A6	→		= A0
A5	→		= A1
A4	→		= A0
A3	→		= A1
A2	→		= A0
A1	W		Transmitted data register
A1	R		Received data register
A0	W		Control register
A0	R		Status register
			ACA PTM (Modem interface)
9F	W		Timer 3 LSB latches
9F	R		LSB buffer (timer 3)
9E	W		MSB buffer (timer 3)
9E	R		Timer 3 counter MSB
9D	W		Timer 2 LSB latches
9D	R		LSB buffer (timer 2)
9C	W		MSB buffer (timer 2)
9C	R		Timer 2 counter MSB
9B	W		Timer 1 LSB latches
9B	R		LSB buffer (timer 1)
9A	W		MSB buffer (timer 1)
9A	R		Timer 1 counter MSB
99	W		Control register 2 (CR2)
99	R		Status register
98	W	CR2:0 = 0	Control register 3
98	W	CR2:0 = 1	Control register 1
97 – 68			**Not used**
			ACA ACIA (Printer interface)
67	→		= 61
66	→		= 60
65	→		= 61
64	→		= 60
63	→		= 61
62	→		= 60
61	W		Transmitted data register
61	R		Received data register
60	W		Control register
60	R		Status register

Address (Hex)	R/W	Other condition	Addressed unit and register
			ACA PTM (Printer interface)
5F	W		Timer 3 LSB latches
5F	R		LSB buffer (timer 3)
5E	W		MSB buffer (timer 3)
5E	R		Timer 3 counter MSB
5D	W		Timer 2 LSB latches
5D	R		LSB buffer (timer 2)
5C	W		MSB buffer (timer 2)
5C	R		Timer 2 counter MSB
5B	W		Timer 1 LSB latches
5B	R		LSB buffer (timer 1)
5A	W		MSB buffer (timer 1)
5A	R		Timer 1 counter MSB
59	W		Control register 2 (CR2)
59	R		Status register
58	W	CR2:0 = 0	Control register 3
58	W	CR2:0 = 1	Control register 1
			ACA PIA (Printer interface)
57	R/W		= 53
56	R/W	→	= 52
55	R/W		= 51
54	R/W	→	= 50
53	R/W		Control register B (CRB)
52	R/W	CRB2 = 0	Data direction register B
52	R/W	CRB2 = 1	Peripheral register B
51	R/W		Control register A
50	R/W	CRB2 = 0	Data direction register A
50	R/W	CRB2 = 1	Peripheral register A
4F–48			**Not used**
47–40	R		**SS3 address**
3F–28			**Not used**
			ADLC (Can also be addressed by the DMAC at DMA)
27	→	→	= 23
26	→		= 22
25	→	→	= 21
24	→		= 20
23	W	CR1:0 = 0	Transmitter FIFO (terminate)
23	W	CR1:0 = 1	Control register 4
23	R		Receiver FIFO
22	W		Transmitter FIFO (continue)
22	R		Receiver FIFO
21	W	CR1:0 = 0	Control register 2
21	W	CR1:0 = 1	Control register 3
21	R		Status register 2
20	W		Control register 1 (CR1)
20	R		Status register 1

Address (Hex)	R/W	Other condition	Addressed unit and register
1F–17			**Not used**
			DMAC
16	R/W		Data chain
15	R/W		Interrupt control
14	R/W		Priority control
13	R/W		Channel 3 control
12	R/W		Channel 2 control
11	R/W		Channel 1 control
10	R/W		Channel 0 control
0F	R/W		Byte count, channel 3 (LSB)
0E	R/W		Byte count, channel 3 (MSB)
0D	R/W		Address, channel 3 (LSB)
0C	R/W		Address, channel 3 (MSB)
0B	R/W		Byte count, channel 2 (LSB)
0A	R/W		Byte count, channel 2 (MSB)
09	R/W		Address, channel 2 (LSB)
08	R/W		Address, channel 2 (MSB)
07	R/W		Byte count, channel 1 (LSB)
06	R/W		Byte count, channel 1 (MSB)
05	R/W		Address, channel 1 (LSB)
04	R/W		Address, channel 1 (MSB)
03	R/W		Byte count, channel 0 (LSB)
02	R/W		Byte count, channel 0 (MSB)
01	R/W		Address, channel 0 (LSB)
00	R/W		Address, channel 0 (MSB)

DISPLAY TERMINAL CONTROLLER



1) Not in DU 4112

E9000 2943E-1

Appendix 3

Memory Address Handling in DTC-B and DTC-C

Memory Address Decoder

Inputs											"Feedback"		Outputs		Comments
MPU/DMA address bus						$\overline{RWM}$	$BUS_{2)}$	MPUM	4/$\overline{8}$	PDB	$\overline{DISPM}$	$\overline{16}$	Signal	Act.	
15	14	13	12	1	0	1)	ENAB	3)	4)	5)	addr				
L	H	H	H			L	H			H			DISPM addr	L	7XXX } Connects character generator to MPU/DMA bus
H	H	H	L			L	H			L					EXXX }
H	H	H	L		H	L		L		L	H		16	L	EXXX
L	H	H	H		H	L		L	H	H	H				7XXX } Address mode 0
L	H	H			H	L		L	L	H	H				6/7XXX
H	H	H	L	H		L		H		L	H				EXXX
L	H	H	H	H		L		H	H	H	H				7XXX } Address mode 1
L	H	H		H		L		H	L	H	H				6/7XXX
						L					H	H	64	L	Not Char. gen or 16k
H	H	H	L					L		L			SWITCH	L	EXXX } Display memory
L	H	H	H					L	H	H					7XXX } area addressed at
L	H	H						L	L	H					6/7XXX } address mode 0

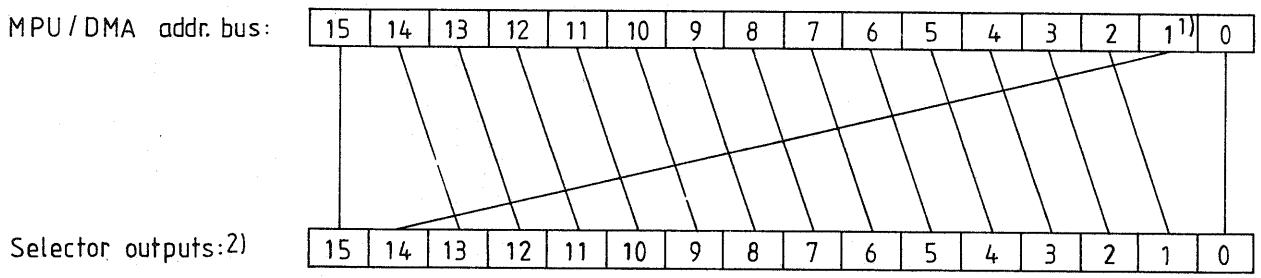
1) From address decoder
2) From MIC PIA PB4: 1 = Enable character generator to MPU/DMA bus
3) From MIC PIA PB5: 0 = Address mode 0, 1 = Address mode 1
4) From MIC PIA PB3: 0 = 8 kbytes, 1 = 4 kbytes display memory
5) From MIC PIA PB7: 0 = High end, 1 = Normal display memory

MPU/DMA Memory Addressing

Inputs																			Outputs		Comments
MPU/DMA address bus																Bus addr	$\overline{SWITCH}$	PDB	4/$\overline{8}$	Bus bit	Level	Addr. mode
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
L																L				15	L	1/0	
														L		L	H			14	L	1	
															L	L	L			14	L	0	
	L															L				13	L	1/0	
		L														L				12	L	1/0	
			L													L	H			11	L	1	
			L													L	L		H	11	L	0	4k
				L												L	L	H	L	11	L	0	8k, normal
				L												L	H			10	L	1	
					L											L	L			10	L	0	
					L											L	H			9	L	1	
						L										L	L			9	L	0	
						L										L	H			8	L	1	
							L									L	L			8	L	0	
							L									L	H			7	L	1	
								L								L	L			7	L	0	
								L								L	H			6	L	1	
									L							L	L			6	L	0	
									L							L	H			5	L	1	
										L						L	L			5	L	0	
										L						L	H			4	L	1	
											L					L	L			4	L	0	
											L					L	H			3	L	1	
												L				L	L			3	L	0	
												L				L	H			2	L	1	
													L			L	L			2	L	0	
													L			L	H			1	L	1	
														L		L	L			1	L	0	
															L	L	H			0	L	1	
				L												L	L		H	0	L	0	4k
			L													L	L	H	L	0	L	0	8k, normal

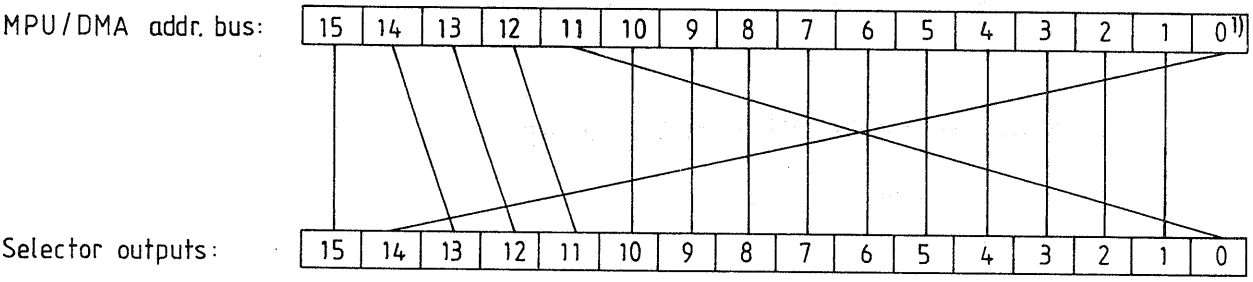
Summary of MPU/DMA Memory Addressing

Address Mode 1[3], 4 and 8 kbytes Display Memory



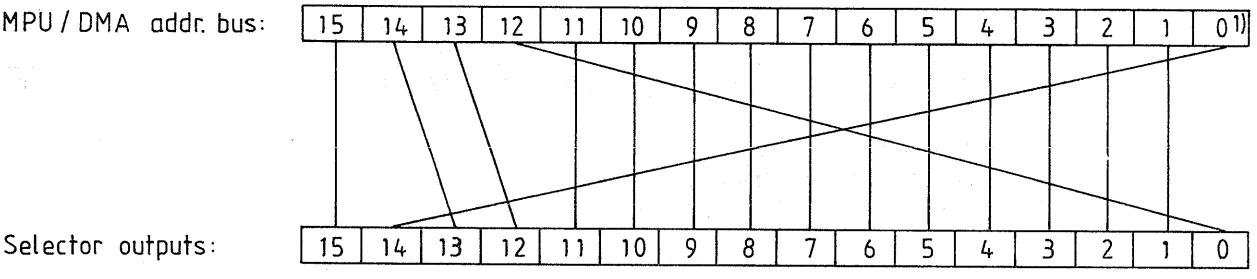
[1] Defines memory block within display memory area; low = 64 k block, high = 16 k block.
[2] Numbered as bus lines on logic diagram.
[3] The same selector functions are always used outside of display memory area.

Address Mode 0, 4 kbytes Display Memory



[1] Defines memory block; low = 64 k block, high = 16 k block.

Address Mode 0, 8 kbytes Display Memory



[1] Defines memory block; low = 64 k block, high = 16 k block.

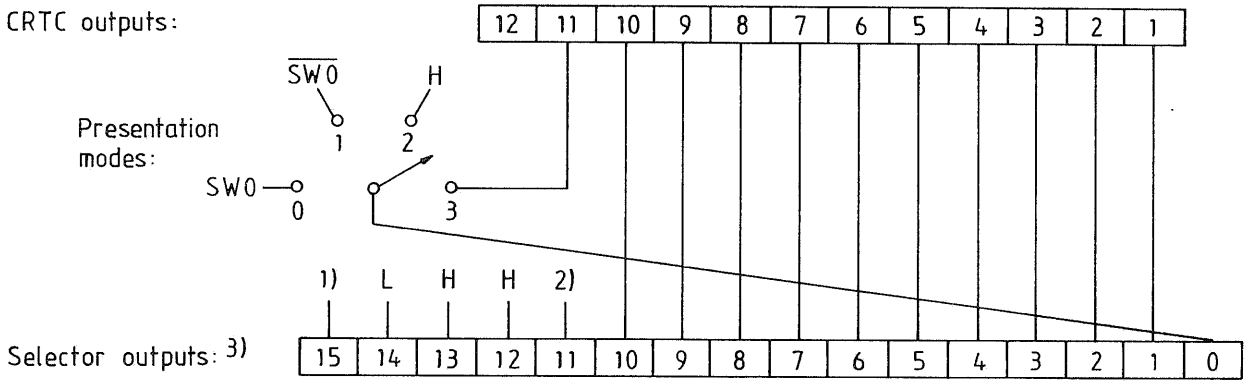
CRTC Memory Addressing

Inputs																		Outputs		Comments
CRTC address bus [1]												Bus addr	PDB [5]	4/8	PM2 [2]	PM1 [3]	SW0 [4]	Bus bit	Level	
12	11	10	9	8	7	6	5	4	3	2	1									
												H	H					15	L	
												H						14	L	
												H						13	H	
												H						12	H	
	L											H	H	L				11	L	8k, normal
												H	L	H				11	L	4k, high end
		L										H						10	L	
			L									H						9	L	
				L								H						8	L	
					L							H						7	L	
						L						H						6	L	
							L					H						5	L	
								L				H						4	L	
									L			H						3	L	
										L		H						2	L	
											L	H						1	L	
L												H		L	L	L		0	L	Pres. mode 3
	L											H		H	L	L		0	L	Pres. mode 3
												H			H	H	H	0	L	Pres. mode 0
												H			H	L	L	0	L	Pres. mode 1

[1] Bus bits 13 and 0 not connected.
[2] From DIA PIA PB2
[3] From DIA PIA PB1
Presentation mode: $11_{(2)} = 0$, $10_{(2)} = 1$, $01_{(2)} = 2$ and $00_{(2)} = 3$
[4] From FPLA DIA II
[5] From MIC PIA PB7: 0 = High end, 1 = Normal display memory

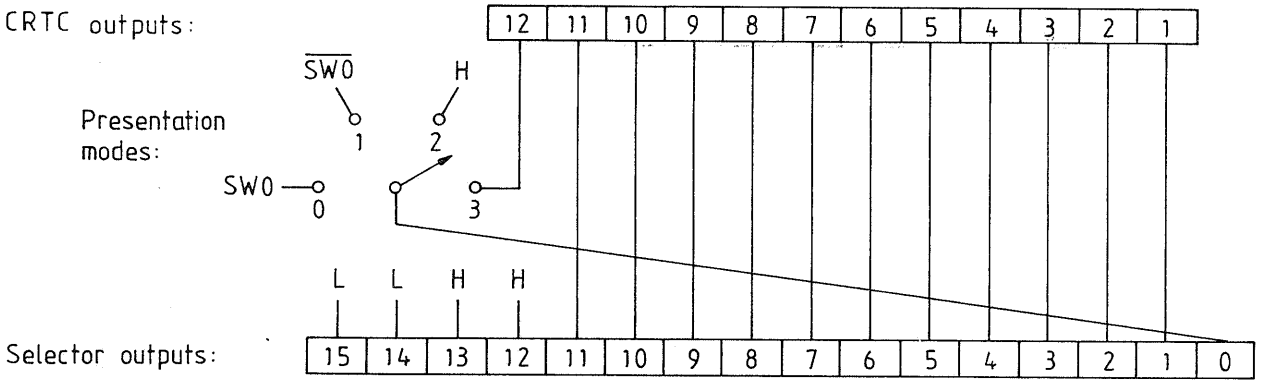
Summary of CRTC Memory Addressing

4 kbytes Display Memory

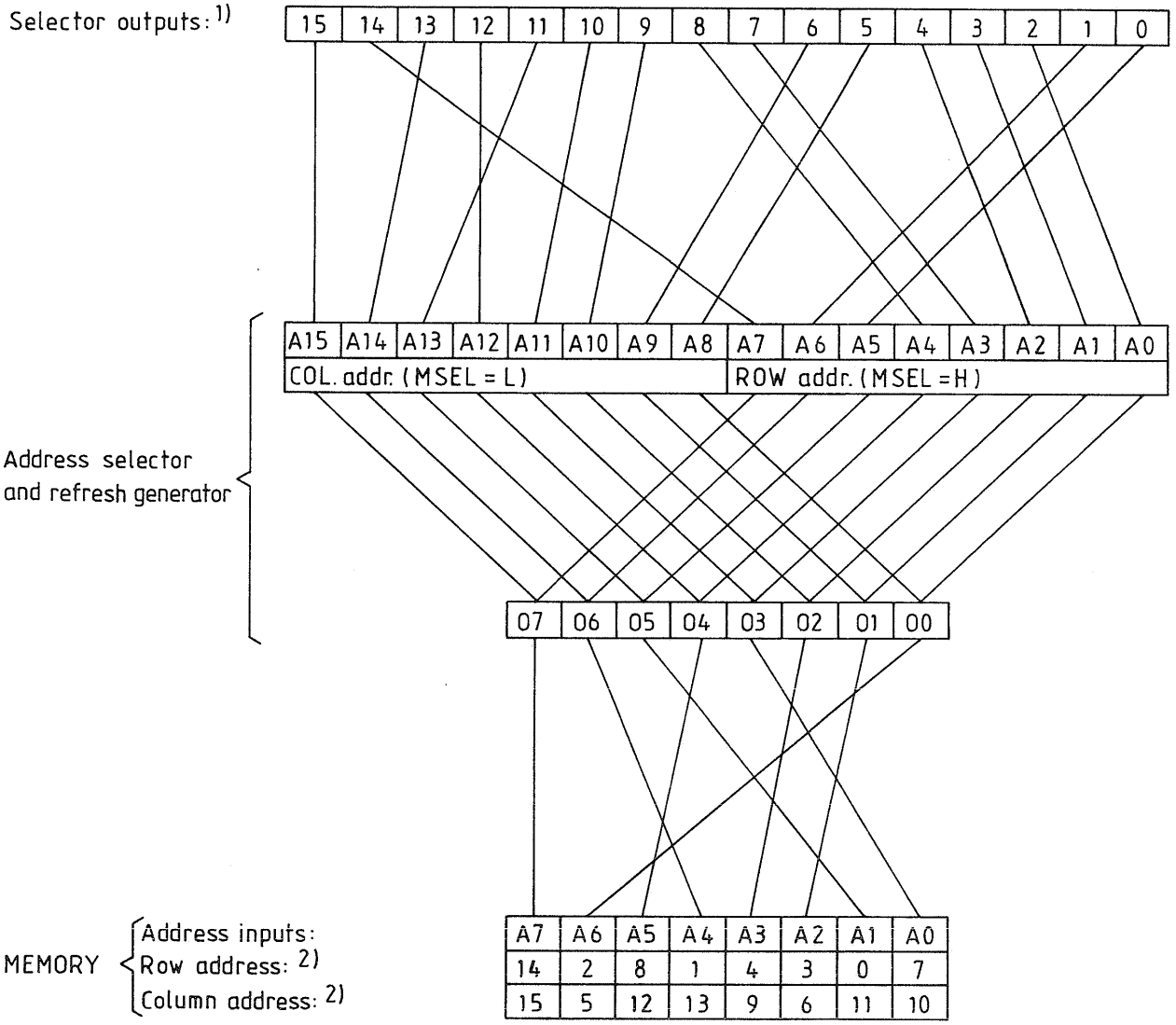


[1] L for PDB = H (Normal display memory)
[2] L for PDB = L (High end display memory)
[3] Numbered as bus lines on logic diagram

8 kbytes Display Memory



Bus/CRTC Selector to Memory Connections



1) Numbered as bus lines on logic diagram
2) Numbered from bus/CRTC selector output bus – i.e. same as 1)

Appendix 4

Character Generator Codes

The binary and hexadecimal values are shown as seen on the diskette or the external data bus. The normal IBM character generators are loaded with the APL characters although these are not utilized.

IBM/APL, National Group A, 25, 33 and 44 Lines

Bits 7 6 5 4 / 3 2 1 0	Hex	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
0 0 0 0	0		@		0	É	P	é	p					¨	P	ι	Θ
0 0 0 1	1	^	|	!	1	A	Q	a	q					A	Q	ρ	O
0 0 1 0	2	§	\	"	2	B	R	b	r	^	[	⊗	⍋	B	R	ω	¯
0 0 1 1	3	¯	`	#	3	C	S	c	s	ˇ	]	□	⍏	C	S	x	
0 1 0 0	4	£	¬	$	4	D	T	d	t					D	T	÷	
0 1 0 1	5	¢		%	5	E	U	e	u					E	U	⊤	■
0 1 1 0	6	¤	}	&	6	F	V	f	v	\	▽	|	⌹	F	V	∩	| |
0 1 1 1	7	Ñ	β	'	7	G	W	g	w	≠	△	≋	⍒	G	W	∪	¯ ¯
1 0 0 0	8	Ø	ø	(	8	H	X	h	x					H	X	≍	|
1 0 0 1	9	→		)	9	I	Y	i	y					I	Y	≎	|
1 0 1 0	A	←	{	*	:	J	Z	j	z	→	≤	≠	⍎	J	Z	⍬	|
1 0 1 1	B	Æ	æ	+	;	K	Ä	k	ä	←	≥	≁	⍕	K	⊃	⍉	▮
1 1 0 0	C		¯	,	<	L	Ö	l	ö					L	⊂	⌶	▮
1 1 0 1	D			−	=	M	Å	m	å					M	⊥	!	▬
1 1 1 0	E	⍐	;	.	>	N	Ü	n	ü	⌈	{	⍐	∘	N	⍺	⍞	▬
1 1 1 1	F	⍗	~	/	?	O	_ [1]	o	¦	⌋	}	⍗	~	O	∈	⍝	

1) Underline in 33 and 44 line formats.

IBM/APL, National Group B, 25, 33 and 44 Lines

Bits 7 6 5 4	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0 Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0 0		@		⓪	ë	P	é	p					¨	P	ι	Θ
0 0 0 1 1	^	|	!	1	A	Q	a	q					A	Q	ρ	O
0 0 1 0 2	§	\	"	2	B	R	b	r	^	[	⊗	⍋	B	R	ω	¯
0 0 1 1 3	û	`	#	3	C	S	c	s	∨	]	□	⍒	C	S	×	
0 1 0 0 4	£	¬	$	4	D	T	d	t					D	T	÷	
0 1 0 1 5	¢		%	5	E	U	e	u					E	U	⊤	■
0 1 1 0 6	ï	}	&	6	F	V	f	v	\	▽	|	⌹	F	V	∩	| |
0 1 1 1 7	â	î	'	7	G	W	g	w	≠	△	≂	⍙	G	W	∪	¯
1 0 0 0 8	°	ü	(	8	H	X	h	x					H	X	≍	|
1 0 0 1 9	¯	▶	)	9	I	Y	i	y					I	Y	≂	|
1 0 1 0 A	ì	{	*	:	J	Z	j	z	→	≤	⍲	⍱	J	Z	⍬	|
1 0 1 1 B	[	]	+	;	K	Ñ	k	ñ	←	≥	⍲	⍧	K	⊃	⍉	▮
1 1 0 0 C		¯	,	<	L	Pt	l	ç					L	⊂	⌶	▮
1 1 0 1 D		◀	–	=	M	è	m	ù					M	⊥	⍎	▬
1 1 1 0 E	ê	¯	.	>	N	à	n	¨	⌈	{	⍋	∘	N	α	⍞	▬
1 1 1 1 F	ô	¿	/	?	O	_	o	¦	⌋	}	⍒	~	O	∈	⍕	

1) Underline in 33 and 44 line formats.

IBM/APL, National Group C, 25, 33 and 44 Lines

Bits 7 6 5 4	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0 Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0 0		@		⓪	ë	P	é	p					¨	P	ι	Θ
0 0 0 1 1	^	|	!	1	A	Q	a	q					A	Q	ρ	O
0 0 1 0 2	§	\	"	2	B	R	b	r	^	[	⊗	⍋	B	R	ω	¯
0 0 1 1 3	û	`	#	3	C	S	c	s	∨	]	□	⍒	C	S	×	
0 1 0 0 4	£	¬	$	4	D	T	d	t					D	T	÷	
0 1 0 1 5	¢		%	5	E	U	e	u					E	U	⊤	■
0 1 1 0 6	ï	}	&	6	F	V	f	v	\	▽	|	⌹	F	V	∩	| |
0 1 1 1 7	â	î	'	7	G	W	g	w	≠	△	≂	⍙	G	W	∪	¯
1 0 0 0 8	ì	ü	(	8	H	X	h	x					H	X	≍	|
1 0 0 1 9	¯	▶	)	9	I	Y	i	y					I	Y	≂	|
1 0 1 0 A	ò	{	*	:	J	Z	j	z	→	≤	⍲	⍱	J	Z	⍬	|
1 0 1 1 B	[	]	+	;	K	ä	k	ö	←	≥	⍲	⍧	K	⊃	⍉	▮
1 1 0 0 C		¯	,	<	L	⁄	l	ç					L	⊂	⌶	▮
1 1 0 1 D		◀	–	=	M	è	m	ù					M	⊥	⍎	▬
1 1 1 0 E	ê	¯	.	>	N	à	n	¨	⌈	{	⍋	∘	N	α	⍞	▬
1 1 1 1 F	ô	~	/	?	O	_	o	¦	⌋	}	⍒	~	O	∈	⍕	

1) Underline in 33 and 44 line formats.

IBM/APL, National Group E, 25, 33 and 44 Lines

Bits 7 6 5 4	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0 Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0 0		@		0	ë	P	é	p					¨	P	ι	θ
0 0 0 1 1	^	|	!	1	A	Q	a	q					A	Q	ρ	o
0 0 1 0 2	§	\	"	2	B	R	b	r	^	[	⊗	⊕	B	R	ω	¯
0 0 1 1 3	ú	`	#	3	C	S	c	s	ˇ	]	□	⊕	C	S	x	
0 1 0 0 4	£	¯	$	4	D	T	d	t					D	T	÷	
0 1 0 1 5	¢		%	5	E	U	e	u					E	U	⊤	■
0 1 1 0 6	í	}	&	6	F	V	f	v	\	▽	|	⊞	F	V	∩	||
0 1 1 1 7	ò	ì	'	7	G	W	g	w	≠	△	≈	⊜	G	W	∪	¯
1 0 0 0 8	°	ü	(	8	H	X	h	x					H	X	≈	|
1 0 0 1 9	~	▶	)	9	I	Y	i	y					I	Y	≈	|
1 0 1 0 A	¡	{	*	:	J	Z	j	z	→	≤	≠	⊕	J	Z	⊕	|
1 0 1 1 B	[	]	+	;	K	Ñ	k	ñ	←	≥	≠	⊕	K	⊃	⊗	▮
1 1 0 0 C		¯	,	<	L	Pt	l	ç					L	⊂	⌶	▮
1 1 0 1 D		◀	−	=	M	è	m	ù					M	⊥	!	▬
1 1 1 0 E	á	¯	·	>	N	à	n	¨	⌈	{	⊕	∘	N	α	⍞	▬
1 1 1 1 F	ó	¿	/	?	O	_	o	¦	⌋	}	⊕	~	O	∈	⍝	

1) Underline in 33 and 44 line formats.

VT100 and PC Overlay Codes

Bits 7 6 5 4	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0 Hex	8	9	A	B	C	D	E	F
0 0 0 0 0					—			
0 0 0 1 1					◆			
0 0 1 0 2					⁞			
0 0 1 1 3					H T			
0 1 0 0 4					F F			
0 1 0 1 5					C R			
0 1 1 0 6					L F			
0 1 1 1 7					°			
1 0 0 0 8					±			
1 0 0 1 9					N L			
1 0 1 0 A					V T	≤		
1 0 1 1 B						≥		
1 1 0 0 C						π		
1 1 0 1 D						≠		
1 1 1 0 E						£		
1 1 1 1 F						•		(ML)

UTS, National Group A, 25 Lines

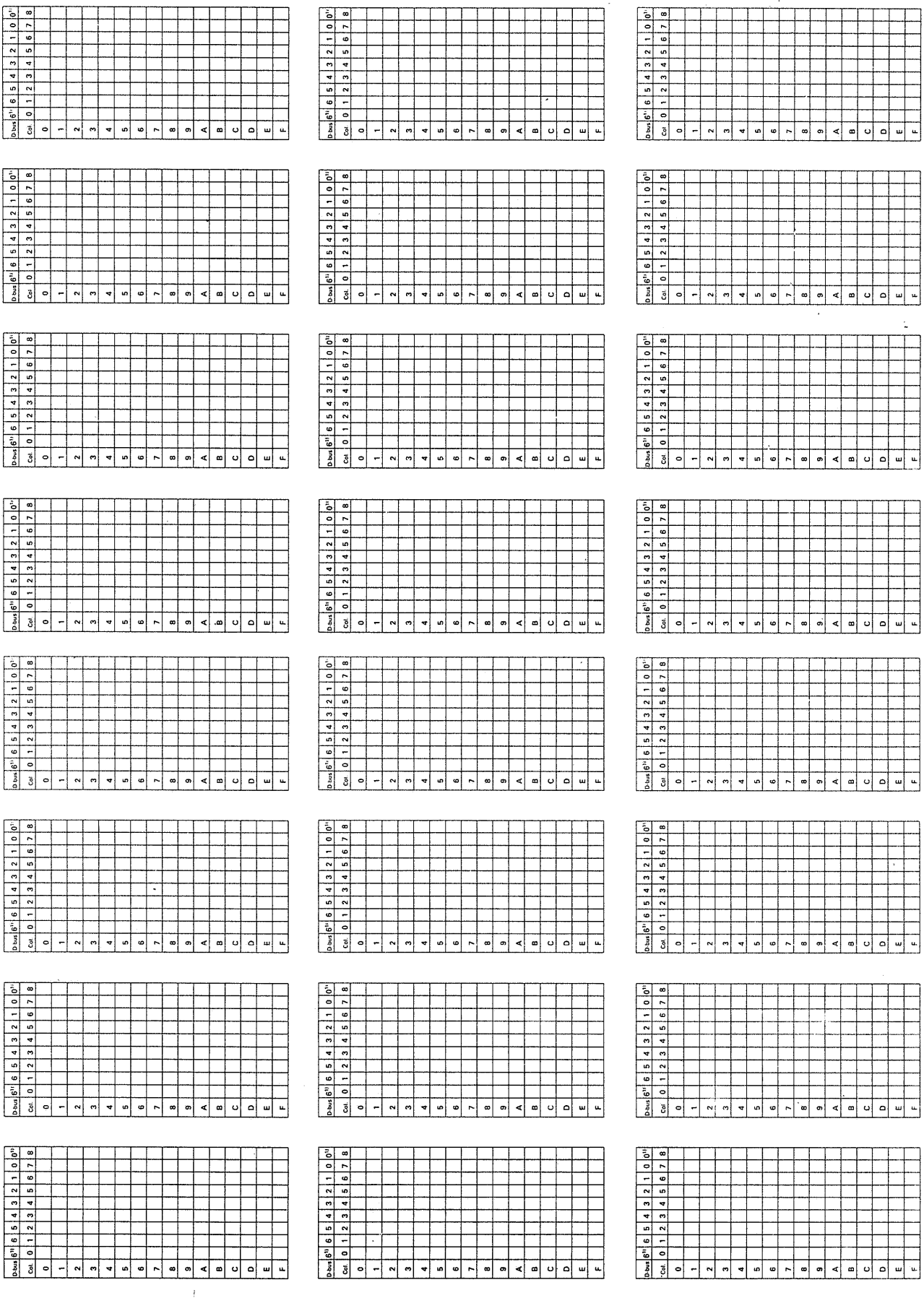
Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	É	P	é	p								
0 0 0 1	1	^	|	!	1	A	Q	a	q								
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	½	`	#	3	C	S	c	s								
0 1 0 0	4	£	—	$	4	D	T	d	t								
0 1 0 1	5	[	°	%	5	E	U	e	u								
0 1 1 0	6	]	}	&	6	F	V	f	v								
0 1 1 1	7	₪	ß	´	7	G	W	g	w								
1 0 0 0	8	Ø	ø	(	8	H	X	h	x								
1 0 0 1	9	→|	~	)	9	I	Y	i	y								
1 0 1 0	A	↓	{	*	:	J	Z	j	z								
1 0 1 1	B	⤓	¤	+	;	K	Ä	k	ä								
1 1 0 0	C	↓↓	▶	,	<	L	Ö	l	ö								
1 1 0 1	D	↵	◀	−	=	M	Å	m	å								
1 1 1 0	E	Æ	▷	·	>	N	Ü	n	ü								
1 1 1 1	F	æ		/	?	O	_	o	¦								

UTS, National Group B, 25 Lines

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	ë	P	é	p								
0 0 0 1	1	^	|	!	1	A	Q	a	q								
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	û	`	#	3	C	S	c	s								
0 1 0 0	4	£	—	$	4	D	T	d	t								
0 1 0 1	5	¿	½	%	5	E	U	e	u								
0 1 1 0	6	ï	i	&	6	F	V	f	v								
0 1 1 1	7	â	î	´	7	G	W	g	w								
1 0 0 0	8	°	ü	(	8	H	X	h	x								
1 0 0 1	9		~	)	9	I	Y	i	y								
1 0 1 0	A	↓	▭	*	:	J	Z	j	z								
1 0 1 1	B	⤓	¤	+	;	K	Ñ	k	ñ								
1 1 0 0	C	↓↓	▶	,	<	L	Pt	l	ç								
1 1 0 1	D	↵	◀	−	=	M	è	m	ù								
1 1 1 0	E	ê	▷	·	>	N	à	n	¨								
1 1 1 1	F	ô		/	?	O	_	o	¦								

Character Dots Decoding

(This sheet is inserted in order to be copied and used for easy character dots decoding from character generator contents as seen on the diskette or the external data bus.)



[1] If bit 7 = 1

Display Unit, DU 4113

Contents

General

Display Unit DU 4113 is a CRT (cathode ray tube) display which can be used in a variety of configurations.

The Microcomputer chapter ought to be understood before this chapter is studied. A good orientation on the Communication chapter is also recommended; especially the Hardware part under Internal Communication via Two-wire.

The chapter Commands and Orders and especially what is said about Extended Data Stream in the chapter Optional Functions, in the Reference Manual IBM 3270 Emulation, should also be read before this chapter.

The DU 4113 is a seven-colour-display unit. The colours are; red, blue, green, yellow, turquoise, pink and white. It is primarily intended for emulations of IBM 3279 colour displays with business graphics feature and full set of programmed symbol (PS) stores. Preparations are, however, made for non-IBM mode operation and the use of a reduced set of programmed symbol stores both in IBM and non-IBM mode. These features are described in this chapter but are normally of no interest to the reader.

Functions

The display unit can communicate with a host computer via a communication processor (cluster configuration). It can communicate with two host computers by means of two communication processors. The display unit can also communicate with flexible disk units. These communications are carried out via a two-wire or a coaxial cable. The display unit is also capable of communicating via a Standard serial interface of ß-type (SS3-ß bus).

Facilities are provided for connections of a keyboard unit (KBU 4143), a keyboard expansion unit (KXU 4146) and a magnetic stripe reader (MSR 4136).

The main tasks of the display unit are:

- Presentation of visual information on a screen. This information may be text entered via the keyboard, messages from a host computer or the operating system, data from a flexible disk etc.
- Communication with host computers and other system units like a keyboard, printers, flexible disk units and communication processors.
- Editing and processing of data, code conversions etc.

Subunits

The main parts of the display unit are:

- DBM-C, display unit basic mechanics assembly: cabinet with internal racks but exclusive of the parts of the cathode ray tube unit and the power supply (see below).
- CTF, CP and terminal fan (included in DBM-C).
- DPS, display power supply including printed circuit board, PCB. See the chapter, "Power Supplies".
- DTC-D, display terminal controller containing a microcomputer, 64 kbytes of basic read/write memory, presentation logic, two-wire (or coaxial) and keyboard interfaces.
- CHGB, character generator board, which can be of two types – either a reduced version with memory space for four single plane programmed symbol stores or the full version with space for four single plane and four triple plane or, when run in non-IBM mode, four triple plane programmed symbol stores.
- CRU, cathode ray tube unit with circuitry for beam control.
- ICBD, interconnection board (included in DBM-C) with connectors for keyboard, two-wire, coaxial cable and a DIP-switch for address setting and connection of two-wire or coaxial cable terminating resistors.

Mechanics

The mechanical build-up of the display unit is shown in Fig. 1. The main parts of the display unit, from the mechanical aspect, are:

- DBM-C, display unit basic mechanics assembly with connectors (for mains, two-wire, coaxial cable and keyboard) and controls (Mains switch and Brightness control). Detailed information on the location of connectors and control devices etc. can be found in the Installation and Maintenance Manual. A fan, CTF, is included in the DBM-C.
- DPS, display terminal power supply.
- Housing (included in DBM-C) for the DTC-D and CHGB boards.
- CRU, cathode ray tube unit including tube, mechanical supports and circuitry.

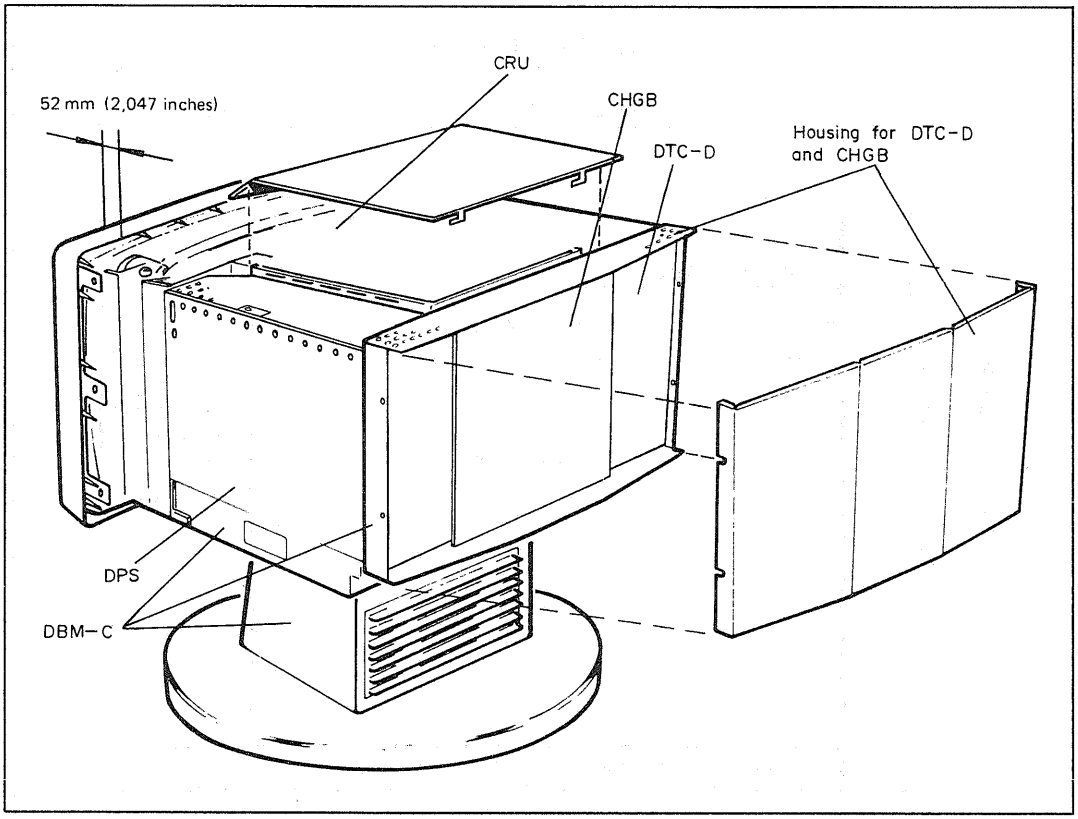


Fig. 1. Mechanical layout of DU 4113

Brief Outline

Presentation

Screen Scanning

Presentation of characters on the screen is made by a raster scan method, i.e. an electron beam makes a number of consecutive horizontal sweeps over the screen. (The "beam" is actually composed of up to three different beams, see Cathode Ray Tube Unit, CRU.) The beam is modulated by video information, thus generating dots that build up characters, a cursor and underlines.

The regeneration rate is 50 Hz, that is, the electron beam builds up a whole picture on the screen every 20 milliseconds. See Fig. 2.

Depending on software the number of lines can be 25 or 33, that is, normally 24 or 32 ordinary text lines and one message line. Of course, any number of these may lack text and thus be invisible.

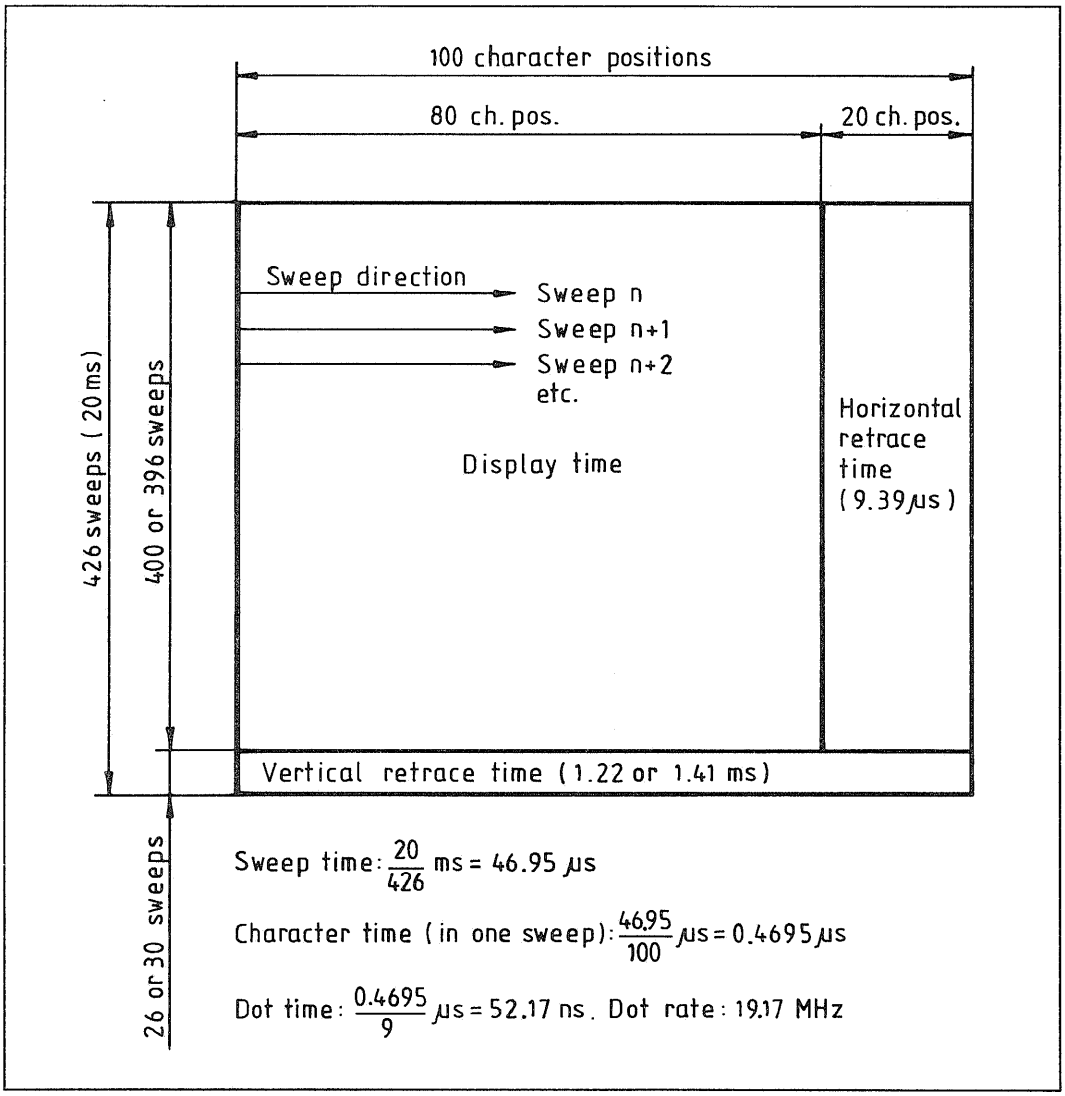


Fig. 2. Screen scanning times

The total number of sweeps is 426. Depending on screen format, 400 (25 lines – including message line) or 396 sweeps (33 lines – including message line) are used for presentation, spacing sweeps included. Thus, 26 or 30 sweep times are used for vertical retrace.

Each sweep is divided into 100 character positions out of which up to 80 may be used for presentation. The remaining position times are used for horizontal retrace. Note that several sweeps are needed to build up the height of a character.

Each character position is horizontally divided into nine dot positions or columns (0 to 8).

The dot rate is 19.17 MHz (see Fig. 2), i.e. equal to the frequency of the basic crystal clock and 18 times the system clock rate (Ø1, 1.065 MHz). Thus, in a specific sweep, the dots of two characters are presented during one system clock period.

(Unlike vertical lines, horizontal lines are actually not formed by discrete dots but by continuous strokes of the electron beam.)

Presentation Geometry

Each sweep in a line is identified by a raster address. Each line starts at raster address 0. See Fig. 3. The number of sweeps per line is programmable and decides e.g. the maximum height of the character matrix. For example, with 16 sweeps per line a 9 × 16 character matrix can be used, i.e. up to 9 horizontal dots (one for each column, see above) and up to 16 vertical dots (one for each sweep) can be used to build a character. The character generator (which will be explained later) may, however, only contain information for 12 sweeps and the character matrix must then be limited to 9 × 12 dots followed by 4 invisible sweeps, separating the text of the different lines.

The display unit can be run in IBM mode and non-IBM mode. The used combinations of mode, number of lines and sweeps as well as the image size on the screen are summarized below:

Mode	Number of lines[1]	Sweeps per line	Used sweeps per line[2]	Screen size (nominally)	
				mm	inches
IBM alternate	33	12	12	180 × 245	7.09 × 9.65
IBM text	25	16[3]	12	180 × 245	7.09 × 9.65
IBM graphic	33[4]	12	12	137[5] × 245	5.39[5] × 9.65
Non-IBM	25	16	16	182 × 245	7.17 × 9.65

[1] Including message line.
[2] Depending on character generator limitations.
[3] Can also be 12, 13, 14 or 15 but the IBM compatibility is then lost.
[4] Only 24 plus the message line are displayed.
[5] Exclusive of not used sweeps and the message line.

Raster address	Column 8[1]	0	1	2	3	4	5	6	7	8	0[2]
(F)[3]											
0											
1											
2											
3											
4											
5											
6											
7											
8											
9											
A											
B											
(C)											
(D)											
(E)											
(F)											
0[4]											
1											

1) Last column of preceding character
2) First column of following character
3) Last sweep of preceding line
4) First sweep of following line

Fig. 3. Raster address/Column number layout

The total number of sweeps is still 400 in IBM graphic mode although at most only 300 (25 × 12) juxtaposed sweeps may be visible.

The above-mentioned alternatives can only be selected (by software) for the full screen, i.e. not for a number of discrete lines.

The horizontal distance between two adjacent dot positions is ≈ 0.34 mm. The vertical distance between two adjacent dot positions (sweeps) is ≈ 0.46 mm.

Thus, the ratio: $\frac{\text{vertical dot spacing}}{\text{horizontal dot spacing}} \approx 1.35$

Some further information about the presentation geometry can be found under Character Generator Organization.

Display Memory

The display memory in DU 4113 is physically a separate memory and some parts of it also substitute for parts of the ordinary read/write memory (see Detailed Description for addresses etc.) but the actual display memory part is only considered here.

The memory contains 8-bit codes for characters, CHs, or field attributes, FAs, at odd addresses and character attributes, CAs, or extended field attributes, EFAs, at even addresses. (Even and odd refer here to how they are seen by the microcomputer.) If there is a CA at an even address the next odd address will contain the corresponding CH. The EFAs and FAs are paired in the same way. Note that the display memory code, due to code conversions performed by the software, may differ from the line code (used at communication with a host computer) and the code received from the keyboard.

The microcomputer can access the memory once during each system clock cycle independently of the cathode ray tube controller (see below).

Principles of Character Generation

The presentation is supervised by a cathode ray tube controller, CRTC – an LSI circuit. See Fig. 4. The CRTC generates a raster address (0) and a number of display memory addresses, then a new raster address (1) and the same memory addresses etc. until a line is completed. Then it goes on with a new set of raster and memory addresses, i.e. the next line. It also generates some other control signals.

The CRTC is timed to produce two memory addresses during each system clock cycle. The address lines are via a selector wired so that both an even and an odd (as seen by the microcomputer) location will be accessed at one addressing by the CRTC. The outputs from the even and odd locations will be considered under Attributes and the output from the odd below.

The output from the display memory at CRTC-addressing is latched and then sent to an attribute generator (see Attributes) and one or three character generator block(s). There is one character generator block in

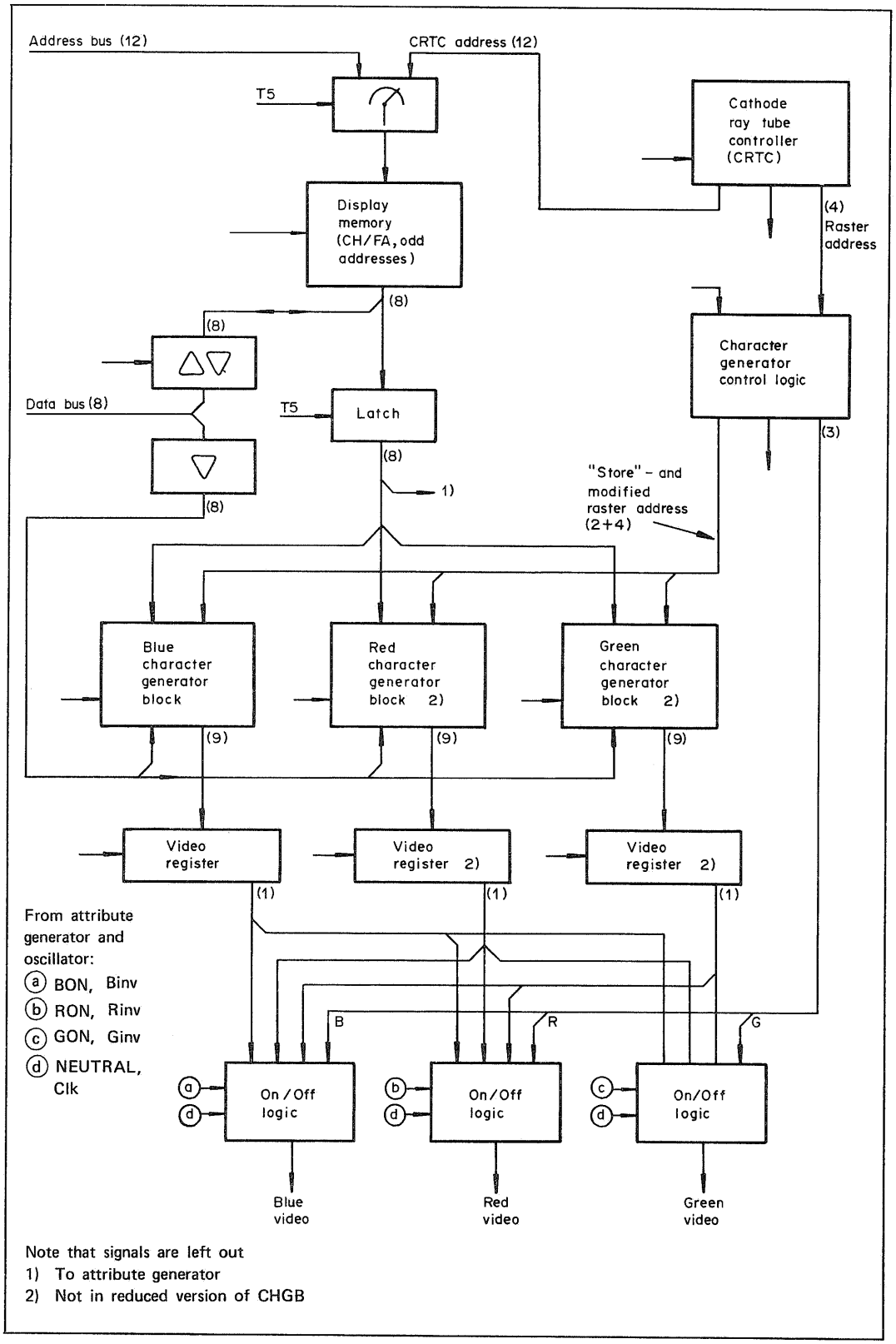


Fig. 4. Principles of character generation

the reduced version of the CHGB and three in the full version. They are called the Blue, Red and Green block. Each block is a 16 k× 9 bits memory and each 9-bit location can contain dot values (on or off) for one colour (Not necessarily the same as in the name of the block.) in one sweep through a character position.

The CRTC will, as said above, also provide a raster address, telling which sweep of a line that is in progress. Together with signals from the attribute generator and other sources, the raster address is fed to a character generator control logic, which modifies the raster address and supplies additional address and control signals.

The character generator block or blocks are thus addressed in parallel by:

- Character code, CH. 8 bits. (The generators are also addressed by FAs, but this is irrelevant.)
- Modified raster address. 4 bits.
- "Store" address. 2 bits from the character generator control logic.

The character generator block(s) can be loaded (and in a special way read) by the microcomputer. The internal organization of the character generators is summarized under the next heading.

The output from a character generator block is loaded in parallel into a video register and serially shifted out from the latter as video information at a rate of 19.17 MHz. There is one video register for each block, that is, only one register in the reduced version but three in the full version of the CHGB.

The video information is then via three on/off logics fed to the cathode ray tube unit as Blue, Red and/or Green video. Note that the outputs from all three video registers are fed to all three on/off logics. It is thus possible to channel information from any character generator block/ video register to any video output (Blue, Red, Green). The selection of path or paths is controlled by signals from the character generator control logic and the attribute generator.

Character Generator Organization

The character generators are organized in different ways depending on selected functional mode and CHGB version. The number of available programmed symbol stores and their types, Single or Triple plane stores, varies. These interrelationships are summarized below:

Store No.	Reduced version		Full version	
	IBM mode	Non-IBM mode	IBM mode	Non-IBM mode
0	Single	Single	Single	Triple
1	Single	Single	Single	Triple
2	Single	Single	Single	Triple
3	Single	Single	Single	Triple
4			Triple	
5			Triple	
6			Triple	
7			Triple	

A character position derived from a single plane store can only contain one colour while that from a triple plane store can be composed of up to seven colours.

All utilized stores must be loaded by the program before they are used for presentation.

In IBM emulations, store 0 (standard character set) and 1 (APL/text character set) must be loaded locally (= not from the host) as required. The available character sets (stored on flexible disks) are shown in Appendix 4. The rest of the character sets can be loaded by the host computer. IBM considers store 6 to be a single plane store. This does not matter since a triple plane store automatically functions as a single plane store if the data stream loads it as if it were a single plane store (yields the same contents in all three planes).

One character code addresses a dot pattern file consisting of one 9-bit location for each one of 12 (IBM mode) or 16 (non-IBM mode) sweeps. One store has room for 256 character dot patterns, but one (MLFA, message line field attribute) or 33 cannot be used as they may be interpreted as attributes (MLFA or FAs). The mappings of the character generators are shown in Figs 5 and 6. Note that the figures refer to ordinary presentation, at message lines e.g. sweep 4 presents the character generator locations for "ordinary" sweep 3 and the last "ordinary" line is not displayed. That these mappings and presentation rearrangements are possible depends on the raster address modification in the character generator logic, see Fig. 4.

Full version			Reduced version
Blue block	Red block	Green block	(Blue block)
Store 4 Sweeps 0–B	Store 4 Sweeps 0–B	Store 4 Sweeps 0–B	Store 0 Sweeps 0–B
Store 0 Sweeps 0–3	Store 0 Sweeps 4–7	Store 0 Sweeps 8–B	Not used
Store 5 Sweeps 0–B	Store 5 Sweeps 0–B	Store 5 Sweeps 0–B	Store 1 Sweeps 0–B
Store 1 Sweeps 0–3	Store 1 Sweeps 4–7	Store 1 Sweeps 8–B	Not used
Store 6 Sweeps 0–B	Store 6 Sweeps 0–B	Store 6 Sweeps 0–B	Store 2 Sweeps 0–B
Store 2 Sweeps 0–3	Store 2 Sweeps 4–7	Store 2 Sweeps 8–B	Not used
Store 7 Sweeps 0–B	Store 7 Sweeps 0–B	Store 7 Sweeps 0–B	Store 3 Sweeps 0–B
Store 3 Sweeps 0–3	Store 3 Sweeps 4–7	Store 3 Sweeps 8–B	Not used

Not valid for message line presentation. Sweep 0 is then always blue and sweeps 1–B contain the information of character generator contents for ordinary (as stated above) sweeps 0–A.

All figures are in hexadecimal.

Fig. 5. IBM mode character generator maps

Full version			Reduced version
Blue block	**Red block**	**Green block**	**(Blue block)**
Store 0 Sweeps 0 – F	Store 0 Sweeps 0 – F	Store 0 Sweeps 0 – F	Store 0 Sweeps 0 – F
Store 1 Sweeps 0 – F	Store 1 Sweeps 0 – F	Store 1 Sweeps 0 – F	Store 1 Sweeps 0 – F
Store 2 Sweeps 0 – F	Store 2 Sweeps 0 – F	Store 2 Sweeps 0 – F	Store 2 Sweeps 0 – F
Store 3 Sweeps 0 – F	Store 3 Sweeps 0 – F	Store 3 Sweeps 0 – F	Store 3 Sweeps 0 – F

Not valid for message line presentation. Sweep 0 is then always blue and sweeps 1 – F contain the information of character generator contents for ordinary (as stated above) sweeps 0 – E.

Note that addressing of store 4 cause subscript of store 0 etc. That is, sweeps 0 – 5 are blanked and sweeps 6 – F contain the information of character generator contents for ordinary (as stated above) sweeps 0 – 9. On the message line: sweep 0 is blue, 1 – 6 are blanked and 7 – F contains ordinary sweeps 0 – 8 information.

All figures are in hexadecimal.

Fig. 6. Non-IBM mode character generator maps

Underlining

Underlining exists in three forms:

- Underlined space or character, generated by the character generator as a normal character.
- Underlined position, generated by the attribute generator.
- Underlined field, generated by the attribute generator.

The two last forms are caused by a character attribute, CA, or an extended field attribute, EFA, respectively. Sweep $B_{(16)}$ in IBM mode or $F_{(16)}$ in

non-IBM mode is used for underline (when required) from the attribute generator, even at message lines.

A blue "overline" (the uppermost sweep) separating the message line from the rest of the screen is produced at the position of a message line field attribute, MLFA, and remains for the rest of the line.

Underlinings are of the same colour as the text except in multi-colour-positions, where they are white.

Cursor

The cursor is generated separately. Its height and vertical location in the character cell is programmable. The cursor can also be blank, steady or blinking at 1.6 or 3.1 Hz. Two types of cursors may be selected by the program:

Cuform 0: Covering cursor, width 9 dots. The cursor part of a character at the position of a steady cursor is covered by always white dots.

Cuform 1: Transparent cursor, width 9 dots. A character at the cursor position is always visible due to text inversion at the cursor position. This means in a single-colour-position that normally dark dots are shown in the colour that applies to the position and normally lit (coloured) dots are dark. In a multi-colour-position the normal colours of the dots are complemented. (Complements; dark/white, green/pink, blue/yellow, turquoise/red and vice versa.) This is, however, not valid for not presented fields, in which the cursor is green.

Cursor blinking means an alteration between the state of a steady cursor (of Cuform 0 or 1) and the state of no cursor.

Global presentation off, the blue uppermost sweep of a message line, and interrow gaps (sweeps $C_{(16)} - F_{(16)}$) are never overridden by the cursor.

Static Attributes

Apart from the control registers of the cathode ray tube controller, defining the display area and timing, cursor position, format and blink, there are two 8-bit registers of a display adapter peripheral interface adapter, DIA PIA, that either directly control some video attributes or define how to interpret the dynamic attributes (see below). The DIA PIA registers are set by software.

Note that the DIA PIA definitions are static, i.e. cannot be made for specific fields on the screen as there is no simple connection between display timing and program except for the vertical synchronization signal.

The DIA PIA functions are specified in the Detailed Description and summarized below:

- Video control. The video signals can be blocked and the cathode ray tube controller (kept) reset. Either action will stop presentation.
- Cursor control. Cuform 0 or 1 can be selected (see Cursor).
- Field attribute interpretation control. Field attribute interpretation can be enabled or disabled, i.e. field attributes are interpreted as characters. The MLFA as well as the EFA accompanying the MLFA and all CAs are, however, always active.
- Stop at end of line or end of last text line (message line excluded) control. The functions of FAs and EFAs can be ended by programmable stops at end of line or end of last text line (message line excluded). Thus line or page wraparound functions of these attributes are inhibited. The message line (caused by the MLFA) will, however, always stop at the end of line.
- Base colour override control. At base colour override the colours are controlled by the CAs or the EFAs and the Base colour flag mentioned below is overridden. Default attributes give "monochrome" presentation.
- Mode control. IBM or non-IBM mode can be selected.

An additional static attribute is set by the software in a microcomputer peripheral interface adapter, MIC PIA, i.e. the Base colour flag indicating "monochrome" or base colour mode. These modes will cause the following colours, depending on the FA bits 5, 3 and 2 when the CA and EFA contain default values:

Bits 5	3	2	Base colour mode	"Monochrome" mode
0	0	X	Green	Green
0	1	0	Red	White
1	0	X	Blue	Green
1	1	0	White	White
X	1	1	Off	Off

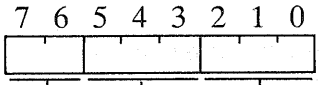
Dynamic Attributes

This section focuses on the presentation logic handling of the dynamic attributes and does not give a general description of attribute usage. It is therefor recommended to study the chapter on Commands and Orders and what is said about Extended Data Stream in the chapter on Optional Functions in the Reference Manual IBM 3270 Emulations.

The attributes are used together with static control bits to define various properties, other than the text itself, of the screen. The attributes are of two main types: CAs and FAs/MLFAs/EFAs. The first type is only valid for one character position but the second applies to a field i.e. a number of character positions.

Extended Field and Character Attribute Formats

Extended field attribute (EFA) and character attribute (CA) information from the host computer (and sometimes CAs from the keyboard) is recoded by the software and stored at even addresses in the display memory. The EFAs and CAs then consist of 8-bit bytes, formatted in the same way. Such a byte is interpreted as an EFA if the following odd address contains an FA or an MLFA or as a CA if the following odd address contains a CH. The format of the byte is as follows:



$000_{(2)}$ = Default. (In CA: symbol store number according to EFA, if also default in EFA: symbol store number 0.)
$001_{(2)}$ = Symbol store number 1.
$010_{(2)}$ = Symbol store number 2.
⋮
$111_{(2)}$ = Symbol store number 7.

Symbol stores 4–7 are only used in IBM mode with full version of character generators. Bit 2 = 1 is in non-IBM mode interpreted as subscript.

$000_{(2)}$ = Default. (In CA: colour according to EFA, if also default in EFA: according to FA.)
$001_{(2)}$ = Blue colour.
$010_{(2)}$ = Red colour.
$011_{(2)}$ = Pink colour.
$100_{(2)}$ = Green colour.
$101_{(2)}$ = Turquoise colour.
$110_{(2)}$ = Yellow colour.
$111_{(2)}$ = Neutral or white colour.

$00_{(2)}$ = Default. (In CA: highlighting according to EFA, if also default in EFA: no highlighting.)
$01_{(2)}$ = Flash.
$10_{(2)}$ = Invert video.
$11_{(2)}$ = Underline.

Field and Message Line Field Attribute Formats

Field attributes (FAs) are supplied by the host computer and message line field attributes (MLFAs) are generated by the display unit software. They are stored among the characters at odd addresses in the display memory. At presentation an MLFA always start one message line and no FAs will be recognized on the rest of that line. The MLFA code is $FF_{(16)}$. If field attribute interpretation is enabled (see: Static Attributes) and no message line is in progress the data from the display memory odd addresses will be checked and recognized as FAs if bits 7, 6 and 1 = $100_{(2)}$. (These codes are handled as characters on the message line.) An FA starts a new field on the screen.

The position of an FA or MLFA is always blank on the screen, even in inverted video. The field extends to the next FA or MLFA unless there is a stop at end of line or at end of last text line (see: Static Attributes).

The hardware reacts on bits 5, 3 and 2 as said under Static Attributes. Note, however, that the software also reacts on bit 5 (unprotected/ numeric input) and 0 (modified data tag).

Principles of Attribute Handling

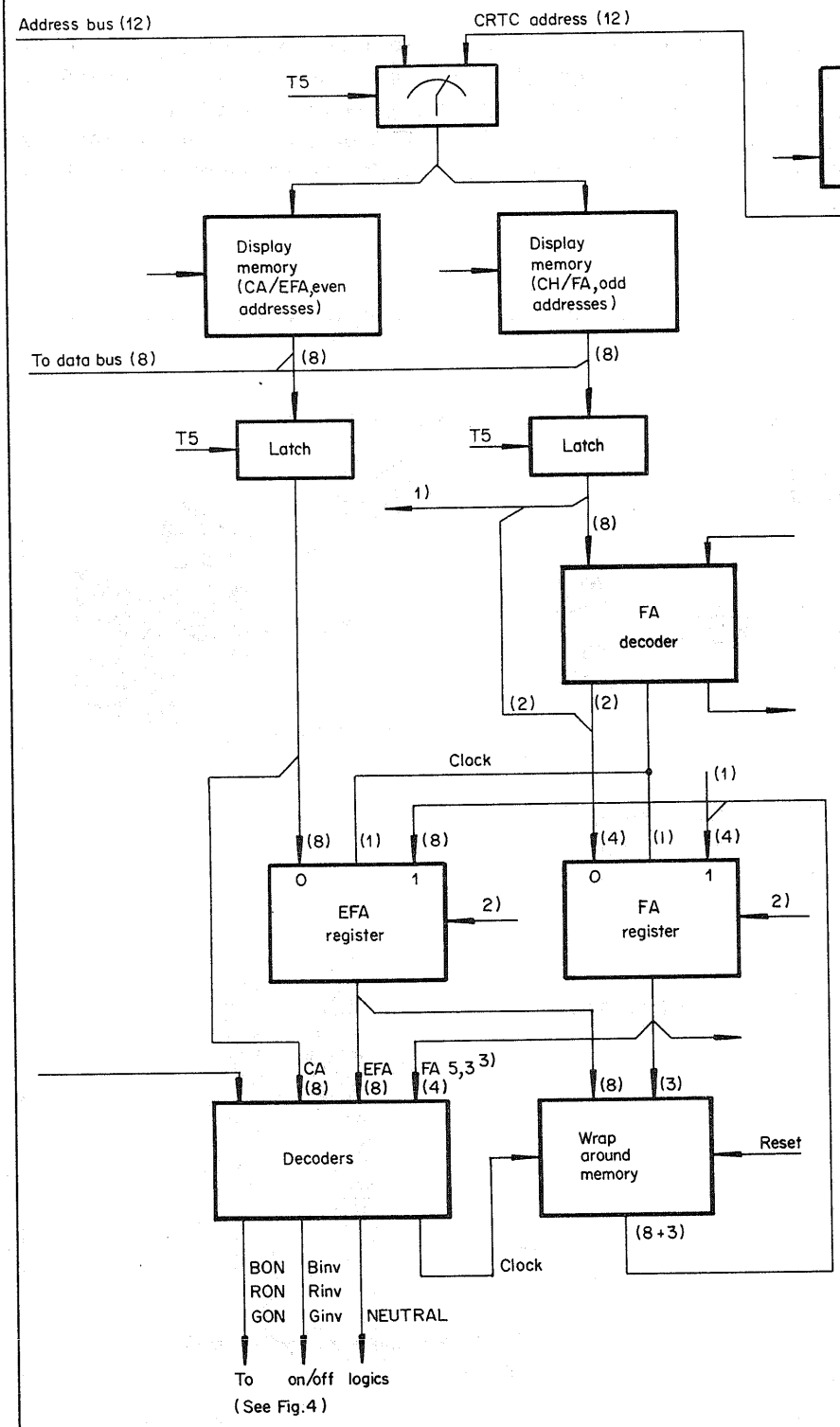
This is only a brief outline of the attribute handling and more information can be found in the Detailed Description.

At presentation the CRTC produces two display memory addresses each system clock cycle. See Fig. 7. The contents of one even and one odd address are accessed at each single addressing. Thus a CA and a CH or an EFA and an FA/MLFA are latched for each position on the screen.

The FA decoder checks the stream of CHs/FAs/MLFAs and generates a clock signal each time an FA or MLFA is encountered, as well as at each end of line (controlled by the horizontal synchronization signal). The clock signal loads an EFA and an FA register. At the encounter of an FA or MLFA the EFA register is loaded with the 8-bit EFA from the display memory and the FA register with the FA bits 5 and 3 as well as two decoded signals.

The contents of the EFA and FA registers (except one bit) are at each end of line except at the end of a message line transferred to a wrap-around memory. The EFA and FA registers are then reloaded from the wrap-around memory before the start of the next line. Note that the contents of the wrap-around memory before a message line will be transferred to the registers before the line after the message line. In this way the EFAs and FAs may be valid from line to line and from the last text line to the first text line (message line excluded). The wrap-around memory can, however, also be reset before each text line or before the first text line is started. This reset is a function of the "stops" described under Static Attributes.

The contents of the EFA and FA registers are also fed to four decoders, which thus have access to the present or preceding EFA and FA information. The present CA information is also supplied to the decoders from the display memory as well as some other signal e.g. the raster address. The decoders interpret this information and the end result is seven control signals, fed to the on/off logics. These control signals will together with three signals from the character generator control logic (see Fig. 4) channel the information from the character generator blocks/ video registers to the applicable blue, red or green video output(s) as well as decide if the video signals shall be inverted or inhibited.



Note that signals are left out

1) To character generator
2) Select controlled by horizontal synchrounization signal derivative
3) And two decoded signals

Fig. 7. Principles of attribute generation

Cathode Ray Tube Unit, CRU

The cathode ray tube unit, CRU, generates pictures according to a raster scan method: three converging electron beams make (if all are on) together a number of consecutive horizontal sweeps over a screen. By means of a mask arrangement each beam hits just one type of colour-emitting phosphor. See Fig. 8. If all beams are on at the same time the mixed effect is white.

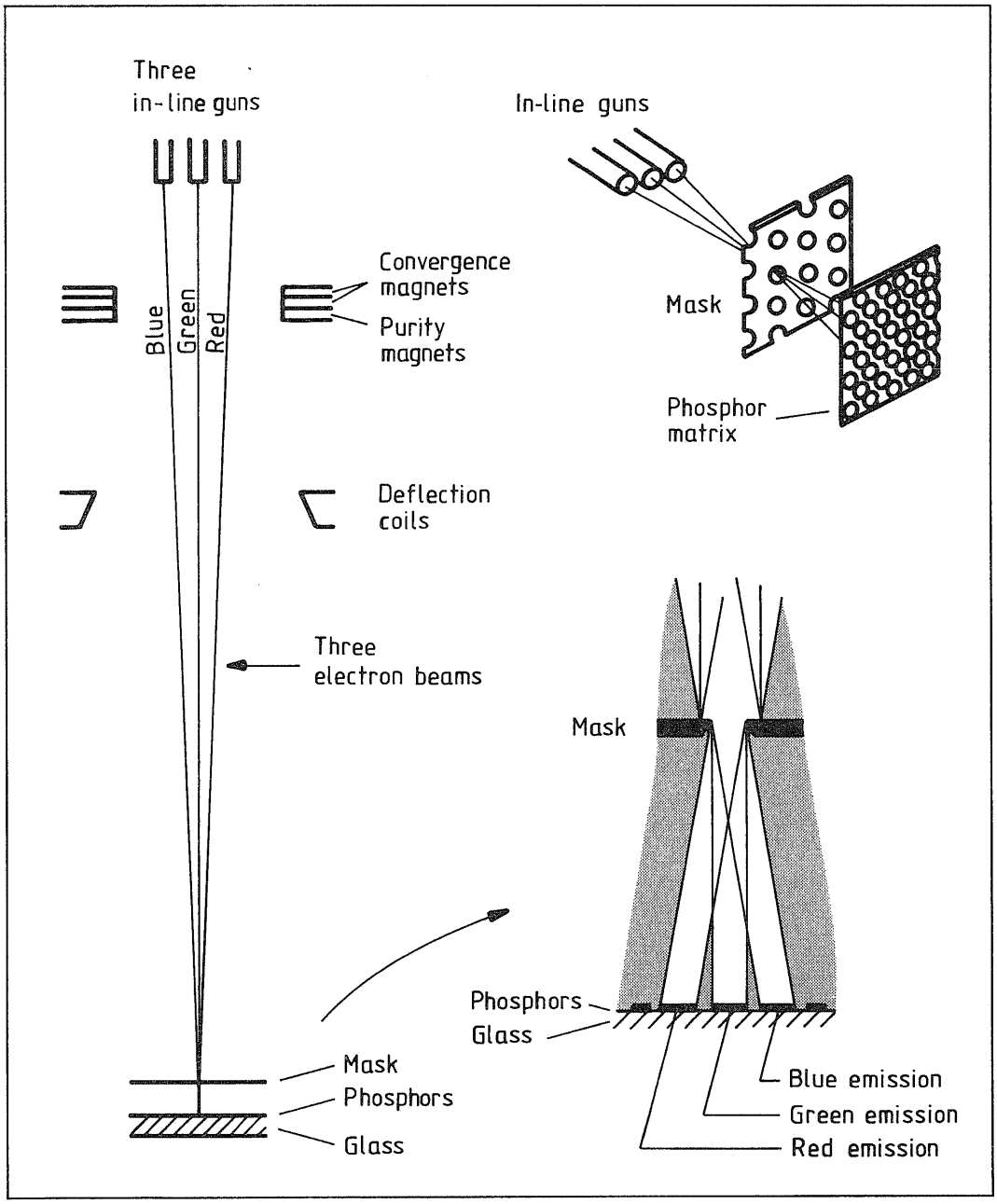


Fig. 8. Principles of colour CRT

The CRU consists of mechanics, a colour cathode ray tube (CRT) with deflection coils and four printed circuit boards (Interface board, CRT socket board, Analog board and High voltage stabilizer board).

The CRT is a 14 inches, high resolution, 90° tube. See Fig. 9. The CRT is fed with high voltage for the anode accelerating the electron beams, voltages for the electrode focusing the beams and the control grids, current for heating of the cathodes and video signals to the cathodes. It is furthermore fed with currents that affect the deflection coils which govern the vertical and horizontal deflection of the beams.

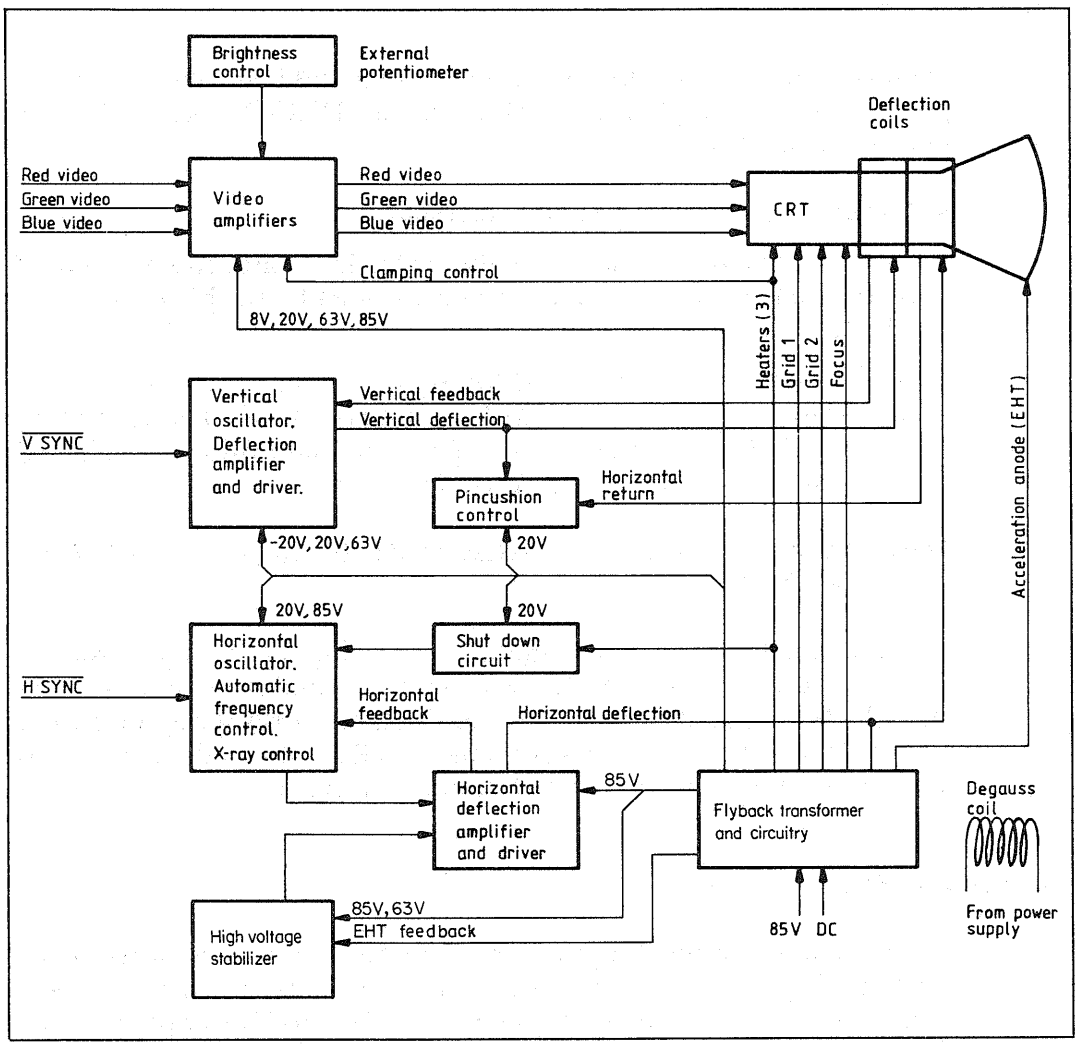


Fig. 9. CRU block diagram

The four logic boards (see Fig. 9) contain circuitry for vertical and horizontal deflection (oscillators, output stages), pincushion correction control and X-ray supervision that will shut down the horizontal deflection (and thereby also inhibit the flyback transformer outputs) if the voltages increase for any reason. The boards also contain video amplifiers, brightness control circuitry and circuits for flashover protection. The beams are modulated by the video information; dots and horizontal lines forming the picture are thus generated on the phosphor coated screen.

The CRU is supplied with 85 V DC from the display terminal power supply. A degaussing coil is activated for a short time at each power on

(from circuits in the terminal power supply). The interface to the DTC-D board consists of five TTL level signals:

- Three video signals for red, green and blue video.
- H SYNC, horizontal synchronization signal. This pulse initiates the horizontal retrace sweep of the beam.
- V SYNC, vertical synchronization signal, initiating the vertical retrace sweep of the beam every 20 ms.

Keyboard Interface

Data communication with the keyboard unit and connected devices is performed via a keyboard interface on the DTC-D board. The line interface consists of:

- Power lines (+5 V, 0 V) to the keyboard.
- A reset keyboard line, used to initialize the keyboard. Reset keyboard is activated by the software via the microcomputer peripheral interface adapter.
- A bidirectional data line used for asynchronous transfers of orders (e.g. alarm on), data (e.g. lamp data) status (e.g. shift key depressed) or key codes/magnetic strip reader data. All transmissions are initiated by the microcomputer on the DTC-D board.

The communication procedure is described in the Keyboard, KBU 4143, KXU 4146, MSR 4136 chapter.

The interface logic consists mainly of an LSI chip, i.e. an asynchronous communication adapter, ACIA, which is connected to the internal data bus. The MPU can thus transmit or receive 8-bit words (programmable) via the ACIA. The ACIA converts the parallel data on the data bus to serial data on the keyboard data line and vice versa.

Two-wire Interface

A general description of the two-wire interface is found in the Communication chapter. The functions are therefore only summarized here. A comprehensive treatment is found in the Detailed Description in this chapter.

By means of the two-wire interface, data can be transferred to or from a buffer in the read/write memory from or to equipment external to the display unit via a two-wire (or coaxial) line. The transmission on the two-wire is in serial format with a bit rate of 300 kHz.

The two-wire interface consists mainly of:

- A direct memory access controller, DMAC, circuit which handles direct memory access, DMA.
- An advanced data link controller, ADLC, which converts 8-bit parallel data to serial data and vice versa. It also supervises the transmission.

- A high speed modem which converts serial data from NRZ format to frequency shifted data and vice versa. It also generates a bit clock and a Clear to send signal.
- A line interface which contains line drivers, line receivers and a line transformer.

The DMAC is connected to work in two channel mode. Channel 0 is used for transmitting and channel 1 for receiving.

Memory Map

The memory map (see Fig. 10) differs somewhat from the description in the Microcomputer chapter.

The total map is divided into a read/write memory (RWM) area, an input/output (I/O) area and an initial program load (IPL) area. The areas are located to the following addresses as seen by the microprocessor:

- RWM; $0000_{(16)}$ – $F6FF_{(16)}$, read and write, $F800_{(16)}$ – $FFFF_{(16)}$, write only.
- I/O; $F700_{(16)}$ – $F7FF_{(16)}$, read and/or write depending on each address.
- IPL; $F800_{(16)}$ – $FFFF_{(16)}$, read only.

The ordinary read/write memory consists physically of 64 kbytes (addresses $0000_{(16)}$ – $FFFF_{(16)}$) but only 56 kbytes ($0000_{(16)}$ – $DFFF_{(16)}$) are used. 8 kbytes ($E000_{(16)}$ – $FFFF_{(16)}$) of memory are supplied in a physically separate memory, called the display memory, although only $5280_{(10)}$ bytes ($E002_{(16)}$ – $F4A1_{(16)}$) are used as proper display memory and the rest substitutes the not used ordinary read/write memory locations.

No parity checking will occur in the RWM and battery backup is not available.

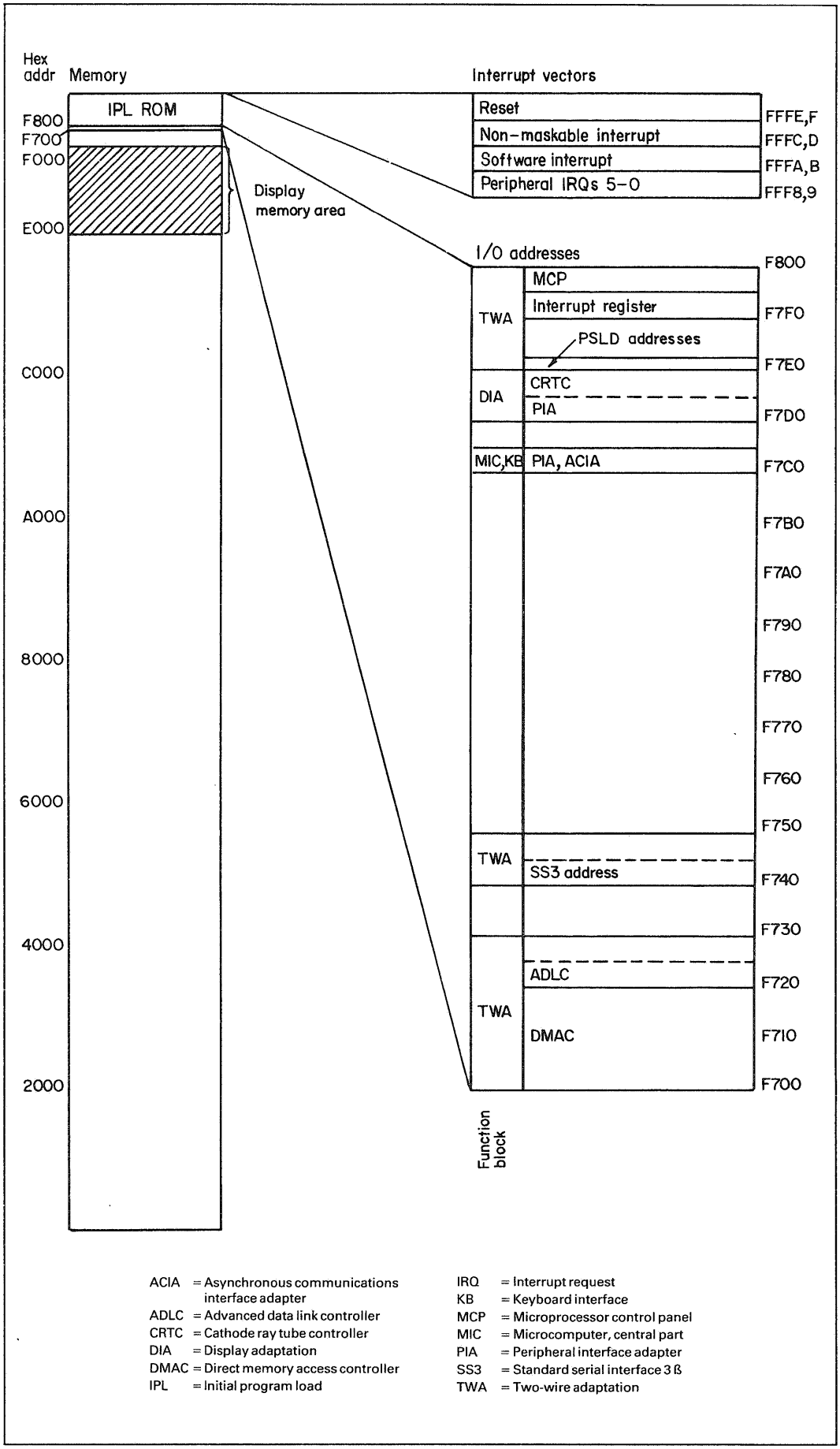
The following circuits and units can be addressed within the I/O area:

- MIC PIA and interrupt register for some central microcomputer functions and MCP for test purposes.
- CRTC and DIA PIA for presentation control.
- KB ACIA for keyboard transfers and control.
- SS3 address register, ADLC and DMAC for two-wire transfers and control.

Detailed I/O addresses are listed in Appendix 1.

The IPL area consists of 2 kbytes of read only memory and contains among other things four interrupt vectors at the addresses $FFF9_{(16)}$ – $FFFF_{(16)}$.

The direct memory access controller, DMAC, can at direct memory access perform read and write operations in the whole read/write area – also in the memory locations which cannot be fully accessed by the microcomputer ($F700_{(16)}$ – $FFFF_{(16)}$).



ACIA = Asynchronous communications interface adapter
ADLC = Advanced data link controller
CRTC = Cathode ray tube controller
DIA = Display adaptation
DMAC = Direct memory access controller
IPL = Initial program load
IRQ = Interrupt request
KB = Keyboard interface
MCP = Microprocessor control panel
MIC = Microcomputer, central part
PIA = Peripheral interface adapter
SS3 = Standard serial interface 3 ß
TWA = Two-wire adaptation

Fig. 10. DU 4113 memory map

Interrupt Handling

The principles of the interrupt system on the DTC-D board are quite different from those described in the Microcomputer chapter. Differences are described below. The following general microcomputer functions are not used:

- Hardware interrupt prioritizing. (The interrupt register, mask register and address modifier are thus left out.)
- Interrupts out.

The locations of the interrupt vectors (i.e. the addresses to the interrupt routines) are found in the memory map.

Reset, Non-maskable and Software Interrupts

The MPU and the microcomputer peripheral interface adapter, MIC PIA, are reset at power on. A general reset is also activated.

The MPU is also reset at a depression of the Reset button. The signal from the Reset button resets only the MPU, but is also fed to the control line CB1 of the MIC PIA. The program can thus check the type of reset (by reading the MIC PIA control register B and testing bit 7, the IRQB 1 flag) and then, if suitable, generate a general reset (by setting and re-setting bit 6 of the MIC PIA peripheral register B). Note that the general reset signal does not reset the MPU or the MIC PIA.

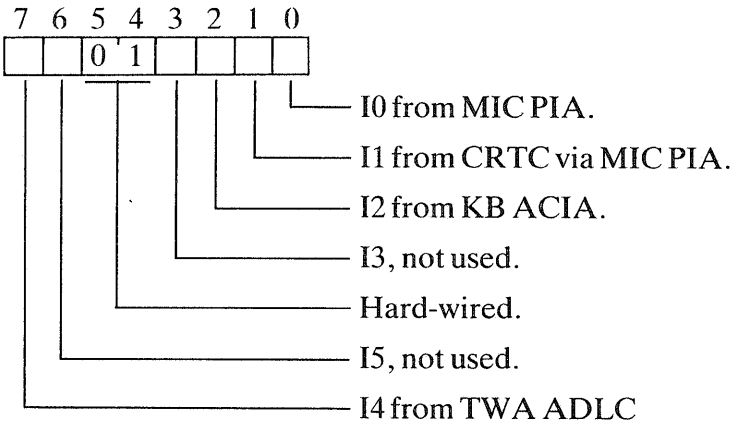
After a reset the MPU will fetch the contents of the memory cells $FFFE_{(16)}$ and $FFFF_{(16)}$ to the program counter and execute a reset program.

NMI, non-maskable interrupt, can only be activated by the microprocessor control panel, MCP, and causes the MPU to fetch the address of the interrupt routine from the cells $FFFC_{(16)}$ and $FFFD_{(16)}$.

SWI, software interrupt, is generated by an MPU instruction and causes the MPU to fetch the address of the interrupt routine from the cells $FFFA_{(16)}$ and $FFFB_{(16)}$.

Interrupt Requests

The interrupt request, IRQ, input of the MPU can be kept deactivated by the test unit MCP (Microprocessor control panel) or be activated by interrupts, I0, I1 etc. The interrupts are OR-ed together. The MPU can check the interrupts by reading an interrupt register (1 = interrupt) at address $F7FO_{(16)}$:



The interrupts are not latched in the interrupt register and must thus be stable until they are handled by the MPU. The interrupts are only serviced if the I-bit of the MPU condition code register is reset. The address to the interrupt routine is located at $FFF8_{(16)}$ and $FFF9_{(16)}$.

The interrupts are further explained below:

I4 originates from the ADLC (advanced data link controller) of the two-wire interface.

I2 originates from the ACIA of the keyboard interface.

I1 originates from the CRTC (cathode ray tube controller) of the presentation logic. It is actually the vertical synchronization (V SYNC) signal fed via the MIC PIA and can be used as a 20 ms timer during presentation.

I0 originates from the CA2 output of the MIC PIA. The program can thus make an interrupt by writing $110_{(2)}$ in bits 5 – 3 of the MIC PIA control register A.

Detailed Description

This section will cover only more complicated circuits or special solutions.

Microcomputer

The microcomputer in DU 4113 differs from the one described in the Microcomputer chapter mainly in the address decoder, the use of the MIC PIA and the interrupt logic. The MPU is also connected somewhat differently.

Basic Timing

A basic 19.17 MHz clock signal is generated by a crystal clock. See Fig. 11. All timing signals, e.g. dynamic memory timing, two-wire transfer bit clock etc. are derived from this clock. Five 2.13 MHz clocks are generated (T1 – T5). Combinations of these are used to define different points of time during each half of the system clock period. The system clock, Ø, (microprocessor instruction clock) has a frequency of 1.065 MHz. It appears in several phases and with different pulse-pause ratios to compensate for propagation delays etc.

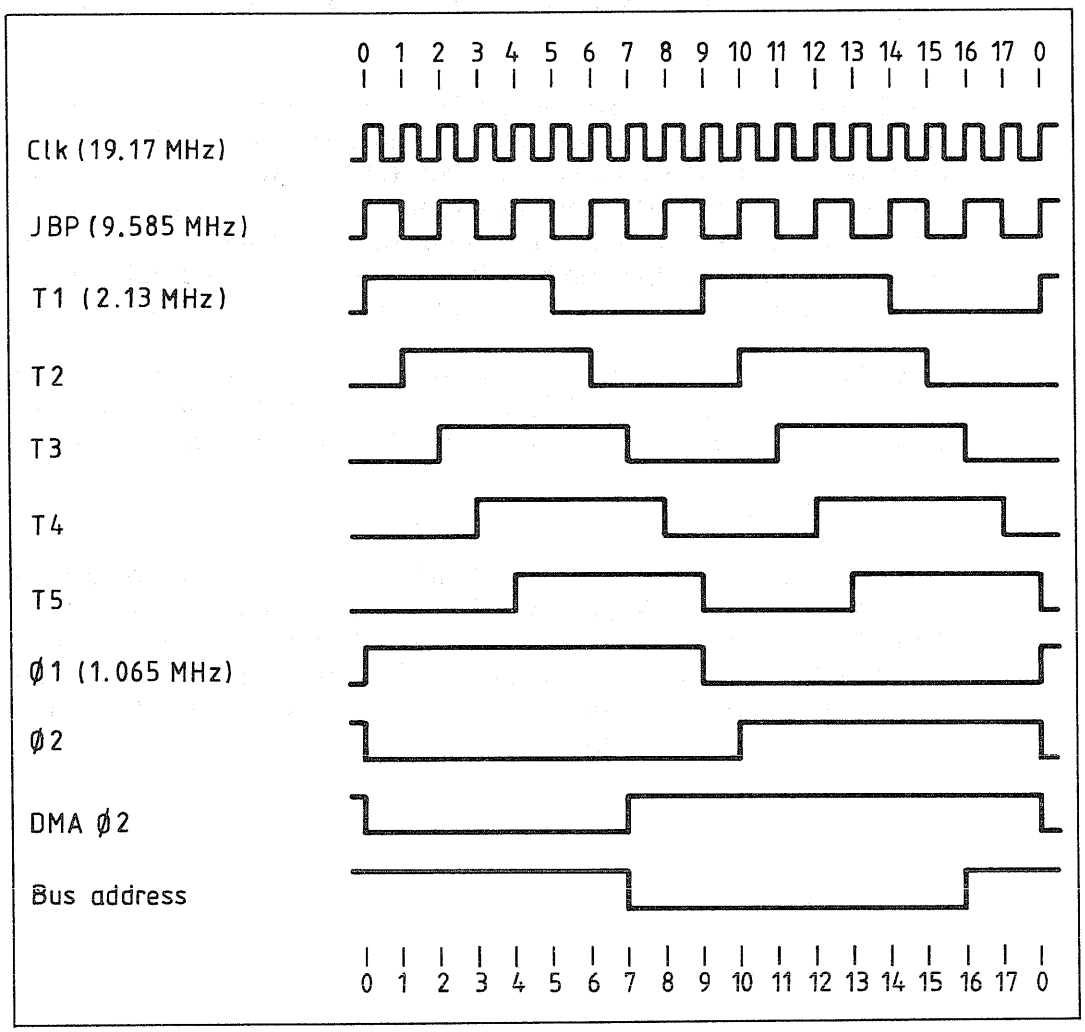


Fig. 11. Basic timing signals

The MPU demands two non-overlapping complementary clocks. Circuitry is added to provide these; MPU Ø1 and MPU Ø2. Observe that they are only fed to the MPU. (MPU Ø2 is also fed to the MCP when it is connected. The MCP or Microprocessor control panel is a test unit.) They differ from the other Ø signals in that they may be frozen in the Ø1 state by a stretch signal (STR). The stretch signal is generated during a direct memory access.

MPU

The NMI (non-maskable interrupt) and HALT inputs of the MPU are only connected to the MCP connector and the TSC (three-state control) input is hard-wired inactive. The address bus and the R/W line are taken to high impedance state by external buffers at direct memory access and the data bus by deactivation of the DBE (data bus enable) input. The DBE input is controlled by the DMA (Direct memory access) signal from the address decoder.

Address Decoding and Direct Memory Access

The principles of the address decoder are as described in the Microcomputer chapter, but the build-up and the signals to and from the address decoder are different. The decoder consists of one FPLA (field programmable logic array) and one PAL (programmable array of logic gates).

The FPLA functions are summarized on the opposite page. The D1 (= DMA grant) and D2 signals are generated as a response to a DMA request from the direct memory access controller, DMAC, and makes it possible for the FPLA to inhibit output signals during DMA cycles 1 – 3 or 2 as well as to generate the DMA and stretch (STR) signals. The rest of the output signals has addressing functions or controls the internal/external data bus buffer (INTERN) or the read/write memory (RWM).

The TWA, F7CD and DMA outputs from the FPLA are then together with Ø2, R/W and AI7 – AI3 (address bits 7 – 3) further decoded in the PAL to more specific addressing signals, summarized on page 28.

See also Memory Map (Fig. 10) and Appendix 1.

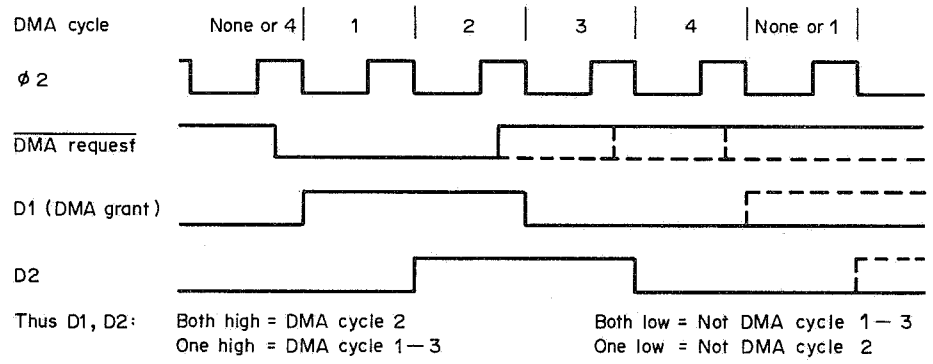
Note that the separations of internal and external address and data buses are not stringent. See Appendix 2. Thus direct memory access from the DMAC affects both the internal address and internal data buses. Note also that the control signals (except ADLCCS) from the address decoder cannot be generated by addresses from the DMAC as the address bus buffers outside the MPU then blocks the address.

Address decoder, FPLA functions

Activating input combinations					Activated output		Comments
A[1] 15–4 Hex	$\overline{VMA}$	D1[2]	D2[2]	R/$\overline{W}$	Signal	Active level	
000–FFF	X	High	High	X	DMA	Low	DMA cycle 2
000–FFF	Low	High	High	X	STR	Low	←DMA cycle 2
F70–F72	Low	One high		X			←DMAC, ADLC addressed
F70–F7F	Low	One low		X	PER I/O	Low	Peripheral I/Os enabled
F70–F72	Low	Low	Low	X	INTERN[3]	Low	←DMAC, ADLC addressed
F74	Low	One low		X			
F7C–F7D	Low	One low		X			
F7F	Low	One low		X			
F80–FFF	Low	One low		High			
F70–F72	Low	Low	Low	X	TWA[4]	High	←DMAC, ADLC addressed
F74	Low	One low		X			
F7E–F7F	Low	One low		X			
F7C–F7D	Low	One low		X	F7CD[4]	High	
F80–FFF	Low	One low		High	F8XXR	Low	IPL ROM Read
000–FFF	High	One low		X	RWM[5]	High[5]	←VMA inactive
F70–F7F	X	One low		X			←I/Os addressed
F80–FFF	X	One low		High			←IPL ROM addressed

[1] A = Address from MPU. Note; not from DMAC.

[2]



[3] Sets the internal/external data bus buffer in (MPU) write position.

[4] See below.

[5] Disables the read/write memory. This is the polarity within the FPLA – the signal is on the output pin and in this description called $\overline{RWM}$ and a low $\overline{RWM}$ (called active) thus enables the read/write memory.

Address decoder, PAL functions

Activating input condition										Activated output		Output activated by address (in hex)
$\overline{Ø2}$	$\overline{DMA}$	TWA	F7CD	R/$\overline{W}$	AI7	AI6	AI5	AI4	AI3	Signal	Active level	
X	X	High	X	X	Low	Low	Low	X	X	DMACS	Low	F700-1F
X or X	Low X	X High	X X	X X	X Low	X Low	X High	X Low	X Low	ADLCCS	Low	F720-7
Low	X	High	X	High	Low	High	Low	Low	Low	SS3	Low	F740-7
X	X	X	High	X	X	X	X	High	X	DIA PIA	High	F7D0-F
X	X	X	High	X	X	High	Low	High	High	CRTCS	Low	F7D8-F
Low	X	High	X	High	High	High	High	High	Low	IRQR	Low	F7F0-7
X	X	High	X	X	High	High	High	High	High	MCP	Low	F7F8-F
X	X	X	X	X	X	X	X	Low	Low	$\overline{A3} \cdot \overline{A4}$	Low	—

Microcomputer PIA, MIC PIA

The MIC PIA controls several central functions. See chapter Microcomputer; Peripheral Interface Adapter, PIA for definitions and a general survey. PA6 – PA4, IRQB, CB2, PB7 and PB5 – PB0 are, however, not used.

The vertical synchronization signal is fed to the CA1 input and will thus set an interrupt flag (CRA bit 7) and activate the IRQA output (if CRA bits 1, 0 = $11_{(2)}$) which results in interrupt 1 to the MPU.

The CA2 output is used to generate interrupt 0 if CRA bit 3 is zero-set (and CRA bits 5, 4 = $11_{(2)}$). Thus an additional software interrupt is available.

The PA outputs/inputs are used in the following way:

PA7 Input Hard-wired low.

PA3 Output 0 = "Monochrome" mode.
1 = Base colour mode.

PA2 Input 0 = MCP connected and in test mode.
1 = MCP not connected or not in test mode.

PA1 Output 0 = Keyboard not reset.
1 = Keyboard reset.

PA0 Input Hard-wired low.

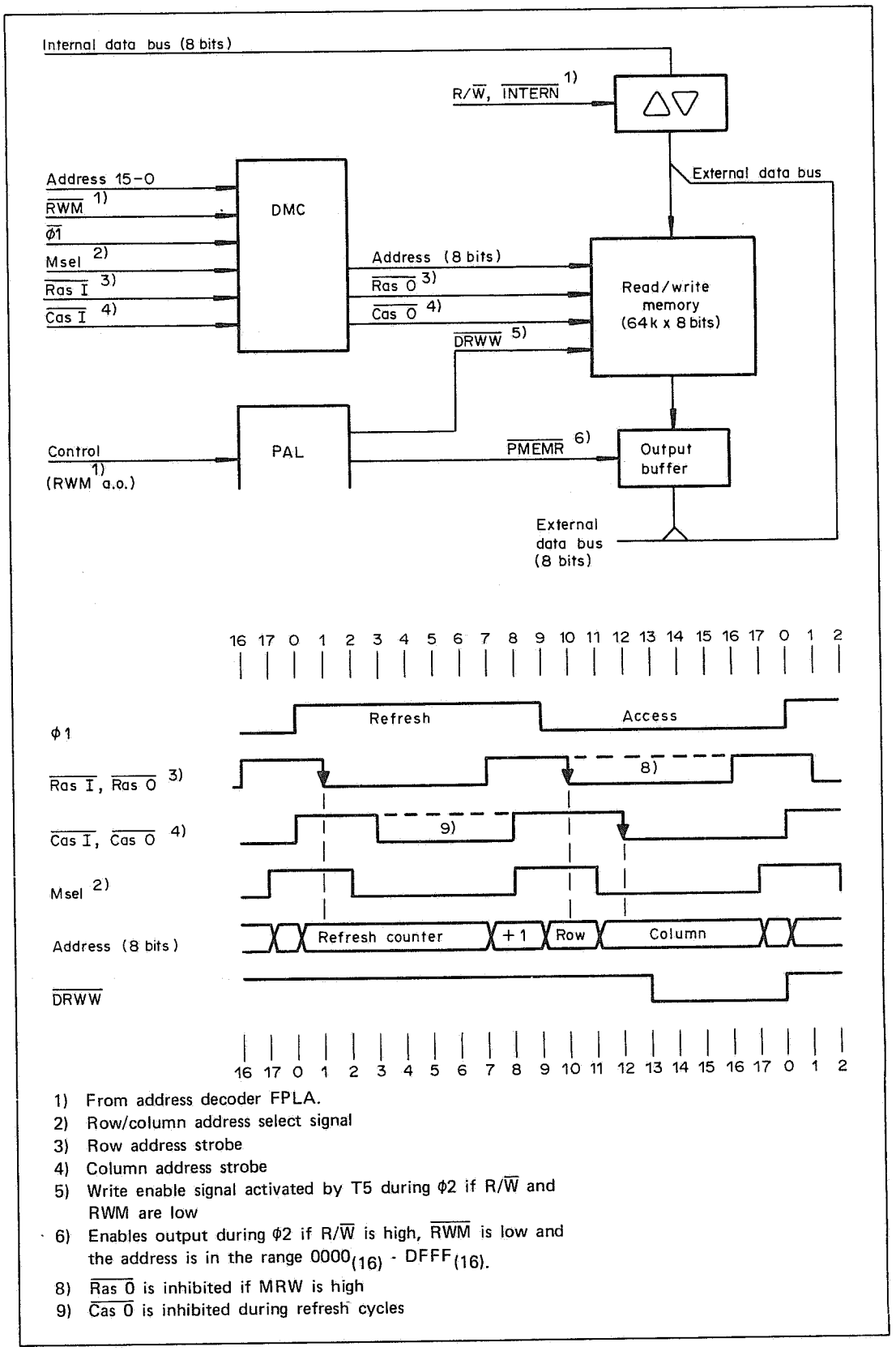
An interrupt flag (CRB bit 7) is set if the Reset button is depressed (as CB1 is then taken low). This must be checked by the program as the IRQB output is not connected.

The PB output is used in the following way:

PB6 1 = Activate general Reset. The output must be set low by software after power on. The MPU and the MIC PIA are not reset by general Reset.

Read/Write Memory

The read/write memory consists of 65536 × 8 bits of dynamic random access memory. See Fig. 12. Input data is fed from the external data bus and output data is fed to the same bus via an output buffer.



1) From address decoder FPLA.
2) Row/column address select signal
3) Row address strobe
4) Column address strobe
5) Write enable signal activated by T5 during Φ2 if R/$\overline{W}$ and RWM are low
6) Enables output during Φ2 if R/$\overline{W}$ is high, $\overline{RWM}$ is low and the address is in the range $0000_{(16)}$ - $DFFF_{(16)}$.
8) $\overline{Ras\ 0}$ is inhibited if MRW is high
9) $\overline{Cas\ 0}$ is inhibited during refresh cycles

Fig. 12. Read/Write memory logic and timing

The MPU cannot read from any of the addresses $E000_{(16)}$ – $FFFF_{(16)}$ as the output buffer is then taken to high impedance state by the PMEMR signal. The MPU cannot write at $F700_{(16)}$ – $F7FF_{(16)}$ as the write enable signal, DRWW, is then blocked by the RWM signal (see Address Decoding and Direct Memory Access). The area $E000_{(16)}$ – $FFFF_{(16)}$ is accessible for write, but not for read operations at direct memory access as the output buffer is taken to high impedance state by these addresses.

The programming of the PAL in Fig. 12 is summarized below:

Activating input condition							Activated output		Comments
Ø2	T5	$\overline{RWM}$	R/$\overline{W}$	AI15	AI14	AI13	Signal	Active level	
High	High	Low	Low				DRWW	Low	Write
High or High or High		Low Low Low	High High High	Low	 Low 	 Low	PMEMR	Low	Read from addresses <$E000_{(16)}$

The read/write memory is organized in 256 rows and 256 8-bit-columns. A dynamic memory controller, DMC, splits the 16-bit address in two 8-bit parts. One part is applied as row address when a select signal, Msel, is high and clocked into the memory by a row address strobe, the other is applied as column address when the select signal is low and clocked by a column address strobe. The combined row and column addresses define one memory location. The row address strobe can be inhibited by the RWM signal and memory operations are then disabled. The above functions will only take place when a refresh control input of the DMC is kept high. This input is tied to Ø1 (inverted) and a memory access can thus only be made during Ø1 low time.

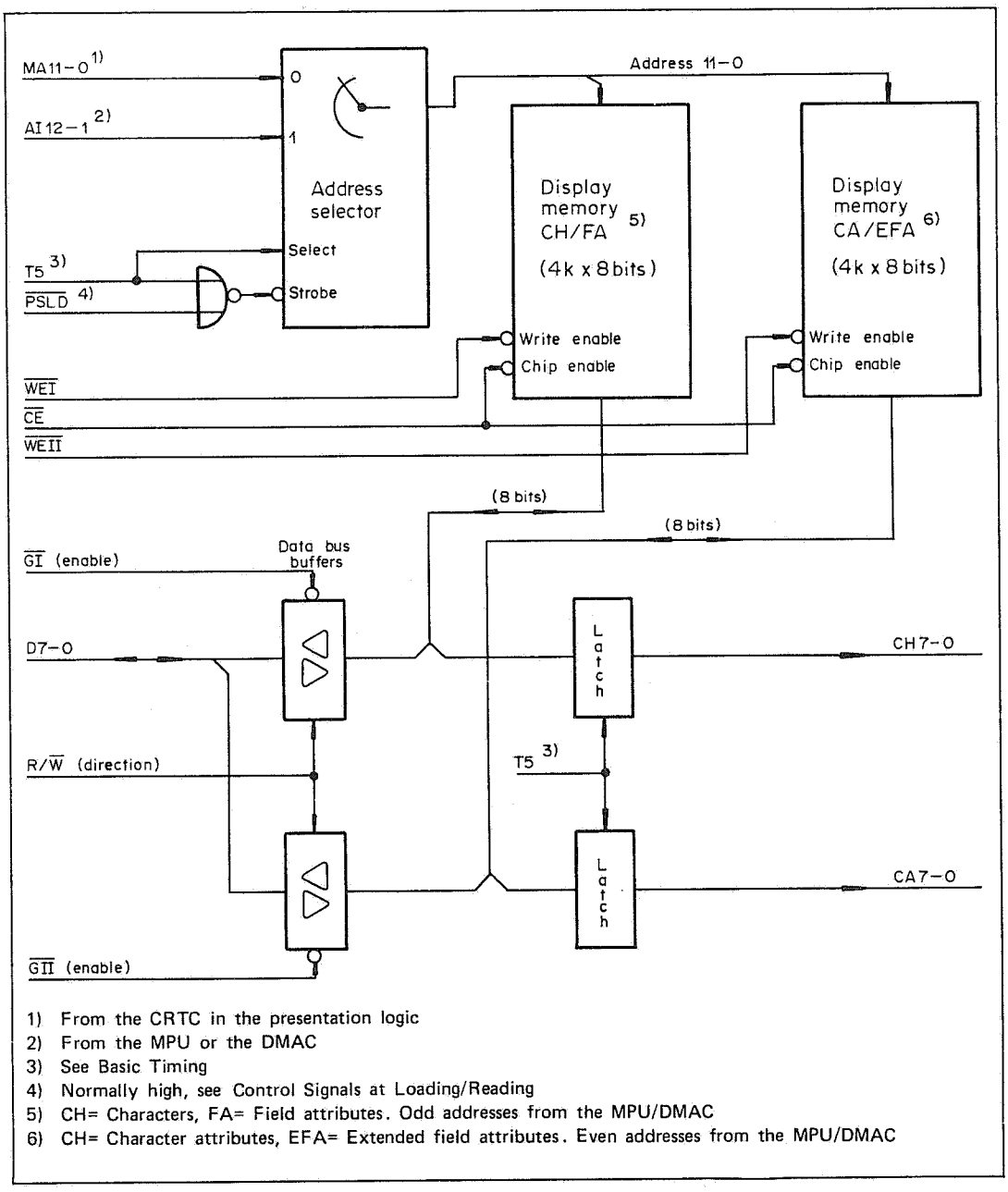
The DMC will supply the inverted contents of an internal 8-bit refresh counter on its address outputs when the refresh control input is low. The counter is incremented in each refresh cycle at low-to-high transition of the row address strobe to the DMC. The row address strobe is propagated to the memory. Only row addressing is necessary in order to obtain the refresh function.

To sum up: during Ø1 high time the address originates from the refresh counter in the DMC. During Ø1 low time the address may originate from the MPU or the direct memory access controller, DMAC. The Row address strobe may, however, be inhibited by the RWM signal from the address decoding FPLA which will delete a memory operation.

Although write operations may occur in the $E000_{(16)}$ – $FFFF_{(16)}$ area of the physical read/write memory the effective area above $DFFF_{(16)}$ is physically located to the display memory blocks (see below), the IPL ROM and the I/O devices.

Display Memory

The display memory circuitry is shown in Fig. 13 and the timing in Fig. 14. The display memory consists of one 4096 × 8 bits block for character (CH) and field attribute (FA) codes and another for character attribute (CA) and extended field attribute (EFA) codes. The address to the memory can, by means of an address selector, be supplied by the MPU or the DMAC (direct memory access controller) on the AI12 – 1 inputs of the address selector or by the CRTC (cathode ray tube controller) of the presentation logic on the MA11 – 0 inputs. The PSLD signal functions at character generator loading/reading and is normally inactive.



1) From the CRTC in the presentation logic
2) From the MPU or the DMAC
3) See Basic Timing
4) Normally high, see Control Signals at Loading/Reading
5) CH= Characters, FA= Field attributes. Odd addresses from the MPU/DMAC
6) CH= Character attributes, EFA= Extended field attributes. Even addresses from the MPU/DMAC

Fig. 13. Display memory logic

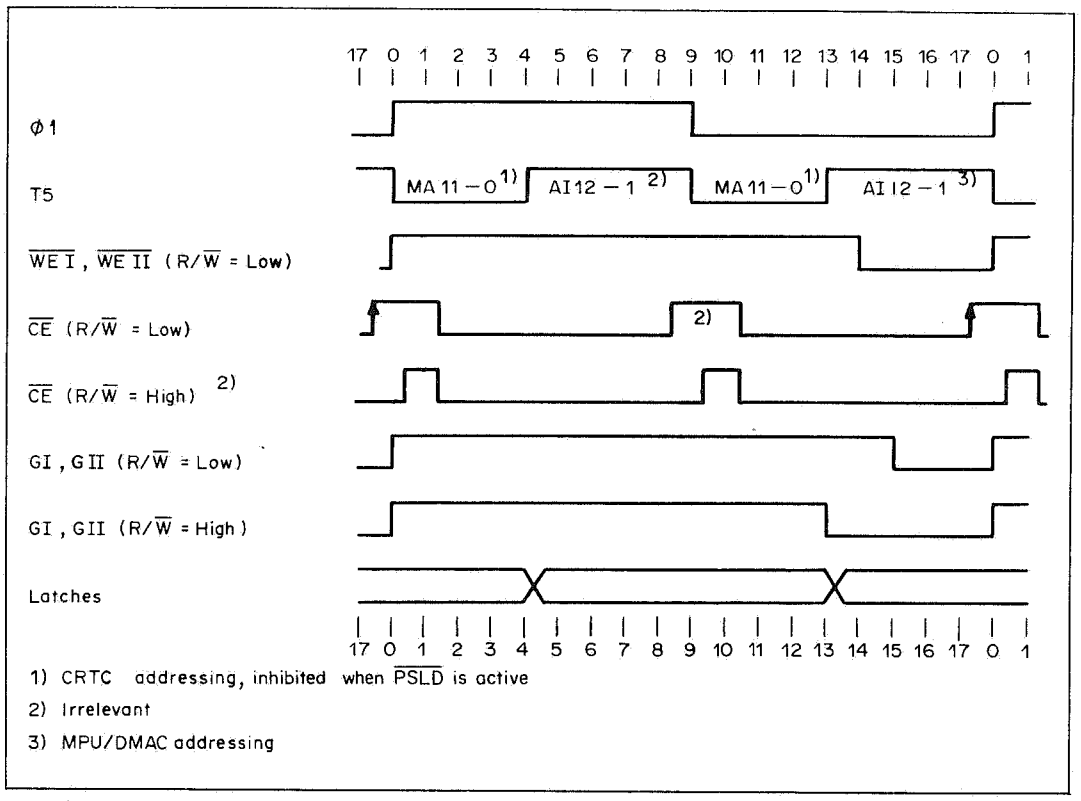


Fig. 14. Display memory timing

MPU/DMAC Display Memory Access

The MPU/DMAC address is supplied to both memory blocks during T5 (see Basic Timing) but the addressing during Ø1 high time is irrelevant (due to absence of write enable and closed bus buffers, see timing and below). At write operations in the $E000_{(16)}$ – $FFFF_{(16)}$ area a write enable signal, WE II, is sent to the CA/EFA-block if address bit 0 =0 or another write enable signal, WE I, to the CH/FA-block if address bit 0 = 1. Thus CAs/EFAs must be located at even addresses and CHs/FAs at odd addresses. The only relevant edge on the chip enable signal, CE, is the low-to-high transition during Ø1 low time when R/W is low. This is the actual write pulse. Note, however, that it is impossible for the MPU to write in the area $F700_{(16)}$ – $F7FF_{(16)}$ due to the RWM signal (see table below).

Data is sent to or from the data bus via two buffers, the upper one in Fig. 13 is for CHs/FAs and the lower one for CAs/EFAs. The enable signal G II is activated by even addresses in the area $E000_{(16)}$ – $FFFF_{(16)}$. The other enable signal, G I, is activated by odd addresses in the same area. The transfer directions through the buffers are controlled by the read/write signal. That the G I and G II timing differs in the read and write cases depends on bus timing considerations. Note, however, that it is impossible for the MPU to read from the area $F700_{(16)}$ – $FFFF_{(16)}$ due to the RWM signal (see table below).

The WE I, WE II, G I and G II signals are generated in a PAL, which functions are summarized below:

Activating input condition										Activated output		Comments (Addresses in hex)
Time						Address						
Ø2	T1	T2	T5	$\overline{RWM}$	R/$\overline{W}$	AI15	AI14	AI13	AI0	Signal	Active level	
High	Low			Low	Low	High	High	High	High	WE I	Low	Write odd E001 – FFFF
High	Low			Low	Low	High	High	High	Low	WE II	Low	Write even E000 – FFFE
High or High		Low	High High	Low Low	High	High High	High High	High High	High High	G I	Low	Read odd E001 – FFFF Write odd E001 – FFFF
High or High		Low	High High	Low Low	High	High High	High High	High High	Low Low	G II	Low	Read even E000 – FFFE Write even E000 – FFFE

CRTC Display Memory Access

The CRTC address, MA11 – 0, is supplied to both memory blocks during T5 low time. The blocks will thus provide one CH or FA code *and* one CA or EFA code twice each instruction clock cycle, e.g. $000_{(16)}$ from the CRTC will read MPU/DMAC addresses $E000_{(16)}$ and $E000_{(16)}$, $FFF_{(16)}$ from the CRTC will read MPU/DMAC addresses $FFFE_{(16)}$ and $FFFF_{(16)}$. The two codes are latched at the beginning of T5 and available to the presentation logic one full T5 period, i.e. one half instruction clock cycle.

Display Memory Proper

Only the $E002_{(16)}$ – $F6FF_{(16)}$ area of the physical display memory blocks is defined as the display memory. The addresses $E000_{(16)}$ and $E001_{(16)}$ are reserved for special functions at character generator loading (see below). There is also a soft reservation of the $F4A2_{(16)}$ – $F6FF_{(16)}$ area for some OS routines. The addresses $F700_{(16)}$ – $F7FF_{(16)}$ are blocked for the MPU as being the I/O area. Although the MPU is unable to read from the $F800_{(16)}$ – $FFFF_{(16)}$ area, this area can be seen as an ordinary read/write memory area.

The memory organization is summarized in Fig. 15.

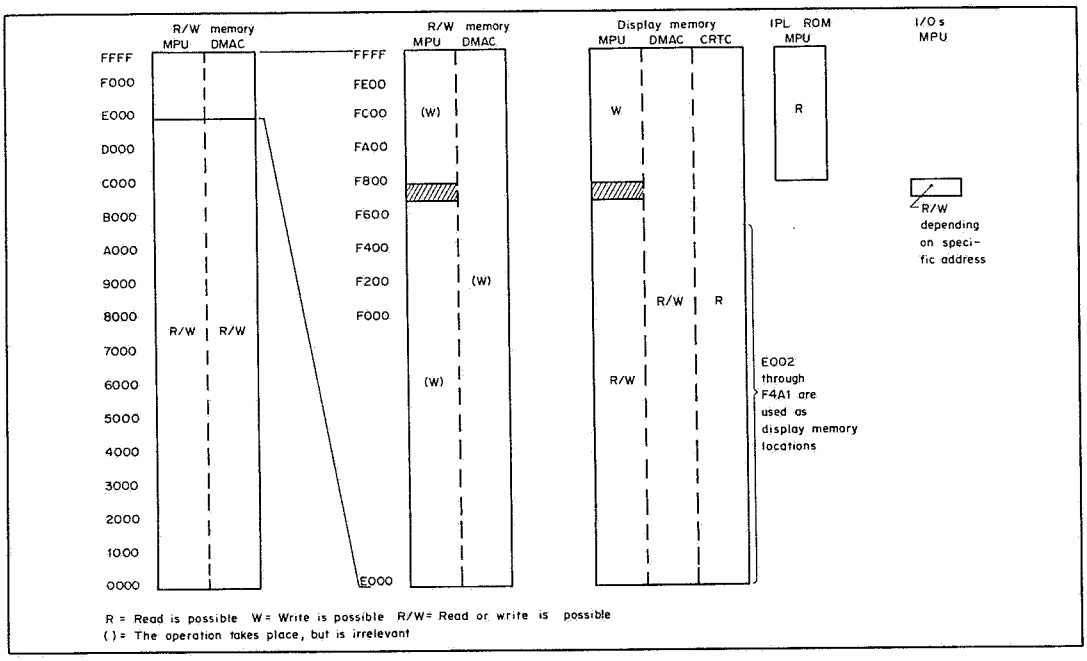


Fig. 15. Memory addressing, sources and destinations

Character Generators

As the organization, loading and reading of the character generators are more complicated than their normal operation, these matters are treated separately below.

Circuitry

Fig. 16 shows one character generator block. The reduced version contains one such block (corresponding to the blue block in the full version) and the full version contains three such blocks called the blue, red or green block. (Note that there is one video register for each block.) Each block contains 16384 × 9 bits of random access memory. The circuitry of one block is described below.

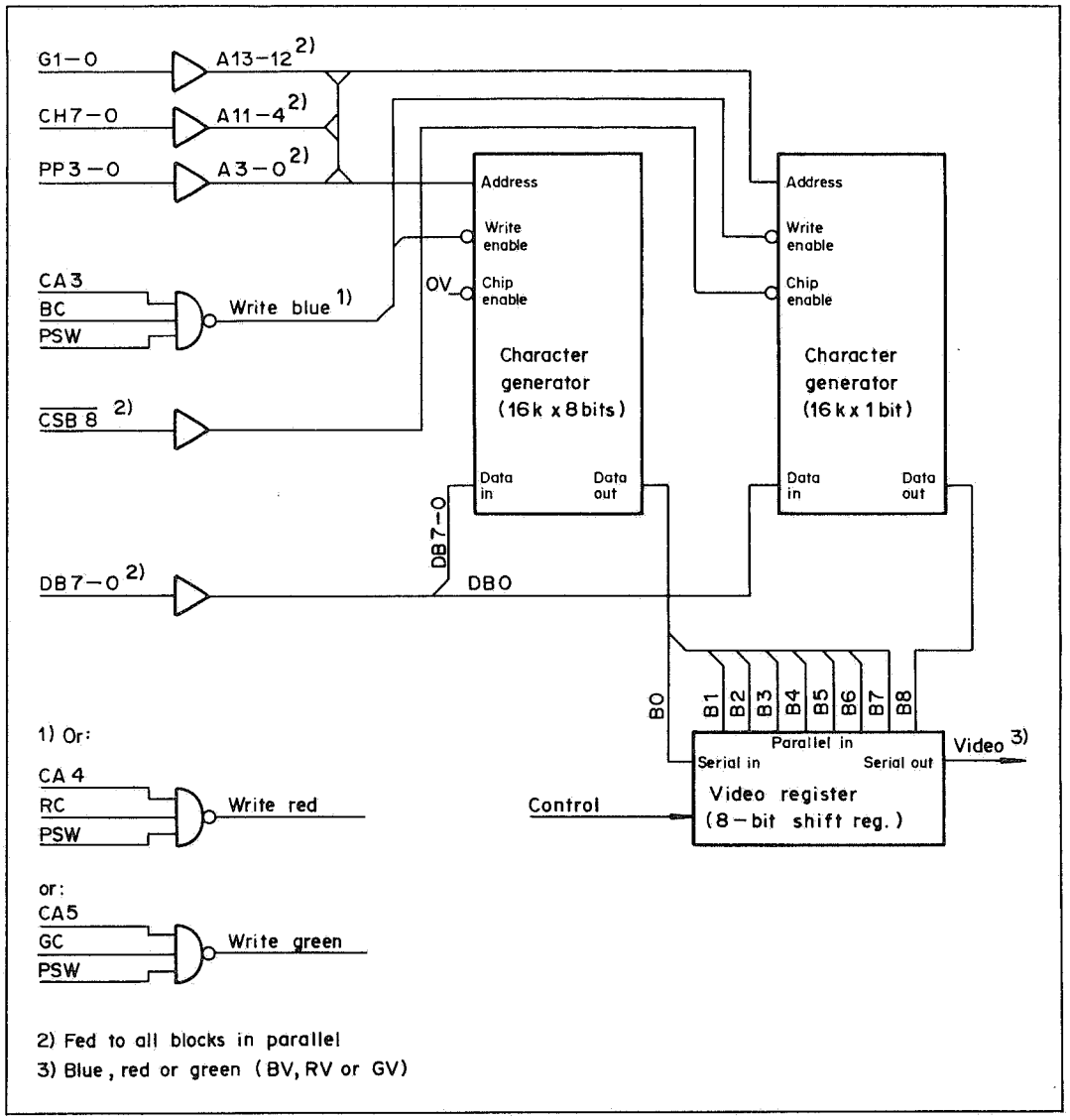


Fig. 16. Character generator logic

One write and 14 address lines are connected to all memory chips (one chip for each one of the 9 bits) but one of the chips (character generator bit 8) can be disabled by a CSB8 signal. All chips are enabled at presentation and a full 9-bit location will then be fed to the video register at each addressing.

The data bus is fed to the data inputs of the memory chips. The alignment of the data bus bits, character generator bits and character matrix column is shown below:

Data bus bit:	0	7	6	5	4	3	2	1	0
Character generator bit:	8	7	6	5	4	3	2	1	0
Column:	0	1	2	3	4	5	6	7	8

It is possible to write all character generator bits simultaneously if character generator bits 8 and 0 are to be equal. In other cases it is necessary to first write character generator bit 8 with a relevant data bit 0 and then overwrite character generator bits 7 – 0 with the bit 8 chip disabled by the CSB8 signal.

Organization

The address is formatted as follows:

	"Store"		Character code								Modified raster address			
Signal:	G		CH								PP			
	1	0	7	6	5	4	3	2	1	0	3	2	1	0
Address:	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The raster address (RA3 – 0) originates during presentation from the cathode ray tube controller and during character generator loading/reading from a raster address counter. The address is then modified (to PP3 – 0) in a ROM. At loading/reading the modifications depend on character generator version, IBM or non-IBM mode and a signal G2. The ROM also generates three signals, BC, RC and GC, denoting the actual block (a write enable condition) and a triple plane signal, TP. The ROM program is shown in Appendix 3. The G1 – 0 signals denote store 3 – 0 and in IBM mode store 7 – 4 if G2 = 1.

A single plane store or one plane in a triple plane store is basically (Not at 8 stores in IBM mode.) arranged as shown in Fig. 17. Figs 18 and 19 are summaries of the character generator allocations at loading in IBM and non-IBM modes. The RA figures denote the raster address as supplied by the raster address counter and then transformed to the modified raster address, PP. Note that one block contains four single plane stores or one plane of four triple plane stores except in the IBM mode case with 8 stores. In the latter case the single plane stores 0 to 3 are allocated to the space PP3 – 1 = $C_{(16)} - F_{(16)}$, which is not used by the triple plane stores 4 to 7, e.g. raster address RA0 – 3 of store 0 is allocated to PP3 – 1 = $C_{(16)} - F_{(16)}$ in the blue block RA4 – 7 of store 0 to PP3 – 1 = $C_{(16)} - F_{(16)}$ in the red block etc.

It is necessary to reload the character generators at change of mode as the character generators are organized in different ways in the different modes.

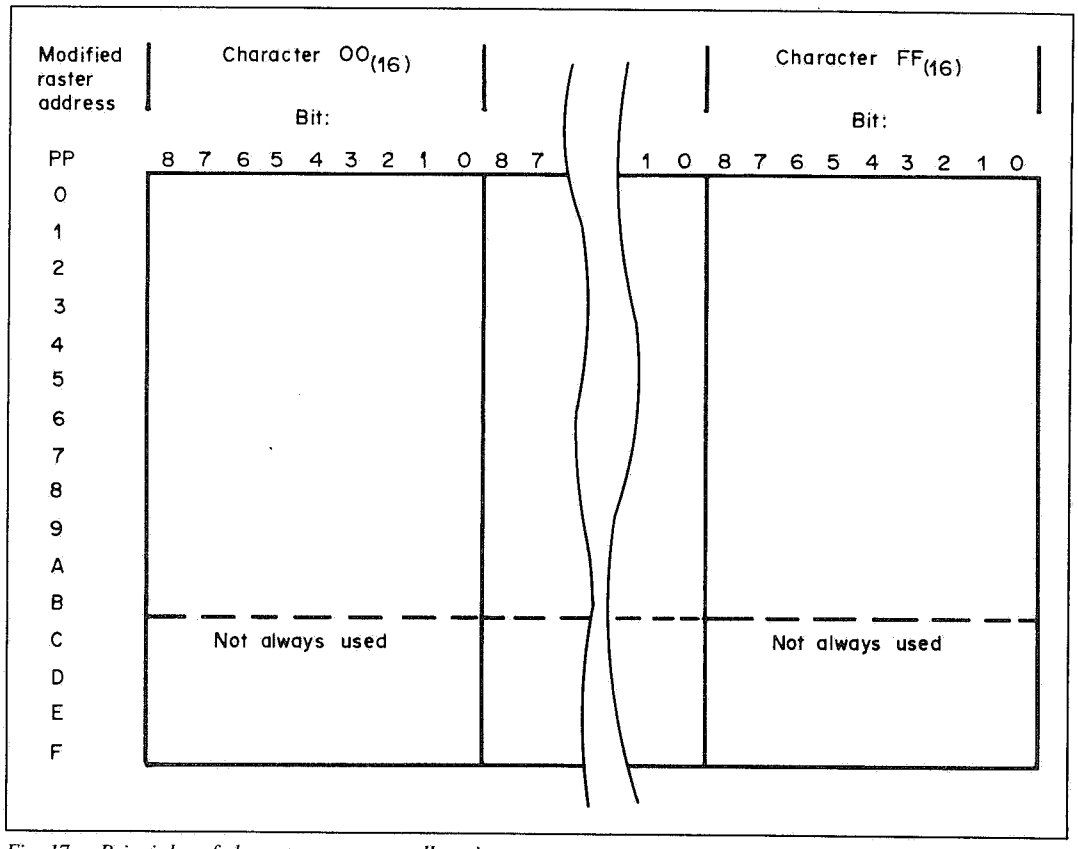


Fig. 17. Principles of character generator allocations

Control Signals at Loading/Reading

Some control signals are generated from a PAL when $F7E0_{(16)}$ through $F7E3_{(16)}$ are addressed. They are summarized below and will be referred to under the two following headings.

Activating input condition										Activated output		Comments (Addresses in hex)
$\overline{Ø1}$	T4	TWA[1]	PSLD[2]	R/$\overline{W}$	AI7	$\overline{A4 \cdot A3}$	AI2	AI1	AI0	Signal	Active level	
High		High		Low	High	Low	Low	Low	Low	PSLDON[2]	High	F7E0, Write[3]
High		High		Low	High	Low	Low	Low	High	PSLDOFF[2]	High	F7E1, Write[3]
High	High	High	High	Low	High	Low	Low	High		PSW	High	F7E2 – 3, Write, PSLD[3]
High		High	High	High	High	Low	Low	Low		PSRD	High	F7E0 – 1, Read, PSLD[3]
High		High	High		High	Low	Low	High		PSADR	High	F7E2 – 3, PSLD[3]
High or		High	Low	High	High	Low	Low	Low	Low	LDEN	High	F7E0, Read[3] or not PSLD
High or		High	High Low	High	High	Low	Low	Low		CKEN	High	F7E0 – 1, Read, PSLD[3] or not PSLD
			High						Low	CSB8	High	AI0 = Low, PSLD

[1] TWA originates from the address decoding FPLA.

[2] PLSD is set by PSLDON and reset by PSLDOFF.

[3] Not at DMA.

		Full version						Reduced version	
		Blue block		Red block		Green block		(Blue block)	
G1 – 0	PP3 – 0	Character 00 – FF		Character 00 – FF		Character 00 – FF		Character 00 – FF	
0	0	Store 4	RA 0	Store 4	RA 0	Store 4	RA 0	Store 0	RA 0
	1	G2 = 1	1	G2 = 1	1	G2 = 1	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C	Store 0	RA 0	Store 0	RA 4	Store 0	RA 8	Not used	
	D	G2 = 0	1	G2 = 0	5	G2 = 0	9		
	E	BC = 1	2	RC = 1	6	GC = 1	A		
	F		3		7		B		
1	0	Store 5	RA 0	Store 5	RA 0	Store 5	RA 0	Store 1	RA 0
	1	G2 = 1	1	G2 = 1	1	G2 = 1	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C	Store 1	RA 0	Store 1	RA 4	Store 1	RA 8	Not used	
	D	G2 = 0	1	G2 = 0	5	G2 = 0	9		
	E	BC = 1	2	RC = 1	6	GC = 1	A		
	F		3		7		B		
2	0	Store 6	RA 0	Store 6	RA 0	Store 6	RA 0	Store 2	RA 0
	1	G2 = 1	1	G2 = 1	1	G2 = 1	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C	Store 2	RA 0	Store 2	RA 4	Store 2	RA 8	Not used	
	D	G2 = 0	1	G2 = 0	5	G2 = 0	9		
	E	BC = 1	2	RC = 1	6	GC = 1	A		
	F		3		7		B		
3	0	Store 7	RA 0	Store 7	RA 0	Store 7	RA 0	Store 3	RA 0
	1	G2 = 1	1	G2 = 1	1	G2 = 1	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C	Store 3	RA 0	Store 3	RA 4	Store 3	RA 8	Not used	
	D	G2 = 0	1	G2 = 0	5	G2 = 0	9		
	E	BC = 1	2	RC = 1	6	GC = 1	A		
	F		3		7		B		

All figures are in hexadecimal.

Fig. 18. Detailed IBM mode character generator maps

G1 – 0	PP3 – 0	Full version: Blue block Character 00 – FF	RA 0	Full version: Red block Character 00 – FF	RA 0	Full version: Green block Character 00 – FF	RA 0	Reduced version (Blue block) Character 00 – FF	RA 0
0	0	Store 0	0	Store 0	0	Store 0	0	Store 0	0
	1	G2 = 0	1	G2 = 0	1	G2 = 0	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C		C		C		C		C
	D		D		D		D		D
	E		E		E		E		E
	F		F		F		F		F
1	0	Store 1	0	Store 1	0	Store 1	0	Store 1	0
	1	G2 = 0	1	G2 = 0	1	G2 = 0	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C		C		C		C		C
	D		D		D		D		D
	E		E		E		E		E
	F		F		F		F		F
2	0	Store 2	0	Store 2	0	Store 2	0	Store 2	0
	1	G2 = 0	1	G2 = 0	1	G2 = 0	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C		C		C		C		C
	D		D		D		D		D
	E		E		E		E		E
	F		F		F		F		F
3	0	Store 3	0	Store 3	0	Store 3	0	Store 3	0
	1	G2 = 0	1	G2 = 0	1	G2 = 0	1	G2 = 0	1
	2	BC = 1	2	RC = 1	2	GC = 1	2	BC = 1	2
	3	CA3 = 1	3	CA4 = 1	3	CA5 = 1	3	CA3 = 1	3
	4		4		4		4		4
	5		5		5		5		5
	6		6		6		6		6
	7		7		7		7		7
	8		8		8		8		8
	9		9		9		9		9
	A		A		A		A		A
	B		B		B		B		B
	C		C		C		C		C
	D		D		D		D		D
	E		E		E		E		E
	F		F		F		F		F

All figures are in hexadecimal.

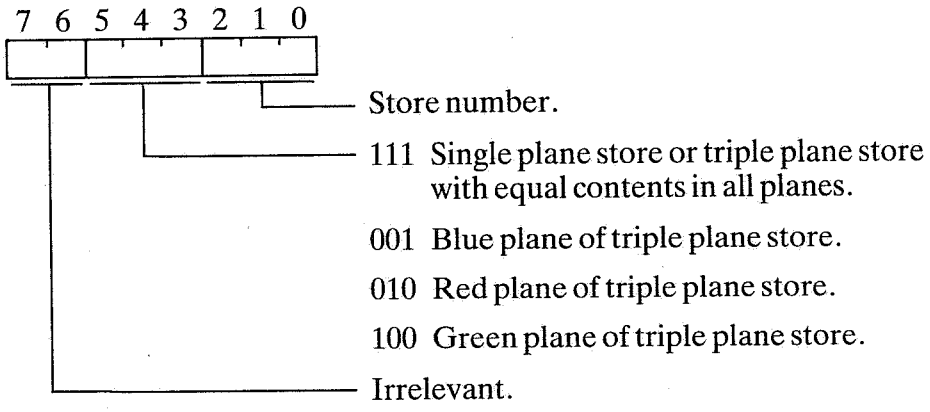
Fig. 19. Detailed non-IBM mode character generator maps

Loading

A dummy write at address $F7E0_{(16)}$ generates a PSLDON signal, which resets the raster address counter and sets a flip-flop generating a PSLD signal. The PSLD signal turns off the presentation and conditions the hardware for character generator loading/reading. A dummy write at address $F7E1_{(16)}$ resets the PSLD signal and restores the hardware after loading/reading of the character generators.

Note in Fig. 13 that the address selector of the display memory is disabled during T5 low time by the PSLD signal and that the selector outputs then are low i.e. addressing $(E)000_{(16)}$ and $(E)001_{(16)}$. The contents of these addresses are then latched when T5 goes high and put on the CA and CH lines. The MPU/DMA access to the display memory is not affected.

Address $E000_{(16)}$ controls at loading the CA5 – 3 signals and via CA2 – 0 the G2 – 0 signals (see Fig. 16) and must be set as follows:



The code of the character to be loaded must be written at address $E001_{(16)}$ as the CH7 – 0 signals used in the character generator for addressing originate from this source. It is impossible to address the character generators directly (see Fig. 16 and under Organization).

When the above preparations have been executed a character file of up to 16 locations can be written.

Writing a byte to address $F7E2_{(16)}$ (PSADR0) stores the byte in bits 7 – 0 of the 9-bit character generator word that is addressed by the raster address counter. Bit 0, which is stored in bit 0 of the 9-bit character generator word, is, however, also stored in bit 8 of the latter. (0 = dot off, 1 = dot on.) Finally, the raster address counter is incremented one step (count cycle: 0, 1, 2, ... E, F, 0, 1 ...).

Writing a byte to address $F7E3_{(16)}$ (PSADR1) operates exactly as above, with the exception that bit 8 of the 9-bit word is not affected, as the CSB8 signal does not then enable the bit 8 chip of the character generator.

A write operation at the addresses $F7E2 - 3_{(16)}$ when PSLD is set generates the control signals PSW, which enables writing and PSADR, which increments the raster address counter. A read operation only generates PSADR, thus enabling incrementation of the raster address counter.

Note that it is best to write 16 times to character generator bits 8 – 0 and then – if bits 8 and 0 are to be different – 16 times to character generator bits 7 – 0. Note also that the raster address counter must be incremented to $F_{(16)}$ or reset if less than 16 locations are written in one character file before another file is written. Then it is only necessary to change the contents of address $E001_{(16)}$ in order to write another character file in the same store or plane of a store.

Reading

Although not required for normal operation it is possible to read the contents of the character generators in a bit-by-bit manner. This is primarily intended for test purposes, but it may also be used to determine whether or not the device contains the full character generator version.

The read facility is enabled in the same way as the load facility, i.e. by a dummy write at the address $F7E0_{(16)}$ (PSLDON). It is disabled by a dummy write at the address $F7E1_{(16)}$ (PSLDOFF). The 9-bit character generator words are also addressed in the same way, i.e. by the contents of the addresses $E000_{(16)}$ and $E001_{(16)}$ in the display memory, but the bits 7 – 3 of location $E000_{(16)}$ are now irrelevant. The raster address counter can also just be stepped by readings of $F7E2_{(16)}$ or $F7E3_{(16)}$. Note that this may be necessary in order to reach the applicable raster address.

The control signals to the video register are shown in Fig. 20. During presentation (PSLD is low; LDEN and CKEN are both high) the register is loaded twice each system clock cycle and shifted at 19.17 MHz.

A dummy write operation at the address $F7E0_{(16)}$ (PSLDON) resets the video register and the now set PSLD signal inhibits both parallel load and clocking of the video register as LDEN and CKEN are held low. A following read operation at the address $F7E0_{(16)}$ will, however, activate the LDEN and CKEN signals. The LDEN signal then enables parallel loading and the CKEN signal allows one clock pulse, which will load the video register with the contents of the addressed character generator location (except bit 0). Additional read operations at the address $F7E1_{(16)}$ activates only the CKEN signal, which will allow one clock pulse to shift the video register one step for each reading. (The first will shift in bit 0.)

Each read operation at $F7E0_{(16)}$ or $F7E1_{(16)}$ when PSLD is set generates a PSRD signal, which activates a three-state buffer feeding the data bus with the following signals:

Signal:	None		GC	RC	BC	GV	RV	BV
Data bus bit:	7	6	5	4	3	2	1	0

The GC, RC and BC signals were described under Organization and the GV, RV and BV signals are the outputs of the green, red and blue video register respectively.

To sum up: Let $C_8 - C_0$ denote bits number 8 – 0 of the addressed character generator location if it belongs to a single plane store and let $C_{G8} - C_{G0}$, $C_{R8} - C_{R0}$ and $C_{B8} - C_{B0}$ denote the bits of the word in the green, red and blue plane, respectively, in the case of a triple plane store. Read

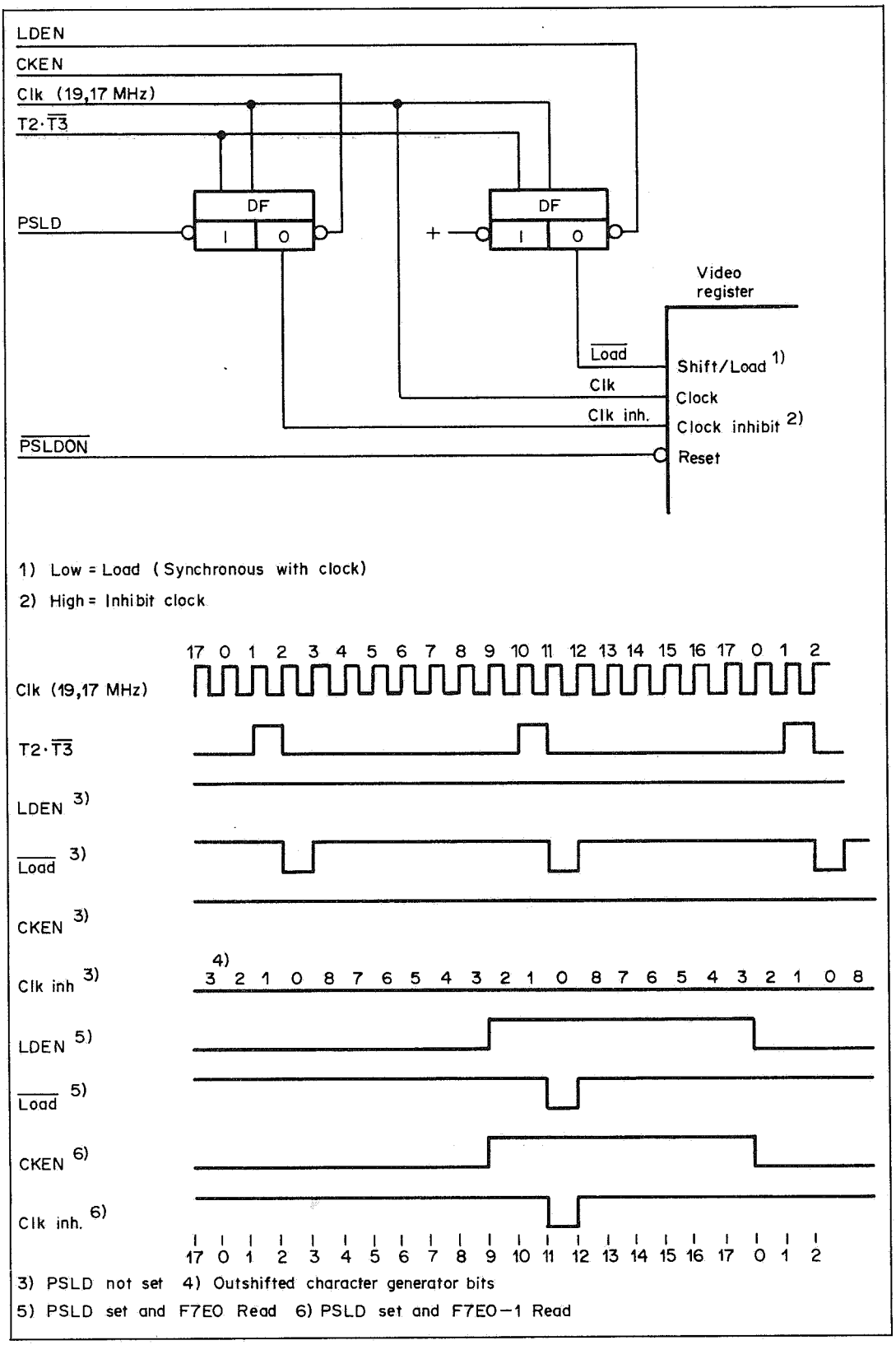


Fig. 20. Video register control and timing

operations at the addresses $F7E0_{(16)}$ and then $F7E1_{(16)}$ will (without any of the effects caused by writing to these addresses) yield a byte according to the table below in which X denotes undefined bits and n = 8 when $F7E0_{(16)}$ is red, 7 the first time $F7E1_{(16)}$ is red, 6 the second time $F7E1_{(16)}$ is red, etc.

Data bus bit								Comments
7	6	5	4	3	2	1	0	
X	X	1	1	1	C_{Gn}	C_{Rn}	C_{Bn}	Triple plane store
X	X	1	0	0	C_n	X	X	Single plane store
X	X	0	1	0	X	C_n	X	Single plane store
X	X	0	0	1	X	X	C_n	Single plane store
X	X	0	0	0	X	X	X	The addressed word does not exist. (Applies to store number 4 – 7 in two cases: when the device lacks the full character generator option or is in non-IBM mode. It also applies to scan line $C_{(16)} - F_{(16)}$ in IBM mode.)

Cathode Ray Tube Controller, CRTC

The presentation on the screen is supervised by the cathode ray tube controller (CRTC) shown in Fig. 21.

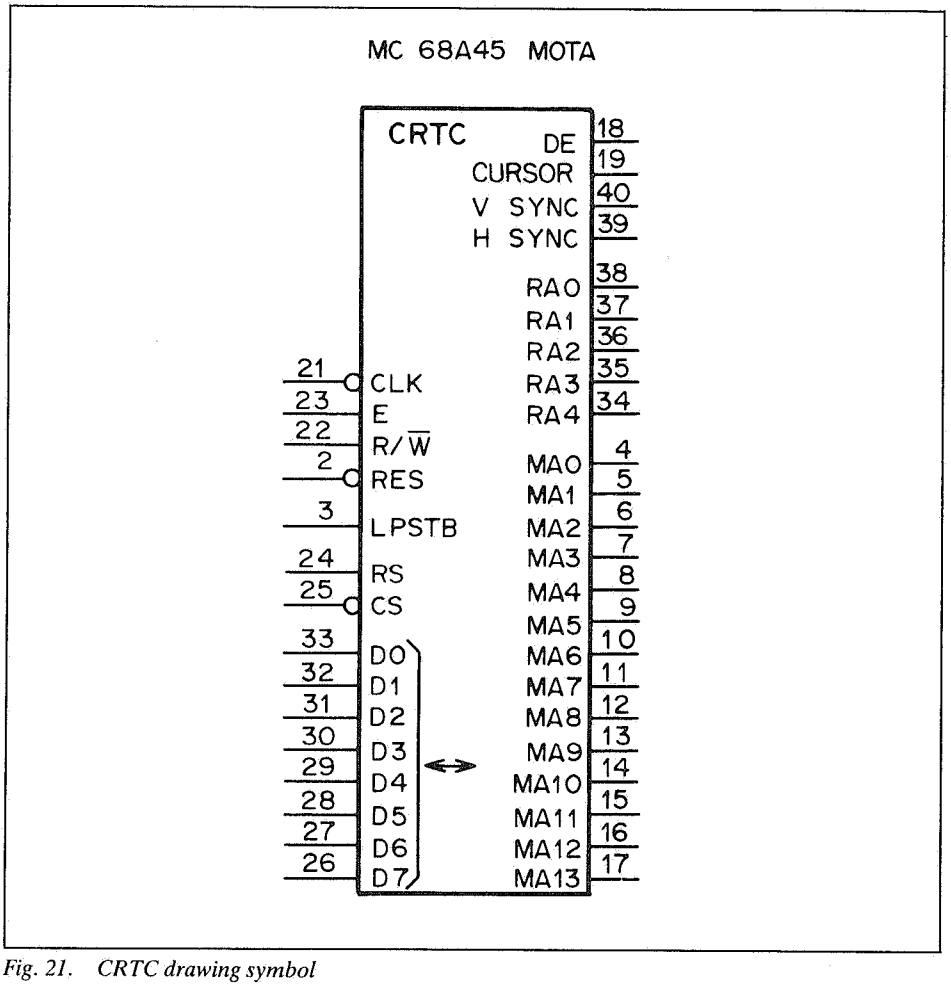


Fig. 21. CRTC drawing symbol

CRTC – MPU Interface and Reset

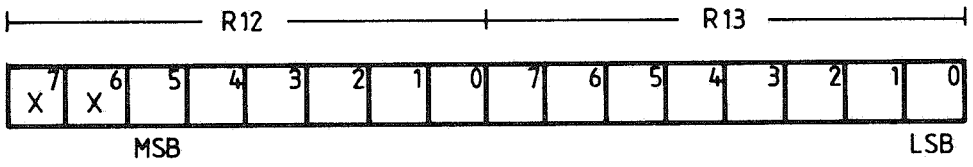
The CRTC interfaces the microcomputer via the following lines:

- Eight bidirectional data lines (D7 – D0) connected to the internal data bus.
- One read/write line (R/W, Low = Write) fed from the R/W output of the MPU.
- One enable line (E) which enables the inputs/outputs of the MPU interface and clocks data to and from the CRTC. It is driven by the Ø2 clock.
- One chip select line (CS) activated when the CRTCS signal from the address decoder is active (CS is thus activated by the internal addresses $F7D8_{(16)}$ – $F7DF_{(16)}$, see Memory Map).
- One register select line (RS) connected to address bit 0 (from the MPU). The CRTC contains one 5-bit write-only address register and 15 other registers of variable length that are accessible from the address bus. The address register is accessed when CS is active and RS = 0, i.e. at the addresses $F7D8_{(16)}$, $F7DA_{(16)}$, $F7DC_{(16)}$ or $F7DE_{(16)}$. One of the 15 registers is addressed when CS is active and RS = 1, i.e. at the addresses $F7D9_{(16)}$, $F7DB_{(16)}$, $F7DD_{(16)}$ or $F7DF_{(16)}$. The contents of the address register decide which one of the 15 registers is accessed, e.g. register 5 (R5) is accessed when the address register contents is $XXX00101_{(2)}$.

The CRTC can be reset by a signal from the display adapter peripheral interface adapter (DIA PIA) fed to the reset (RES) input of the CRTC. All outputs go to low level and all internal counters are cleared at reset but the internal registers are unaffected. The CRTC must be kept reset during or be reset after initial programming.

CRTC Registers

The above mentioned registers are summarized in the table below. The most significant bits are not used when a register contains less than eight bits. When a double register contains more than eight bits it is formatted as in the following example:



Register	Number of bits	Name	Function
R0	8	Character times per sweep	Determines the total number of horizontal character (displayed + non-displayed) position times per sweep minus one.
R1	8	Characters per line	Determines the number of characters to be displayed on a line.
R2	8	H SYNC position	Determines at which horizontal character position the H SYNC signal should start.
R3	4	H SYNC width	Determines the number of character times the H SYNC signal should be active.
R4	7	Lines total	Determines the total number of lines (displayed + non-displayed) minus one.
R5	5	Vertical total adjust	Determines how many sweep times should be added to the "Lines total" time to form a full frame time.
R6	7	Lines displayed	Determines how many lines should be displayed.
R7	7	V SYNC position	Determines after which line the V SYNC signal should be generated (V SYNC is active during the time of one line).
R8	2	Interlace control	Should be 00 or 10 as interlace is not used.
R9	5	Maximum raster address	Determines the number of sweeps per line minus one. (Must match with DIA PIA programming.)
R10	B+P+5	Cursor start and blink	The five bits determine at which sweep of a line the cursor top is located. The B and P bits, cursor blink and blink rate. See also Fig. 22.
R11	5	Cursor end	Determines at which sweep of a line the cursor base is located. See Fig. 22.
R12 R13	14	Display start address	Determines at which linear address in the display memory (character position in memory) the display should start.
R14 R15	14	Cursor location	Points out at which linear address (character position in memory) the cursor should be displayed.
R16 R17	14	Light pen hit address	Not used.

R0 – R13 are write-only registers, R14 – R15 are read/write registers and R16 – R17 are read-only registers.

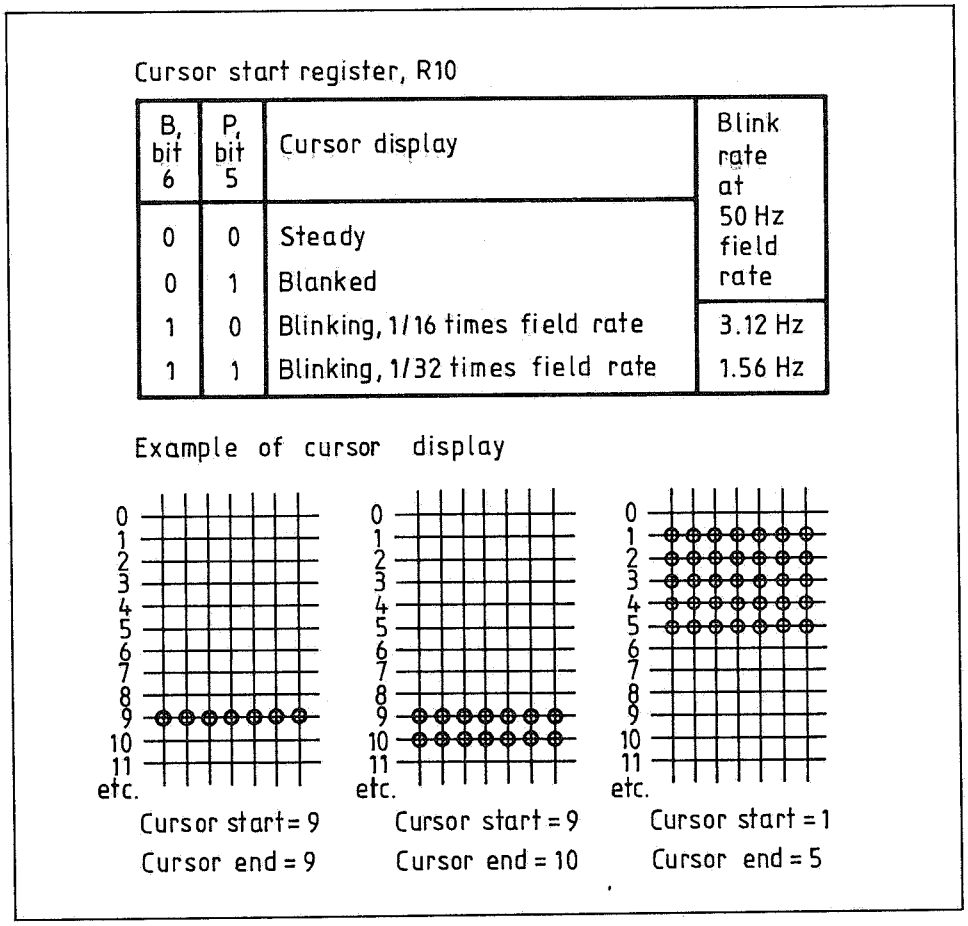


Fig. 22. Cursor control functions of R10 and R11

CRTC Control and Address Outputs

The timing of the CRTC control and address outputs, as well as the internal counters, is controlled by a clock signal (T1) fed to the clock (CLK) input of the CRTC. The clock provides two periods for each system clock cycle, i.e. one clock period for each horizontal character position. See Fig. 23.

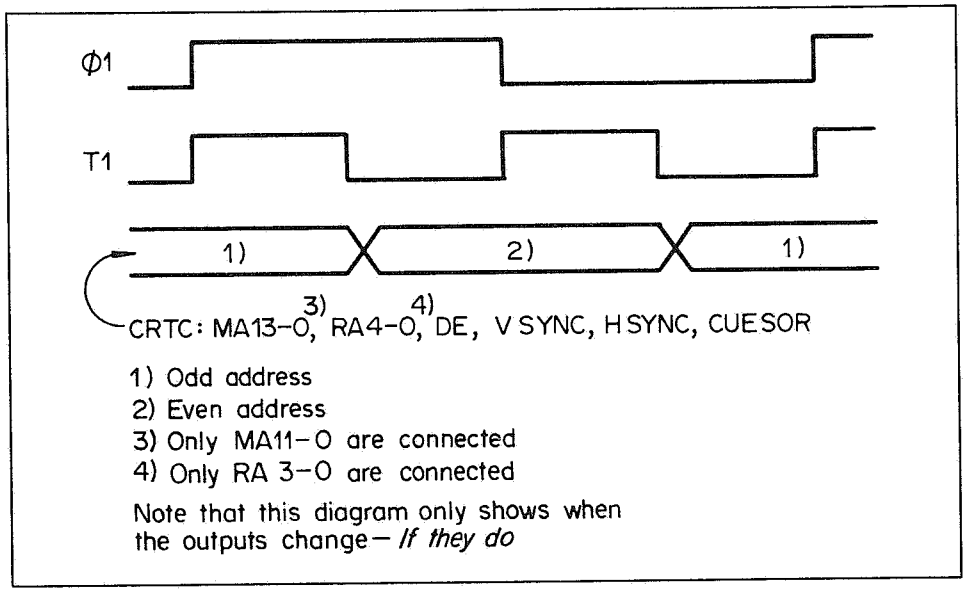


Fig. 23. CRTC timing

The following control outputs are provided:

- DE, device enable, is the master enable signal for the video generating circuitry. This signal is active during the display time of the lines, i.e. during the part of the frame time decided by R1 and R6 but not during horizontal and vertical retrace time or when the CRTC is reset.
- CURSOR, cursor display, is active during the time when cursor dots should be produced by the video generating circuits, i.e. at applicable sweeps, decided by R10 and R11, through a specific character position, decided by R14 and R15.
- H SYNC, horizontal synchronization signal, tells the logic that the cathode ray should retrace and start the next sweep over the screen. The start and duration of the H SYNC is programmed in R2 and R3.
- V SYNC, vertical synchronization signal, governs the vertical retrace of the ray that precedes the regeneration of an image on the screen. The start of the V SYNC is programmed in R7.

The V SYNC signal is buffered and then called VSB when fed to various parts of the logic. A latch is provided which synchronizes the DE, CURSOR, H SYNC and some other control signals with the rising edge of T5. See Fig. 24.

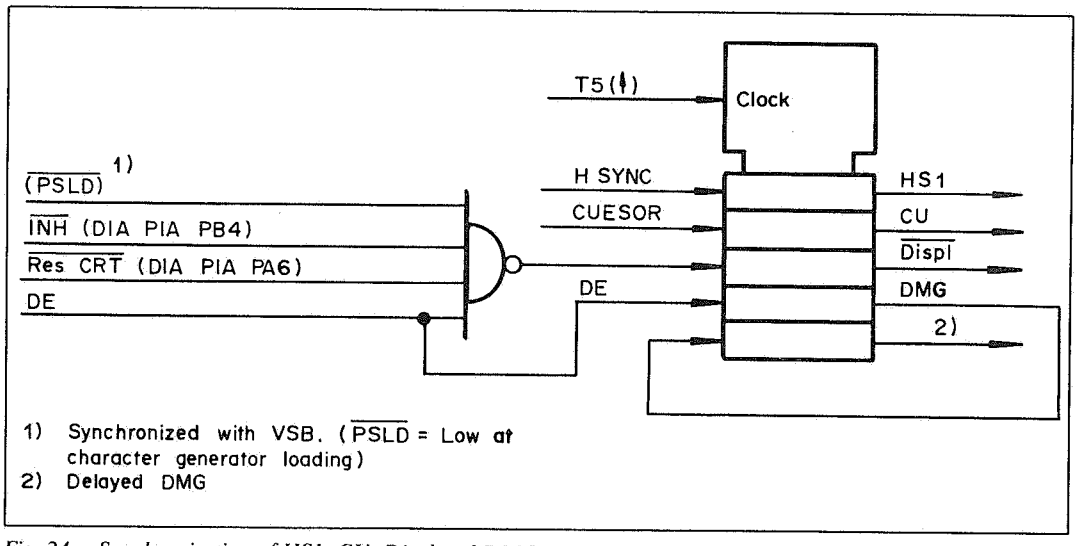


Fig. 24. Synchronization of HS1, CU, Displ and DMG

The CRTC provides a 5-bit raster address (RA4 – 0) out of which four bits are used. They are fed to the raster address modification ROM via the raster address counter, which is transparent during presentation, i.e. when the PSLD signal, used at character generator loading/reading, is inactive. They are then changed in the ROM to a modified raster address (PP3 – 0), which is fed to the character generators simultaneously with the code for the actual character. Thus, for each line the original raster address tells the actual sweep number, i.e. provides values from 0 up to the programmed value of R9 and the modified raster address tells in which locations of the character files the actual dot pattern is stored in the character generator. The original raster address (from the raster address counter outputs, RA3 – 0) is also used in the attribute generator (see Attribute Generator). The character generator functions are described under Character Generators.

The CRTC also provides a 14-bit linear address (MA13 – 0) out of which 12 (MA11 – 0) are fed to the display memory, i.e. the address of the actual character and character attribute or field and extended field attributes to be handled. This address is supplied by an internal linear address counter. This counter is started at a value decided by R12 and R13 and is stepped as many times as decided by R1 for each sweep. For each line it counts through the *same* address sequence as many times as decided by the value in R9. Then it goes on after the next H SYNC and makes a similar sequence for each displayed line i.e. as many times as decided by R6. This whole sequence starts over again after the next V SYNC, unless the programming of the CRTC has been changed.

Note that the display memory area as seen by the CRTC starts at 0 and ends when the position decided by R1 and R4 is reached. Thus it is possible to obtain a wrap-around function by starting the presentation at any suitable address, written in R12 and R13. When the CRTC has reached the highest address it restarts from 0 again. See Fig. 25. (Do not mix up this wrap-around with the wrap-around of attributes.)

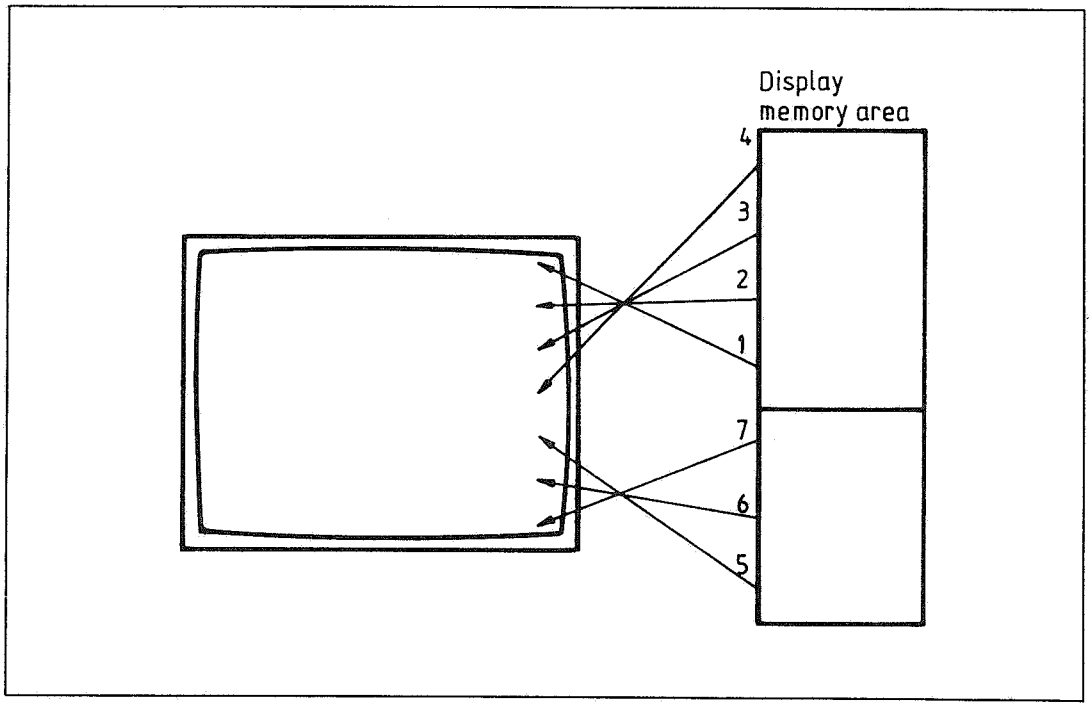


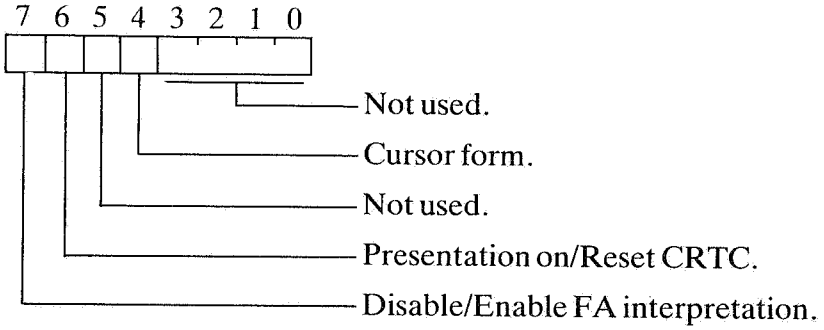
Fig. 25. Principles of screen wrap-around

Display Adapter Peripheral Interface Adapter, DIA PIA

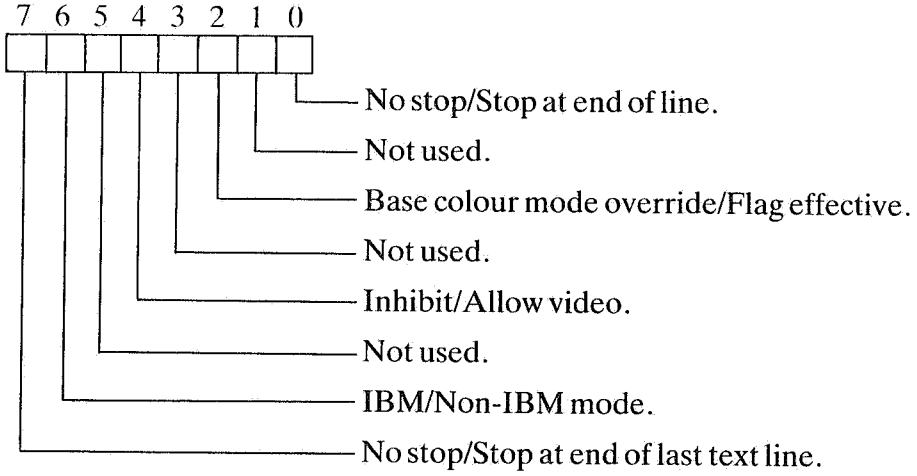
The display adapter peripheral interface adapter, DIA PIA is connected to the bus system of the microcomputer and controls several functions in the presentation logic. For definitions and a general survey, see the chapter on Microcomputer; Peripheral Interface Adapter, PIA. The MPU-interface consists of the 8-bit internal data bus, the R/W signal, address bus bits 3, 1 and 0 to the chip select and two register select inputs. Two other chip select inputs (CS1 – 0) are activated by the DIA PIA signal from the address decoding PAL. The timing is supplied by Ø2 on the E input. The DIA PIA is reset by the general reset signal.

The IRQ lines are not used and neither are the control lines on the peripheral side. The control register (CRA and CRB) contents shall be $04_{(16)}$ except when the DIA PIA is initiated. The peripheral register bits (PA7 – 0 and PB7 – 0) are programmed as outputs:

DIA PIA PA7 – PA0



DIA PIA PB7 – PB0



DIA PIA Peripheral Register A

The functions of the peripheral register A outputs are further described below:

PA7 0 = Disable FA interpretation. (Does not affect interpretation of MLFA, $FF_{(16)}$.)
1 = Enable FA interpretation.

PA6 0 = Enable CRTC.
1 = Reset CRTC. PA6 must be set during or set-reset after initial programming of the CRTC.

PA4 0 = Cuform 0 or covering cursor.
1 = Cuform 1 or transparent cursor.

DIA PIA Peripheral Register B

The functions of the peripheral register B outputs are further described below:

PB7 0 = No stop at end of last text line (message line excluded) if PB0 = 0. (Page wrap-around.)
1 = Stop at end of last text line, i.e. attributes do not wrap-around from the last text line (message line excluded) to the first line of the screen.

PB6 0 = IBM mode. (Character files contain 12 locations.)
1 = Non-IBM mode. (Character files contain 16 locations.)

PB4 0 = Inhibit video. The video signals to the cathode ray tube unit are blocked.
1 = Do not inhibit video.

PB2 0 = Base colour mode override, i.e. the Base colour mode flag (see Microcomputer PIA, MIC PIA; PB3) is ineffective and the colours are controlled by character attributes or extended field attributes.
1 = Base colour mode flag (MIC PIA PB3) is effective.

PB0 0 = No stop at end of line. A FA is valid until the next FA or to the stop at end of last text line (message line excluded) if PB7 = 1. (Line wrap-around.)
1 = Stop at end of line. A FA is valid until the next FA or until end of line.

Attribute Generator and On/Off Logics

FA Decoder

The character and field attribute codes (on CH7 – 0) from the display memory are decoded in an FA-decoder together with some other signals. See Fig. 13 and the table below. This decoder thus enables the logic to differentiate between character codes and field attribute codes.

Activating input condition												Activated output		Comments
					Feed-back signals		CH					Signal		
FCC[1]	PSLD[2]	DMG[3]	HS1[4]	T2	$\overline{SR11}$[5]	SV11[6]	7	6	1	$\overline{5\cdot4\cdot1\cdot0}$[7]	$\overline{3\cdot2}$[8]	Name	Active level	
High or or		High High			High	High	High High	Low High	Low	Low	Low	ASV11	Low	FA and not message line MLFA ($FF_{(16)}$) Video off
High		High		Low	High		High	Low	Low			AT11	Low	FA and not message line
		High					High	High		Low	Low	SR	Low	MLFA ($FF_{(16)}$)
								Low			Low	SV	High	Video off[9]
High or or	Low Low Low	High High	High	Low Low Low	High		High High	Low High	Low	Low	Low	AREGCK	Low	FA and not message line MLFA ($FF_{(16)}$) Horizontal sync.

[1] From DIA PIA PA7. High when FA interpretation is enabled. [2] High at character generator loading/reading.
[3] High at presentation. [4] Synchronized H SYNC. [5] Low at message line. [6] High at video off.
[7] I.e. all high gives low. [8] I.e. both high gives low. [9] From CH3 – 2

The output signals are further explained together with the logic they control. The SR and SV signals, however, are explained below.

The SR signal is generated by an MLFA (message line field attribute, $FF_{(16)}$) when presentation is on from the CRTC (DE has set DMG high). The signal is latched in an FA register (see below) and then called SR11. SR11 is used to tell the logic that a message line is in progress.

The SV signal is actually generated each time CH6 = 0 and CH3 – 2 = $11_{(2)}$ but is only latched in the FA register (see below) when the combination appears in a field attribute. (The FA register is also clocked by the MLFAs but then CH6 = 1.) The SV signal is called SV11 after the FA register and is then used to turn off the video and will thus inhibit presentation.

FA and EFA Registers

Two registers, the FA and EFA registers, are provided to store information from the field attributes and the extended field attributes. See Fig. 26. They are clocked by the AREGCK signal from the FA decoder. Note that the conditions for this signal actually state when the logic reacts on an attribute, i.e. when DIA PIA PA7 enables field attribute interpretation, no character generator loading is in progress, presentation is on from the CRTC (DE has set DMG high), a message line has not started and an FA or an MLFA is encountered in the code (CH7 – 0) stream from the display memory. The AREGCK signal is also activated at each horizontal synchronization signal when no character generator loading is going on.

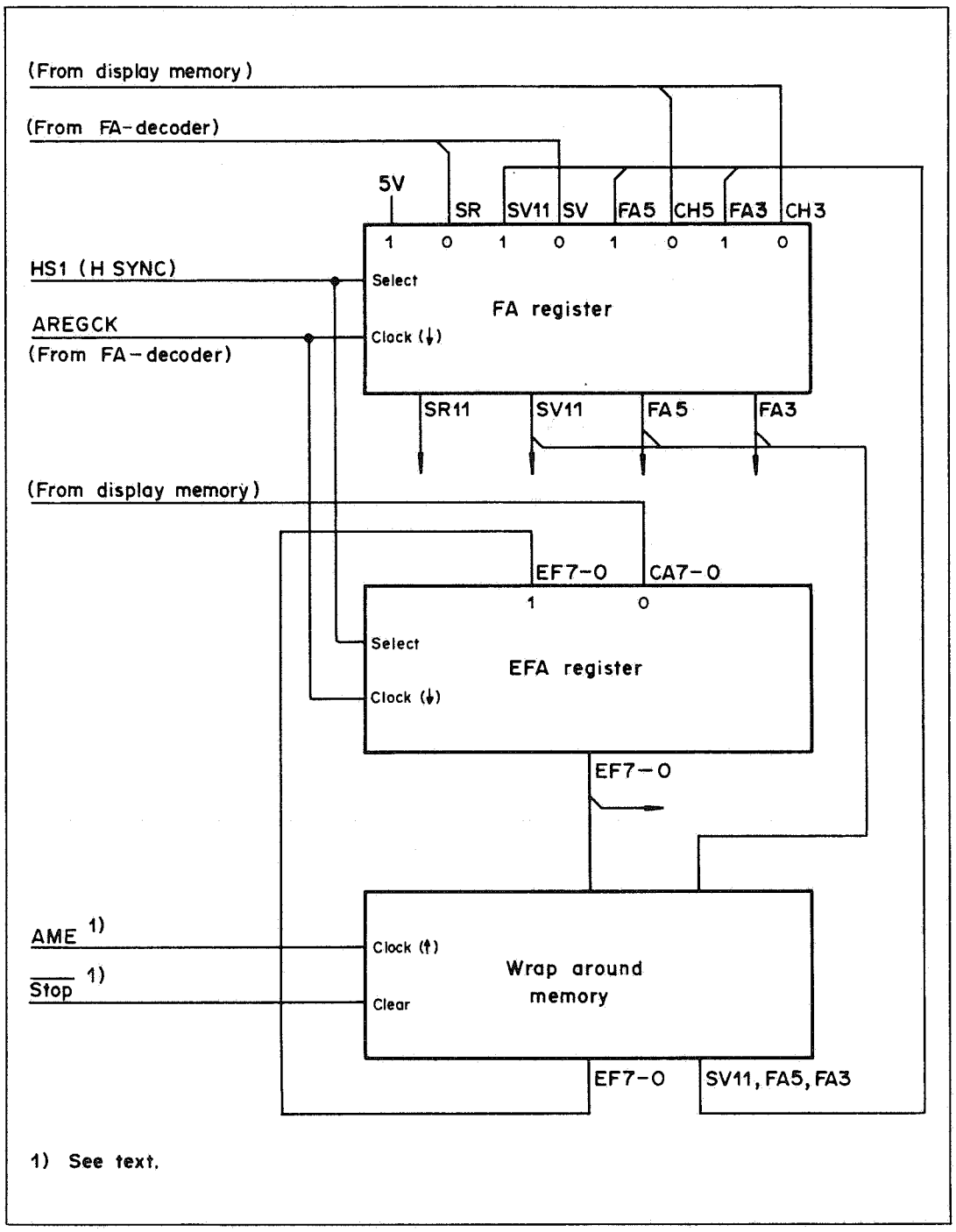


Fig. 26. FA and EFA registers with wrap-around memory

The inputs to the FA and EFA registers can be selected from one of two sources depending on the horizontal synchronization signal. At presentation the selected inputs to the FA register are fed from the FA decoder (SR and SV) and the display memory (CH5 and CH3). The source of the EFA register information is then the CA7 – 0 lines from the display memory. When the horizontal synchronization signal is active the registers are fed from a wrap-around memory (see below).

The outputs of the registers, corresponding to the CH5, CH3 and CA7 – 0 inputs, are called FA5, FA3 and EFA7 – 0 as the bits now certainly come from field attributes and extended field attributes.

Wrap-Around

The contents of the FA (except the SR11 bit) and EFA registers can be stored in a wrap-around memory. See Fig. 26. The memory is loaded (clocked by an AME signal, see under Highlighting Decoder) when sweep $B_{(16)}$ has passed the last position of a text line in IBM mode or when sweep $F_{(16)}$ has done the same in non-IBM mode. This will, however, not occur after a message line or when the memory is reset by a Stop signal.

The wrap-around memory will thus contain valid attribute information from the just-presented text line (not message line) or be reset, i.e. contain zeroes. This information is reloaded in the FA and EFA registers before the next line is started which will cause a "wrap-around" function of the attributes if the Stop signal is kept inactive.

The Stop signal is always activated when DIA PIA PB0 is set. That is, zeroes will be transferred from the wrap-around memory to the FA and EFA registers before each new line is started. The Stop signal is also activated coincidently with the vertical synchronization signal if DIA PIA PB7 is set and, as the horizontal synchronization signal runs during vertical retrace time, zeroes will be transferred from the wrap-around memory before the first line of the screen is presented. The AT11 signal from the FA decoder ensures that no page wrap-around occurs if no field attribute (MLFA excluded) existed on the just-presented page.

CA and EFA Decoding

The remaining circuitry of the attribute generator consists mainly of decoders of PAL type. Three are essentially used to decode the CA *or* EFA bits as follows. The highlighting decoder interprets bits 7 and 6, i.e. the extended highlighting bits. It also generates raster address dependent signals. The colour ATB decoder interprets bits 5 through 3, i.e. the colour bits. If these are all zeroes it also considers the FA bits 5 and 3. The character generator (Chargen) selector interprets bits 2 through 0, i.e. the symbol store bits. It also generates three other control signals.

Most of the outputs from the above mentioned decoders are then further decoded in a colour decoder which outputs are synchronized in a latch before they are fed to the on/off logics discussed later.

The different decoders are described in detail below.

Highlighting Decoder

The highlighting decoder functions are summarized below:

Activating input conditions												Activated output		Comments
				EF		CA		RA[4]				Signal		
$\overline{IBMM}$[1]	DMG	DMG11[2]	$\overline{SR11}$[3]	7	6	7	6	3	2	1	0	Name	Active level	
Low or High	Low Low	High High	High High					High High	Low High	High High	High High	AME	High	Clocks wrap-around memory at end of line
Low								High	High			DSW	High	Sweeps $C_{(16)} - F_{(16)}$ in IBM mode
			Low					Low	Low	Low	Low	SW0	High	Sweep 0 of message line
or				High	Low	High Low	Low Low					Inv F	High	Invert
or				Low	High	Low Low	High Low					Blink	High	Flash
Low or Low or High or High				High	High	High Low High	High Low High	High High High	Low Low High	High High High	High High High	USW	High	Underline

[1] From DIA PIA PB6. Low at IBM mode.

[2] Delayed DMG: DMG = Low and DMG11 = High will thus occur whenever display enable (DE) from the CRTC goes inactive.

[3] From the FA register. Low at message line presentation.

[4] Raster address originating from the CRTC. (The addresses generated by the raster address counter at character generator loading/reading are irrelevant as the presentation then is turned off.)

The function of the AME signal was discussed under Wrap-Around. The DSW and SW0 signals inform the colour decoder about dead sweeps in IBM mode and about the first sweep of a message line.

The InvF signal states when a position or a field shall be inverted due to the fact that the CA bits 7 and 6 are $10_{(2)}$ or EF7 – 6 are $10_{(2)}$ when CA7 – 6 are $00_{(2)}$.

The Blink and USW signals are generated in an analogous way when the mentioned bits are $01_{(2)}$ and $11_{(2)}$ respectively. The Blink signal states when a position or field shall be flashing and the USW signal when it shall be underlined. Note, however, that the USW signal only is active during raster address $B_{(16)}$ in the IBM mode and $F_{(16)}$ in the non-IBM mode, i.e. during the sweep generating the actual underline.

Character Generator Selector

The character generator selector functions are summarized below:

Activating input conditions												Activated output		Comments
						EF			CA			Signal		
PSLD[1]	$\overline{ASV11}$[2]	$\overline{CU \cdot CUF}$[3]	BT[4]	Inv F[5]	Blink[5]	2	1	0	2	1	0	Name	Active level	
or Low						High			High Low	Low	Low	G2	High	Store number as in extended field **or** character attribute
or Low							High		Low	High Low	Low	G1	High	
or Low								High	Low	Low	High Low	G0	High	
or	High High	Low Low	Low	Low Low	Low							B2	High	Intermediate control signals
High or or or	Low	High High High	High	High	High							B1	High	
Low or Low	High High		Low		Low							S1	High	

[1] High at character generator loading/reading.

[2] From the FA decoder.

[3] Low for transparent cursor (DIA PIA PA4 = 1) and cursor address (from the CRTC).

[4] Always running clock from a blink timer.

[5] From the highlighting decoder.

Note that the store number (G2 – 0) can only originate from the CA bits at character generator loading/reading (PSDL = High).

The B2, B1 and S1 signals are intermediate control signals, used in the colour decoder described later. The B2 signal is high at a transparent cursor address if it is not in an FA, MLFA or inverted position or in a not displayed field (FA3 – 2 = $11_{(2)}$). If flashing is ordered, the B2 signal is only high at the beforementioned conditions when a blink timer generates a low output.

The B1 signal is high:

- At character generator loading/reading.
- At blink timer high time when flashing is ordered; but not in a transparent cursor position.
- At an FA or MLFA position or in a not displayed field (FA3 – 2 = $11_{(2)}$) except when it is in a transparent cursor position.
- In inverted positions but not if it is in a transparent cursor position.

The S1 signal is never high at character generator loading/reading, FA or MLFA positions or in a not displayed field (FA3 – 2 = $11_{(2)}$) but otherwise activated if flashing is off or, if on, at blink timer low time.

Colour ATB Decoder

The colour ATB decoder functions are summarized below:

Activating input conditions													Activated output		Comments
			FA		EF			CA			Feed-back		Signal		
SV11[1]	$\overline{BAS}$[2]	TP[3]	5	3	5	4	3	5	4	3	$\overline{BA}$	$\overline{RA}$	Name	Active level	
Low or Low		High / High			High	High	High	High / Low	High / Low	High / Low			NEUTRAL	Low	Up to seven colours in the same position
Low or Low or Low or Low	Low / High		High	High	Low	Low	High / Low	Low	Low	High / Low			BA	Low	Blue colour
Low or Low or Low				High	Low	High / Low	Low	Low	High / Low	Low			RA	Low	Red colour
or or or					High / Low	Low	Low	High / Low	Low	Low	Low / High	Low / High	GA	Low	Green colour

[1] From the FA register. Low at video on.

[2] Low for Base colour flag effective (DIA PIA PB2 = 1) and base colour mode (MIC PIA PA3 = 1).

[3] From the raster address modification ROM. High at triple plane store addressing.

The first two condition combinations for each signal are easy to understand if compared with what is said under Dynamic Attributes in the Brief Outline. The remaining condition combinations are relevant only when both the CA and the preceding EFA contains default values and will then function as said below:

- The BA signal will be activated if FA5 = 1 in base colour mode (blue only) or if FA3 = 1 in "monochrome" mode (blue component in white).
- The RA signal will be activated if FA3 = 1 in "monochrome" mode (red component in white).
- The GA signal will be activated when both the BA and RA signals are activated (green component in white) and when both BA and RA are deactivated (default green).

Colour Decoder

The colour decoder interprets the outputs from the three decoders described above to combined conditions for the on/off logics. The functions of the colour decoder are summarized below:

Activating input conditions												Activated output		Comments[4]
From highlighting decoder			From character generator selector			From colour ATB decoder						Signal		
DSW	SW0	USW	B2	B1	$\overline{S1}$	$\overline{BA}$	$\overline{RA}$	$\overline{GA}$	$\overline{Displ}$[1]	CU[2]	CUF[3]	Name	Active level	
Low or Low	Low								Low	High	Low			Cov. curs.
	High								Low			Binv	Low	Sweep 0
Low	Low		Low	Low		Low			Low					
Low or Low	Low		Low	Low			Low		Low	High	Low	Rinv	Low	Cov. curs.
Low or Low	Low		Low	Low				Low	Low	High	Low	Ginv	Low	Cov. curs.
Low or Low	Low	Low			High	Low			Low	Low		BON	Low	
	Low	Low			High	Low			Low		High			Tr. curs.
Low or Low	Low	Low			High		Low		Low	Low		RON	Low	
	Low	Low			High		Low		Low		High			Tr. curs.
Low or Low	Low	Low			High			Low	Low	Low		GON	Low	
	Low	Low			High			Low	Low		High			Tr. curs.

[1] Low at presentation on, i.e. when CRTC DE is high, DIA PIA PA6 = 0, DIA PIA PB4 = 1 and (VSB-synchronized) PSLD is low.

[2] High at cursor address.

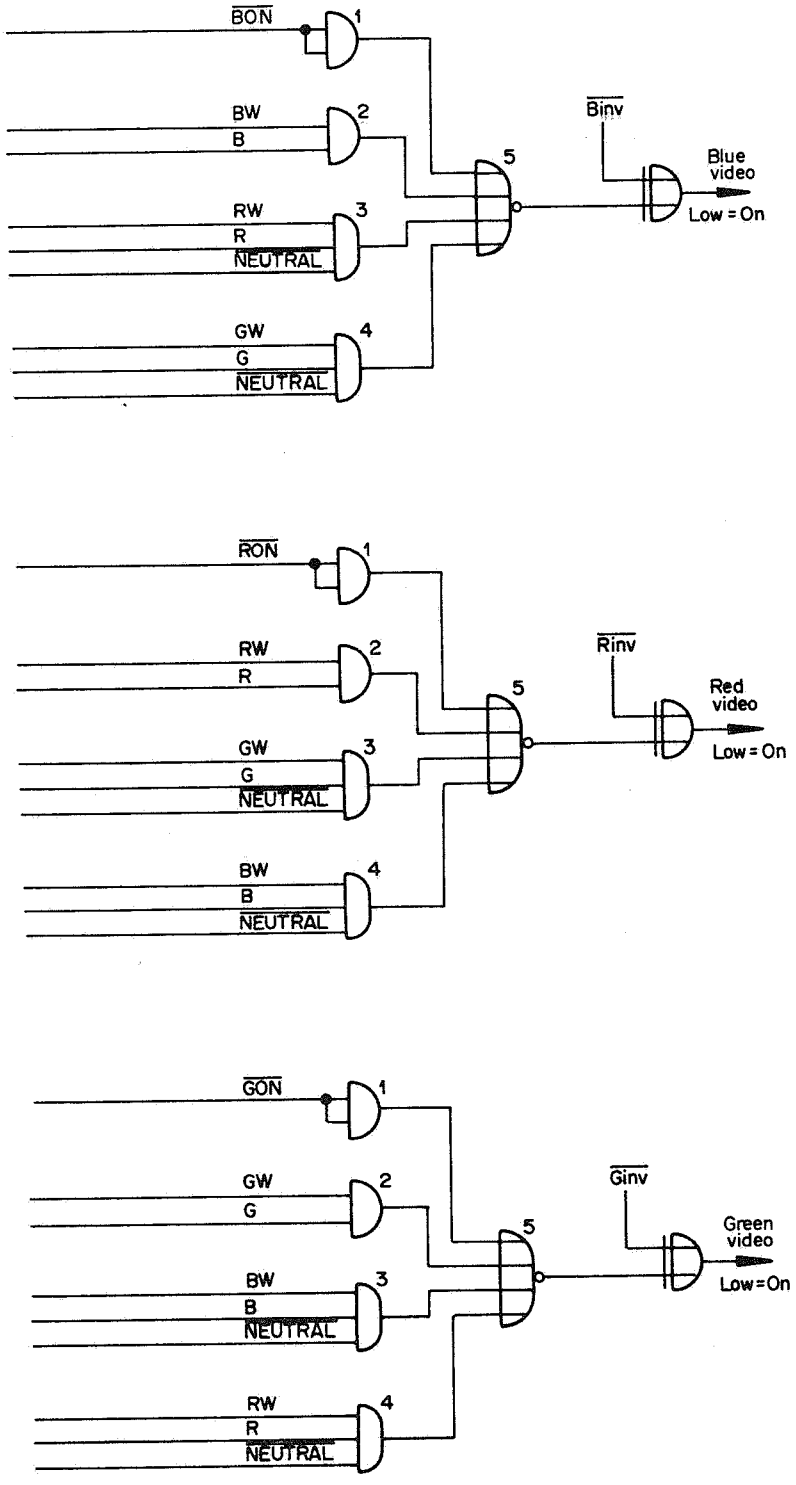
[3] High at transparent cursor, i.e. when DIA PIA PA4 = 1.

[4] The rows marked with Cov. curs. (Covering cursor), Tr. curs. (Transparent cursor) and Sweep 0 (of message line) apply in the implied cases. The rest of the rows are otherwise valid.

The functions of the outputs are described under On/Off Logics below. Trace the input conditions to their sources in order to find out the status of a certain output. An example is inserted after the section on the on/off logics.

On/Off Logics

The on/off logics are shown in Fig. 27. As the three groups function in similar ways the signals will, in this section, be called: Invert (Binv, Rinv, Ginv), On (BON, RON, GON), Video (BW, RW, GW) and Block (B, R, G) signals. The Block signals originate from the raster address modification ROM (then called BC, RC and GC and of the same polarity as B, R and G). BC, RC and GC can be found in Appendix 3.



Signal sources:

Binv, Rinv, Ginv: Colour decoder
BON, RON, GON: Colour decoder
NEUTRAL: Colour ATB decoder
BW, RW, GW: Character generator via video registers
B, R, G: Raster address modification ROM

Fig. 27. On/Off logics

The Invert signals will invert or not invert the output of gate 5 in Fig. 27. If an On signal is high the output of gate 5 is independent of the inputs from gate 2 – 4. The effect of On and Invert is summarized below:

On	Invert	Comment
Low	Low	Normal video
Low	High	Inverted video
High	Low	Forced video on
High	High	Forced video off

The normal and inverted video cases can, however, be handled in different ways depending on the NEUTRAL signal. If this signal is low gate 3 and 4 are disabled and each colour channel will, via gate 2, be fed from its corresponding character generator block (blue block controls blue colour etc.). This alternative is used at presentation from triple plane stores. Note that all three Block signals are then high.

If the NEUTRAL signal is high, gate 3 and 4 are enabled in addition to gate 2. This will feed the same video information to all three of the colour channels independent of from which block the information comes. A high Block signal denotes the working block. Note, however, that the channels can be turned on or off by the On and Invert signals and that these signals are affected by the colour bits in the attributes. If e.g. a yellow character is to be presented on a not inverted screen RON, Rinv, GON and Ginv must be low to turn on the red and green channels, while BON and Binv must be high to turn off the blue channel. The NEUTRAL signal is high at presentation from single plane stores.

Example on Attribute Generator Functions

This example is included in order to make it possible to check a trace of conditions through the attribute generator and the on/off logics.

Consider the presentation of an ordinary blue letter in an ordinary position on a display unit in IBM mode and with the full version of character generators. Say that character generator 0 is working, sweep 3 is just presented and:

Bit: 7654 3210 7654 3210 7654 3210
FA = 1000 0000 EFA = 0000 1000 CA = 0000 0000

FCC = High PSLD = Low DMG = High HS1 = Low
CH7, 6, 1 = not High, Low, Low or High, High, High
TP = Low $\overline{\text{Displ}}$ = Low CU = Low CUF = High

FA decoder result:

$\overline{\text{SR11}}$ = High (inact.)
SV11 = Low (inact.)
$\overline{\text{ASV11}}$ = High (inact.)
$\overline{\text{AT11}}$ = High (inact.)
$\overline{\text{AREGCK}}$ = High (inact.)

Highlighting decoder result:

AME = Low (inact.)
DSW = Low (inact.)
SW0 = Low (inact.)
InvF = Low (inact.)
Blink = Low (inact.)
USW = Low (inact.)

Character generator sel. result:

G2 – 0 = Low (inact.)
B2 = Low (inact.)
B1 = Low (inact.)
S1 = High (act.)

Colour ATB decoder result:

$\overline{\text{NEUTRAL}}$ = High (inact.)
$\overline{\text{BA}}$ = Low (act.)
$\overline{\text{RA}}$ = High (inact.)
$\overline{\text{GA}}$ = High (inact.)

Colour decoder result:

$\overline{\text{Binv}}$ = Low (act.)
$\overline{\text{Rinv}}$ = High (inact.)
$\overline{\text{Ginv}}$ = High (inact.)
$\overline{\text{BON}}$ = Low (act.)
$\overline{\text{RON}}$ = High (inact.)
$\overline{\text{GON}}$ = High (inact.)

The red and green colour channels of the on/off logics are forced off as Rinv, Ginv, RON and GON are all high. The blue channel is enabled for normal video as Binv and BON are low. RC and GC from the raster address modification ROM are low, i.e. R and G are low which leaves only gate 2 open through the on/off logics as BC and B are high. The blue video (BW) will thus control the blue gun of the CRT.

Cathode Ray Tube Unit, CRU

Please refer to circuit diagram E34112 2030 (TX-1404EFA). The diagram shows the circuitry, the cathode ray tube and the interconnections.

Deflection Circuits

The horizontal deflection is governed by an integrated circuit (IC 401), which contains an automatic frequency control circuit (AFC) a horizontal oscillator and an X-ray protection circuit. The H SYNC signal is fed to pin 17 and a feed-back pulse from a flyback transformer (T551) to pin 15. The horizontal drive signal is output from pin 12 unless inhibited by the X-ray protection circuit, which is controlled via a shut down circuit by the heater voltage (from T551). The horizontal drive signal is amplified by a transistor (Q502) and a transformer (T501) in order to generate a strong drive current for the switching horizontal output transistor (Q551), which drives the primary winding of the flyback transformer and the horizontal deflection coil. The accelerating voltage of the anode (EHT) is stabilized by a circuit that affects the characteristics of the primary winding (of T551) and its serial condensor (C564) circuit. The secondary windings of the flyback transformer supplies the internal voltages to the CRU.

The V SYNC signal is inverted (in Q401) and fed to (pin 8 of) the vertical oscillator in the integrated circuit (IC 401). This circuit governs the vertical deflection and the output (pin 6) is amplified (in Q402, Q403, Q404, Q406, Q407, Q408) and fed to the vertical deflection coil.

The vertical deflection voltage is tapped for a pincushion correction signal, which via a transistor (Q751) and a saturable reactor (T751) affects the horizontal deflection.

Video Amplifiers and Brightness Control

The video circuitry has three identical circuits for red, green and blue video. Each one of the three video signals is received by a nand gate (IC 1301) and is fed further to an open-collector nand gate (IC 1302), which output is fed with a variable voltage via a transistor (Q1311). The voltage level is a function of the external Brightness potentiometer position and a colour mixing circuitry (see below). The signal is then inverted (by Q301/Q302/Q303) and amplified (by Q304/Q305/Q306 and Q308, Q311/Q309, Q312/Q310, Q313) before it is fed to the cathode of the tube. Furthermore there is a clamping circuit (around Q307) for adjustments of the cut-off levels of the three different cathodes.

The colour mixing circuitry (IC 1303, IC 1304 and surrounding components) is included for ergonomic reasons. It provides less saturated primary colours as a compensation for the fixed colour coordinates of the phosphors and the effect of the front glass. E.g. an addition of weak blue and green video signals to a normally powered red video signal generates a less saturated red (nearer to white) impression of the colour on the screen than without the addition.

Adjustments

There are six magnets on the neck of the tube and a number of trimming potentiometers for adjustments. These are described in the Installation and Maintenance Manual under Service Manuals Colour Cathode Ray Tube Units.

Keyboard Asynchronous Communication Adapter, KB ACIA

ACIA Functions

For definitions and a general survey, see Asynchronous Communication Adapter, ACIA under Internal Communication with Printers in the Communication chapter.

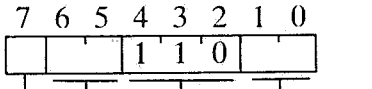
The MPU interfaces the ACIA by the 8-bit internal data bus, the Read/Write line, the address bus and one interrupt line. Address bit 0 is directly fed to the register select input of the ACIA and the inversion of bit 2 to one chip select input. The two other chip select inputs of the ACIA are supplied by the address decoder. The timing on the enable input is supplied by Ø2. The IRQ line is fed to the interrupt register as interrupt 2.

Both the transmitter and receiver clocks are derived from Ø2 in a divide-by-14 counter and the clock rate is thus 76.07 kHz. (The RTS output is not used nor are the CTS and DCD inputs which are hard-wired low.)

The received and transmitted data lines are connected to each other on the outside of the line buffers, thus creating a bidirectional line. (The ACIA can thus receive the same data as it transmits.)

ACIA Programming

The ACIA control register is usually programmed in the following way:



$10_{(2)}$ during operation. The external clock (76.07 kHz) is divided by 64 and the bit transfer rate is thus 1188 Hz.
$11_{(2)}$ during reset periods (Master reset).

The word length is eight bits of data, one even parity bit and one stop bit.

$01_{(2)}$ during transmitting. Interrupts at transmitted data register empty are enabled.
$X0_{(2)}$ during reception from the keyboard or when not communicating with the keyboard.

0 when not communicating with the keyboard. Receiver interrupts are disabled.
1 during communication with the keyboard. Interrupts at received data register full and overrun are enabled. (The receiver interrupt may be used to check when the last bit has been shifted out at transmission.)

Two-wire Interface

The two-wire interface can work in MPU mode or DMA (direct memory access) mode. The MPU mode is used when the direct memory access controller, DMAC, and the advanced data link controller, ADLC, circuits are initialized or supervised by the MPU. DMA mode is used at DMA transfers.

The main circuits and signals of the two-wire interface are shown in Fig. 28. Please refer to the figure when reading the following text.

Direct Memory Access Controller, DMAC

The DMAC handles DMA transfers between the read/write memory and the ADLC (see below). For definitions and a general survey of the DMAC, see the Microcomputer chapter.

The DMAC is in MPU mode addressed by the DMACS signal from the address decoder and five address bits. The R/W line is also fed to the DMAC. The bus buffers will allow the five address bits and R/W to pass in direction to the DMAC in the MPU mode.

At a service request from the ADLC (RDSR or TDSR active) the DMAC will make a DMA request and when this is accepted (DMA grant active) go to the DMA mode (with TXSTB active at DMA cycle 2). The DMAC then provides a memory address from an internal address counter, a

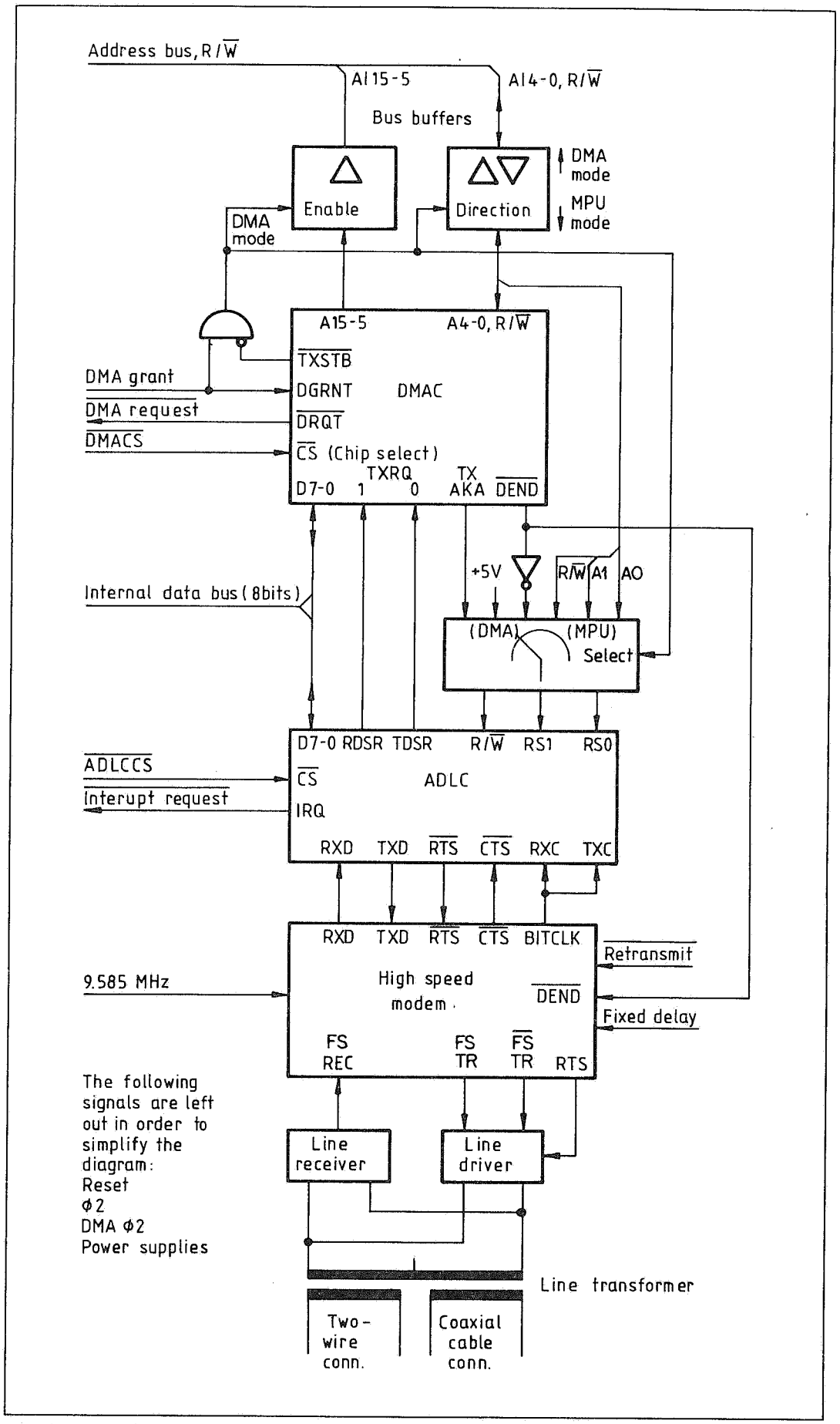


Fig. 28. Principles of two-wire interface

R/W signal (to the ordinary R/W line) and control signals to the ADLC (TXAKA, producing a read/write signal to the ADLC and DEND, active when the last byte of a block is transferred). The bus buffers will allow all fifteen address bits and R/W to pass in direction from the DMAC in the DMA mode.

The DMAC is connected for TSC steal mode and two channel mode. Channel 0 is used for transmitting and channel 1 for receiving.

Advanced Data Link Controller, ADLC

The ADLC converts parallel data from the internal data bus to serial data (TXD) for the high speed modem or serial data from the high speed modem (RXD) to parallel data for the data bus. For definitions and a general survey of the ADLC, see the Communication chapter.

The ADLC is programmed to generate a Request to send (RTS) signal to the high speed modem and to interrupt the MPU. The two-wire interrupt thus originate from the ADLC – not the DMAC. (The data carrier detect input, DCD, is hard-wired active. The DTR/LOC and FLAGDET outputs are not used.)

It is possible to make transfers of two-wire data via the MPU but at present only DMA transfers are used.

The ADLC is in MPU mode addressed by the ADLCCS signal from the address decoder and two address bits (AI1 – 0). The R/W line is also fed to the ADLC. The ADLCCS signal from the address decoder is also activated at DMA cycle 2 but the R/W and RS0 inputs are in DMA mode fed from the DMAC and the RS1 is then fed from a hard-wired high. The addressing is summarized below:

Address conditions							Register addressed and read/write function
MPU mode			DMA mode			Address control bit[1]	
R/W (R/W)	A1 (RS1)	A0 (RS0)	TXAKA (R/W)	+5 V (RS1)	DEND (RS0)		
0	0	0		[2]		X	Write control register 1
0	0	1		[2]		0	Write control register 2
0	0	1		[2]		1	Write control register 3
0	1	0	0	1	0	X	Write transmitter FIFO[3]
0	1	1	0	1	1	0	Write transmitter FIFO[4]
0	1	1	0	1	1	1	Write control register 4
1	0	0		[2]		X	Read status register 1
1	0	1		[2]		X	Read status register 2
1	1	X	1	1	X	X	Read receiver FIFO

[1] Control register 1 bit 0. [2] 0 is not possible. [3] Frame continue. [4] Frame terminate.

High Speed Modem

The high speed digital modem consists of one chip, which contains a modulator, a demodulator, a frequency divide and phase correction logic, and a delay circuit for Clear to send. See Fig. 29.

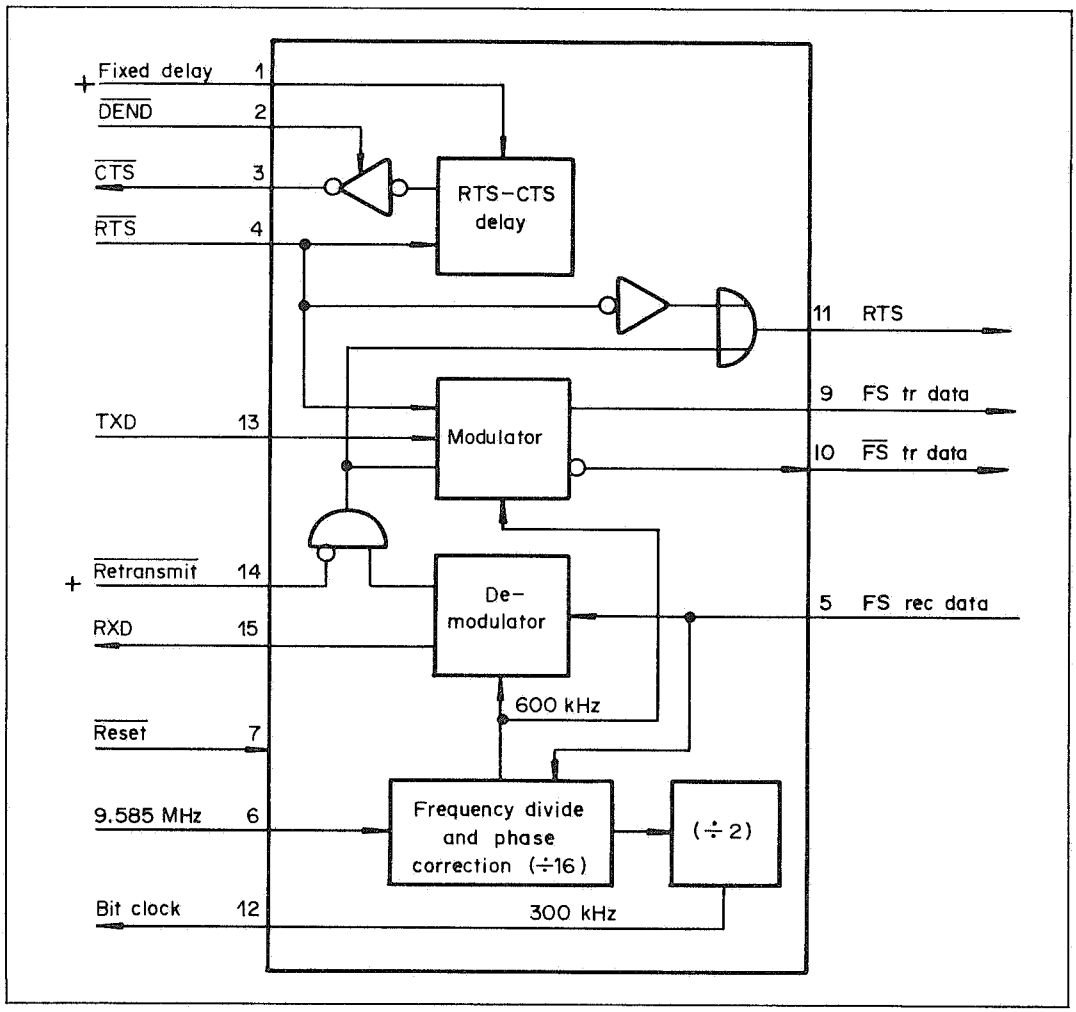


Fig. 29. High speed modem circuit

Data to be transmitted (TXD, from the ADLC) is presented in serial format to the modulator. The modulator converts the signal to a frequency shifted signal (FS tr data) which then is fed to the two-wire if Request to send (RTS) is active.

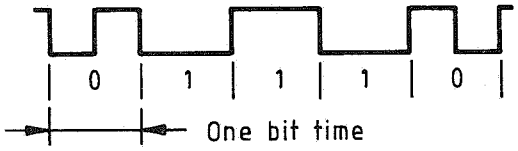
The frequency-shifted data transferred from the two-wire (FS rec data) is fed to the demodulator and to the frequency divide and phase correction logic. An internal strobe from the latter logic shifts the frequency shifted received data into the demodulator and generates a Bit clock fed to the ADLC. The demodulator converts the received data to demodulated valid data (RXD, fed to the ADLC). (Received data is not, as Retransmit is hard-wired high, fed to the modulator and retransmitted.)

The modulation method, clock generation and phase correction are described in the Communication chapter.

The Request to send output (RTS) follows (but inverted) the Request to send input (RTS). (It is also a function of received data in the not used retransmit mode.)

Clear to send (CTS) follows the Request to send input (RTS) with a delay of 32 (controlled by strapping of the fixed delay input) bit clock periods. This delay will enable the modem in the receiving unit (e.g. in a CPR) to synchronize before the transmission is started. A frequency corresponding to the state of the transmitted data input (TXD) will be sent on the FS tr data lines during the delay. An active DEND signal (at the end of a transmission sequence) from the DMAC will deactivate Clear to send.

Frequency-shifted received data is validated, i.e. each bit must consist of an LH, HL, LL or HH and there must be a shift of polarity between the bits in order to be accepted:



Received data (RXD) is always high when received data is invalid or before the modem has synchronized.

Line Interface

The high speed modem is connected to a line transformer via a line driver and a line receiver. The line transformer has separate windings for a two-wire and a coaxial cable connection. (See Fig. 28.)

Transmitting

To initiate a block transmission, using DMA, the program must prepare itself to answer an interrupt from the ADLC (interrupt 4) when the whole block (frame) is transmitted. The DMAC channel 0 address register, byte count register and control register should also be programmed with:

- Start address of DMA, i.e. the memory address of the first byte (except flag) to be transmitted on the two-wire connection.
- Number of memory bytes to be transmitted.
- Order to count up or down when addressing the memory.
- Order to produce the read state on the R/W line when addressing the memory.
- TSC steal mode.
- Set Enable TXRQ 0 in the DMAC priority control register in order to make the DMAC react to TDSR (Transmit data service request) from the ADLC.

The ADLC must be programmed for:

- Correct word length (8 bits).
- NRZ data format.
- Non-loop mode.
- Transmitter DMA mode etc.

The transmitting section of the ADLC is released and the Request to send (RTS) control bit is set by the program. After a delay Clear to send (CTS) is signalled back to the ADLC, and thus the Transmitter data service request (TDSR) output from the ADLC is no longer inhibited.

The following will happen as an answer to a TDSR signal:

- A DMA request signal (DRQT) will be sent to the timing logic.
- This logic will answer with a DMA grant signal (DGRNT), defining the DMA cycles 1 and 2.
- The DMAC will then produce a low level on the transfer acknowledge A (TXAKA) output as channel 0 is used. This will ensure a write strobe for the ADLC when;
- The DMA mode signal is activated by the Transfer strobe (TXSTB).

At the same time the ADLC is enabled, its transmit data register is addressed and a memory cell is addressed by the DMAC as the drivers for the address lines from the DMAC are enabled.

After a direct memory access the byte is shifted to the high speed modem. Thus it takes some time before the transmit data register is again available. The microcomputer is consequently undisturbed by any DMA request during this time.

When the DMA transfers (consisting of the number of bytes defined in the DMAC byte count register) are being completed, i.e. when the last byte is being fetched from the memory, the DEND output of the DMAC is activated. This will result in:

- A termination of the frame (write into the "transmit terminate FIFO" register; the ADLC adds CRCC and flag).
- A deactivation of the Clear to send (CTS) signal. The ADLC will react by not signalling any more TDSR.

  The non-CTS signal causes an interrupt. As an answer, the program resets the RTS signal, clears status bits in the ADLC and prepares the DMAC and the ADLC for reception.

Receiving

Preparatory programming of the DMAC and the ADLC for receiving operations resembles the one described in the Transmitting paragraph. Furthermore, the program ensures that an interrupt will be answered by a program check of receiver status etc.

When bytes following a leading flag are received, the ADLC sends receiver data service request (RDSR) to the DMAC. These bytes will then be stored in the memory according to DMAC channel 1 programming. The DMA transfer is terminated as a response to an interrupt caused by error (framing error, abort received or receiver overrun) or by frame valid status. The program then resets the ADLC and takes care of the received data on interrupt 0 level. Note that Request to send must be reset during reception.

Appendix 1

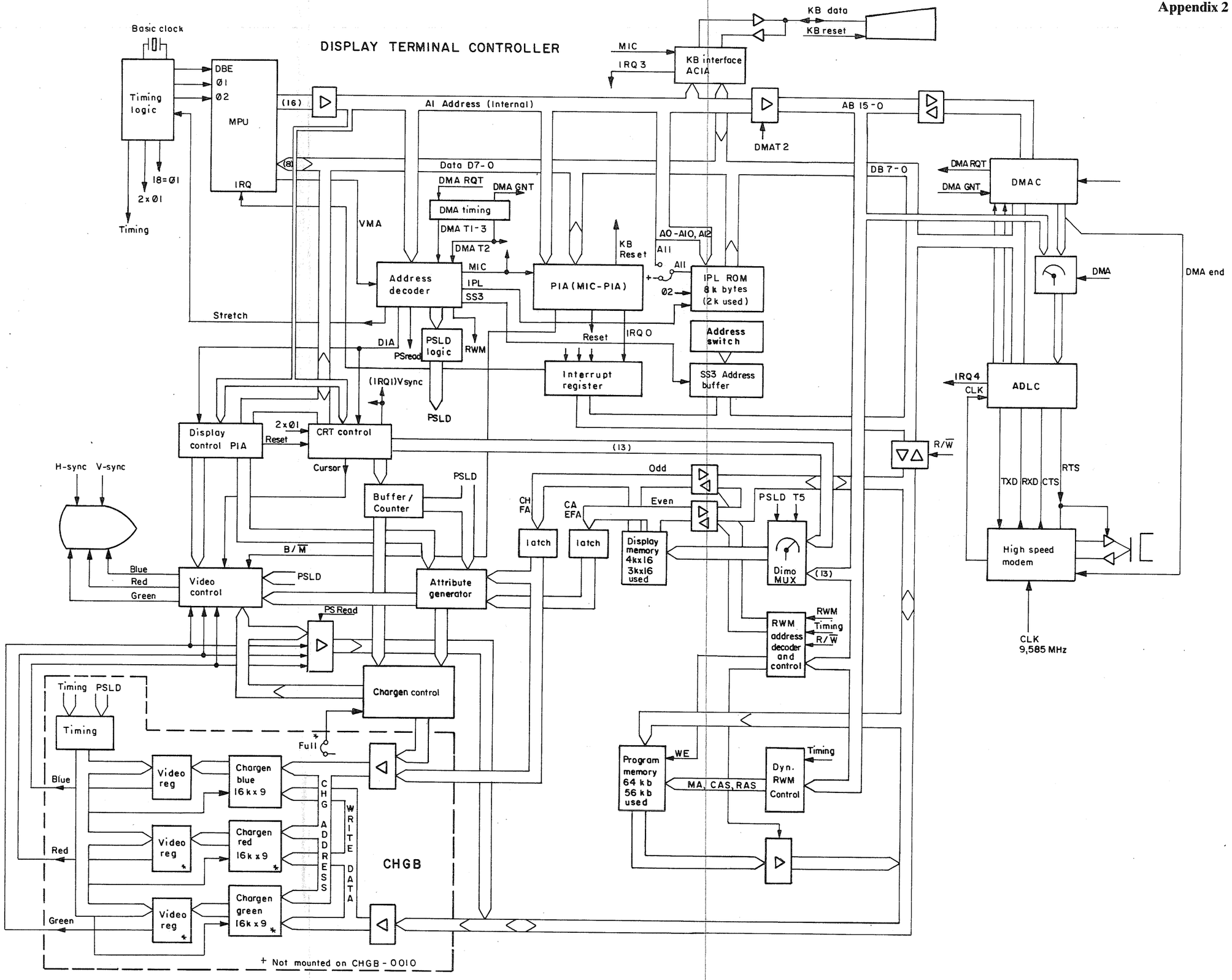
I/O Addresses, $F7FF_{(16)} - F700_{(16)}$

(The first two figures, $F7_{(16)}$, are omitted in the address column below. R/W = W means write, R/W = R means read and R/W = R/W means read or write as seen by the MPU.)

Address (Hex)	R/W	Other condition	Addressed unit and register
			MCP
FF	W		STORE (bit 7), CCR, ACCB, ACCA, XHI, XLO, PCHI and PCLO switches
FE	W		Data 7–0 switches
FD	W		Address 15–8 switches
FC	W		Address 7–0 switches
FB	R		A, B, C, D, NMI, IRQ, VMA and R/W LEDs
FA	R		Data 7–0 LEDs
F9	R		Address 15–8 LEDs
F8	R		Address 7–0 LEDs
F7–F0	R		**Interrupt register**
EF–E4			**Not used**
			Character generators
E3	R/W	See text	PSADR1
E2	R/W	See text	PSADR0
E1	R/W	See text	PSLDOFF
E0	R/W	See text	PSLDON
			CRTC
DF	→	→	= D9
DE	W		= D8
DD	→	→	= D9
DC	W		= D8
DB	→	→	= D9
DA	W		= D8
D9	W	AR = $00_{(16)}$	Character times per sweep
D9	W	AR = $01_{(16)}$	Characters per line
D9	W	AR = $02_{(16)}$	H SYNC position
D9	W	AR = $03_{(16)}$	H SYNC width
D9	W	AR = $04_{(16)}$	Lines total
D9	W	AR = $05_{(16)}$	Vertical total adjust
D9	W	AR = $06_{(16)}$	Lines displayed
D9	W	AR = $07_{(16)}$	V SYNC position
D9	W	AR = $08_{(16)}$	Interlace control ($00_{(16)}$ used)
D9	W	AR = $09_{(16)}$	Maximum raster address
D9	W	AR = $0A_{(16)}$	Cursor start and blink
D9	W	AR = $0B_{(16)}$	Cursor end
D9	W	AR = $0C_{(16)}$	Display start address (MSB)
D9	W	AR = $0D_{(16)}$	Display start address (LSB)
D9	R/W	AR = $0E_{(16)}$	Cursor location (MSB)
D9	R/W	AR = $0F_{(16)}$	Cursor location (LSB)
D9	R	AR = $10_{(16)}$	Light pen hit address (MSB, not used)
D9	R	AR = $11_{(16)}$	Light pen hit address (LSB, not used)
D8	W		Address register (called AR above)
			DIA PIA
D7	R/W		= D3
D6	R/W	→	= D2
D5	R/W		= D1
D4	R/W	→	= D0
D3	R/W		Control register B (CRB)
D2	R/W	CRB2 = 0	Data direction register B
D2	R/W	CRB2 = 1	Peripheral register B
D1	R/W		Control register A (CRA)
D0	R/W	CRA2 = 0	Data direction register A
D0	R/W	CRA2 = 1	Peripheral register A

Address (Hex)	R/W	Other condition	Addressed unit and register
CF–C8			**Not used**
			MIC PIA
C7	R/W		Control register B (CRB)
C6	R/W	CRB2 = 0	Data direction register B
C6	R/W	CRB2 = 1	Peripheral register B
C5	R/W		Control register A (CRA)
C4	R/W	CRA2 = 0	Data direction register A
C4	R/W	CRA2 = 1	Peripheral register A
			KB ACIA
C3	→		= C1
C2	→		= C0
C1	W		Transmitted data register
C1	R		Received data register
C0	W		Control register
C0	R		Status register
BF–48			**Not used**
47–40	R		**SS3 address**
3F–28			**Not used**
			ADLC
			(Can also be addressed by the DMAC at DMA)
27	→	→	= 23
26	→		= 22
25	→	→	= 21
24	→		= 20
23	W	CR1:0 = 0	Transmitter FIFO (terminate)
23	W	CR1:0 = 1	Control register 4
23	R		Receiver FIFO
22	W		Transmitter FIFO (continue)
22	R		Receiver FIFO
21	W	CR1:0 = 0	Control register 2
21	W	CR1:0 = 1	Control register 3
21	R		Status register 2
20	W		Control register 1 (CR1)
20	R		Status register 1
			DMAC
1F–17			Not used
16	R/W		Data chain
15	R/W		Interrupt control
14	R/W		Priority control
13	R/W		Channel 3 control
12	R/W		Channel 2 control
11	R/W		Channel 1 control
10	R/W		Channel 0 control
0F	R/W		Byte count, channel 3 (LSB)
0E	R/W		Byte count, channel 3 (MSB)
0D	R/W		Address, channel 3 (LSB)
0C	R/W		Address, channel 3 (MSB)
0B	R/W		Byte count, channel 2 (LSB)
0A	R/W		Byte count, channel 2 (MSB)
09	R/W		Address, channel 2 (LSB)
08	R/W		Address, channel 2 (MSB)
07	R/W		Byte count, channel 1 (LSB)
06	R/W		Byte count, channel 1 (MSB)
05	R/W		Address, channel 1 (LSB)
04	R/W		Address, channel 1 (MSB)
03	R/W		Byte count, channel 0 (LSB)
02	R/W		Byte count, channel 0 (MSB)
01	R/W		Address, channel 0 (LSB)
00	R/W		Address, channel 0 (MSB)

DISPLAY TERMINAL CONTROLLER



E9000 3450E

Block Diagram, DU 4113

Appendix 3

Raster Address Modification ROM Program

Explanations

The program is split in four tables in order to give a better general idea of the structure.

The involved signals are explained below:

CGSEL	0 = Full version of character generators.
$\overline{\text{IBMM}}$	0 = IBM mode, i.e. DIA PIA PB6 = 0.
PSLD	1 = Programmed symbol store loading/reading in progress.
$\overline{\text{SR11}}$	0 = Message line presentation.
G2	In IBM mode: 0 = Single plane store (PS3 – 0). 1 = Triple plane store (PS7 – 4). In non-IBM mode: 1 = Presentation is moved 6 sweeps downward.
RA3 – 0	Raster address originating from the cathode ray tube controller or the raster address counter.
TP	0 = Single plane store. 1 = Triple plane store.
GC	1 = Activate green block.
RC	1 = Activate red block.
BC	1 = Activate blue block.
PP3 – 0	Modified raster address.

The symbols in the tables have the following interpretations:

0	Low level.
1	High level.
X	Irrelevant (outputs are normally 0).
a,b,c,d ⋮	Corresponding bits are equal. Et cetera.
	Blank position means same value as the above filled in position.

Program

Full Version, IBM Mode

Inputs									Outputs								Comments
CGSEL	$\overline{\text{IBMM}}$	PSLD	$\overline{\text{SR11}}$	G2	RA3	RA2	RA1	RA0	TP	GC	RC	BC	PP3	PP2	PP1	PP0	
0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	Message line
					0	0	0	1		0	0	1	1	1	0	0	
					0	0	1	0					1	1	0	1	
					0	0	1	1					1	1	1	0	
					0	1	0	0					1	1	1	1	
					0	1	0	1		0	1	0	1	1	0	0	
					0	1	1	0					1	1	0	1	
					0	1	1	1					1	1	1	0	
					1	0	0	0					1	1	1	1	
					1	0	0	1		1	0	0	1	1	0	0	
					1	0	1	0					1	1	0	1	
					1	0	1	1					1	1	1	0	
					1	1	X	X		0	0	0	X	X	X	X	
		0	0	1	0	0	0	0	1	0	0	0	X	X	X	X	
					0	0	0	1		1	1	1	0	0	0	0	
					0	0	1	0					0	0	0	1	
							⋮							⋮			
					1	0	1	1					1	0	1	0	
					1	1	X	X		0	0	0	X	X	X	X	
0	0	0	1	0	0	0	b	a	0	0	0	1	1	1	b	a	Text line
					0	1	b	a		0	1	0	1	1	b	a	
					1	0	b	a		1	0	0	1	1	b	a	
					1	1	X	X		0	0	0	X	X	X	X	
		0	1	1	0	0	b	a	1	1	1	1	0	0	b	a	
					0	1	b	a					0	1	b	a	
					1	0	b	a					1	0	b	a	
					1	1	X	X		0	0	0	X	X	X	X	
0	0	1	X	0	0	0	b	a	0	0	0	1	1	1	b	a	Loading
					0	1	b	a		0	1	0	1	1	b	a	(or reading)
					1	0	b	a		1	0	0	1	0	b	a	of stores 0 – 3
					1	1	X	X		0	0	0	X	X	X	X	
0	0	1	X	1	0	0	b	a	1	1	1	1	0	0	b	a	Loading
					0	1	b	a					0	1	b	a	(or reading)
					1	0	b	a					1	0	b	a	of stores 4 – 7
					1	1	X	X		0	0	0	X	X	X	X	

Reduced Version, IBM Mode

Inputs									Outputs								Comments
CGSEL	$\overline{\text{IBMM}}$	PSLD	$\overline{\text{SR11}}$	G2	RA3	RA2	RA1	RA0	TP	GC	RC	BC	PP3	PP2	PP1	PP0	
1	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	Message line
					0	0	0	1		0	0	1	0	0	0	0	
					0	0	1	0					0	0	0	1	
							⋮								⋮		
					1	0	1	1					1	0	1	0	
					1	1	X	X		0	0	0	X	X	X	X	
		0	0	1	X	X	X	X									
1	0	0	1	0	0	0	b	a	0	0	0	1	0	0	b	a	Text line
					0	1	b	a					0	1	b	a	
					1	0	b	a					1	0	b	a	
					1	1	X	X		0	0	0	X	X	X	X	
		0	1	1	X	X	X	X									
1	0	1	X	0	0	0	b	a	0	0	0	1	0	0	b	a	Loading (or reading) of stores 0 – 3
					0	1	b	a					0	1	b	a	
					1	0	b	a					1	0	b	a	
					1	1	X	X		0	0	0	X	X	X	X	
		1	X	1	X	X	X	X									

Full Version, Non-IBM Mode

Inputs									Outputs								Comments
CGSEL	$\overline{\text{IBMM}}$	PSLD	$\overline{\text{SR11}}$	G2	RA3	RA2	RA1	RA0	TP	GC	RC	BC	PP3	PP2	PP1	PP0	
0	1	0	0	0	0	0	0	0	1	0	0	0	X	X	X	X	Message line
					0	0	0	1		1	1	1	0	0	0	0	
					0	0	1	0					0	0	0	1	
							⋮								⋮		
					1	1	1	1					1	1	1	0	
		0	0	1	0	0	0	0		0	0	0	X	X	X	X	
					0	0	0	1									
							⋮										
					0	1	1	0									
					0	1	1	1		1	1	1	0	0	0	0	
					1	0	0	0					0	0	0	1	
							⋮								⋮		
					1	1	1	1					1	0	0	0	
0	1	0	1	0	d	c	b	a	1	1	1	1	d	c	b	a	Text line
		0	1	1	0	0	0	0		0	0	0	X	X	X	X	
					0	0	0	1									
							⋮										
					0	1	0	1									
					0	1	1	0		1	1	1	0	0	0	0	
					0	1	1	1					0	0	0	1	
							⋮								⋮		
					1	1	1	1					1	0	0	1	
0	1	1	X	0	d	c	b	a	1	1	1	1	d	c	b	a	Loading (or reading) of stores 0 – 3
		1	X	1	X	X	X	X		0	0	0	X	X	X	X	

Reduced Version, Non-IBM Mode

Inputs									Outputs								Comments
CGSEL	$\overline{\text{IBMM}}$	PSLD	$\overline{\text{SR11}}$	G2	RA3	RA2	RA1	RA0	TP	GC	RC	BC	PP3	PP2	PP1	PP0	
1	1	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	Message line
					0	0	0	1		0	0	1	0	0	0	0	
					0	0	1	0					0	0	0	1	
							⋮								⋮		
					1	1	1	1					1	1	1	0	
		0	0	1	0	0	0	0		0	0	0	X	X	X	X	
					0	0	0	1									
							⋮										
					0	1	1	0									
					0	1	1	1		0	0	1	0	0	0	0	
					1	0	0	0					0	0	0	1	
							⋮								⋮		
					1	1	1	1					1	0	0	0	
1	1	0	1	0	d	c	b	a	0	0	0	1	d	c	b	a	Text line
		0	1	1	0	0	0	0		0	0	0	X	X	X	X	
					0	0	0	1									
							⋮										
					0	1	0	1									
					0	1	1	0		0	0	1	0	0	0	0	
					0	1	1	1					0	0	0	1	
							⋮								⋮		
					1	1	1	1					1	0	0	1	
1	1	1	X	0	d	c	b	a	0	0	0	1	d	c	b	a	Loading (or reading) of stores 0 – 3
		1	X	1	X	X	X	X		0	0	0	X	X	X	X	

Appendix 4

Character Sets

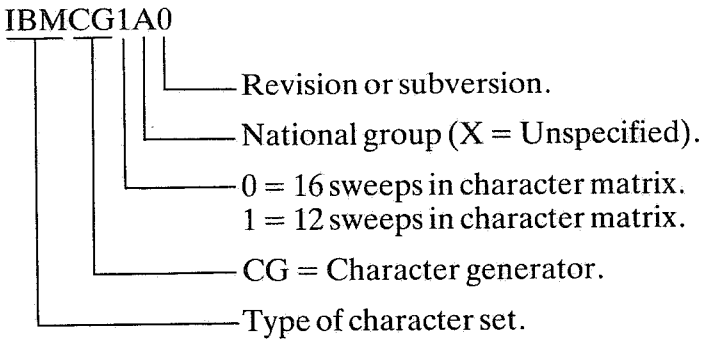
Explanations

This Appendix shows the resulting character on the screen as a function of the used character code. This depends on the character generator contents. Only the locally, from available diskettes, loaded character sets are considered here.

There are five national groups: A (SE/FI, NO, DK, GB, US), B (BE, FR), C (CH), D (DE/AT) and E (ES). Store 0 is normally used for the basic character sets (codes $00_{(16)}$ – $70_{(16)}$). Store 1 is primarily used for the APL character sets while running in IBM emulations.

The message line symbols are included in all IBM and IBM/APL character sets.

The table names, within brackets on the following pages, can be interpreted as shown in the following example:



The following general signs are used:


Illegal code, presented as – (hyphen).


Message line symbols.


Superscript.


Subscript.

IBM Typewriter, National Group A (IBMCG1A0)

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	É	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q								
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	—	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t								
0 1 0 1	5	¢	·	%	5	E	U	e	u								
0 1 1 0	6	¤	}	&	6	F	V	f	v								
0 1 1 1	7	Ω		'	7	G	W	g	w								
1 0 0 0	8	Φ	φ	(	8	H	X	h	x								
1 0 0 1	9			)	9	I	Y	i	y								
1 0 1 0	A		{	*	:	J	Z	j	z								
1 0 1 1	B	Æ	æ	+	;	K	Ä	k	ä								
1 1 0 0	C		¯	,	<	L	Ö	l	ö								
1 1 0 1	D			–	=	M	Å	m	å								
1 1 1 0	E	↑	¯	.	>	N	Ü	n	ü								
1 1 1 1	F	↓	~	/	?	O	_	o	¦								

IBM Typewriter, Alternate, National Group A (IBMCG1A1)

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	É	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q								
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	—	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t								
0 1 0 1	5	¢		%	5	E	U	e	u								
0 1 1 0	6	¤		&	6	F	V	f	v								
0 1 1 1	7	Ω		'	7	G	W	g	w								
1 0 0 0	8	Φ	φ	(	8	H	X	h	x								
1 0 0 1	9			)	9	I	Y	i	y								
1 0 1 0	A			*	:	J	Z	j	z								
1 0 1 1	B	Æ	æ	+	;	K	Ä	k	ä								
1 1 0 0	C		¯	,	<	L	Ö	l	ö								
1 1 0 1	D			–	=	M	Å	m	å								
1 1 1 0	E	↑	¯	.	>	N	Ü	n	ü								
1 1 1 1	F	↓	~	/	?	O	_	o	¦								

IBM Typewriter, Non-diacritic/Diacritic, National Group B (IBMCG1B0)

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	ë	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q	4	■	B					
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	û	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t	A							
0 1 0 1	5	¢		%	5	E	U	e	u	■		■	▶				
0 1 1 0	6	ï	}	&	6	F	V	f	v								
0 1 1 1	7	â	î	'	7	G	W	g	w								
1 0 0 0	8	°	ü	(	8	H	X	h	x		→						
1 0 0 1	9		▶	)	9	I	Y	i	y	?		—	←				
1 0 1 0	A	ì	{	*	:	J	Z	j	z								
1 0 1 1	B	[	]	+	;	K	Ñ	k	ñ								
1 1 0 0	C		¯	,	<	L	Pt	l	ç	à	^	~					
1 1 0 1	D		◀	–	=	M	è	m	ù	a	⇧	_	⇩				
1 1 1 0	E	ê	¯	·	>	N	à	n	¨								
1 1 1 1	F	ô	¿	/	?	O	_	o	¦								

IBM Typewriter, Diacritic, National Group C (IBMCG1C0)

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	ë	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q	4	■	B					
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	û	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t	A							
0 1 0 1	5	¢		%	5	E	U	e	u	■		■	▶				
0 1 1 0	6	ï	}	&	6	F	V	f	v								
0 1 1 1	7	â	î	'	7	G	W	g	w								
1 0 0 0	8	ì	ü	(	8	H	X	h	x		→						
1 0 0 1	9		▶	)	9	I	Y	i	y	?		—	←				
1 0 1 0	A	ò	{	*	:	J	Z	j	z								
1 0 1 1	B	[	]	+	;	K	ä	k	ö								
1 1 0 0	C		¯	,	<	L	‰	l	ç	à	^	~					
1 1 0 1	D		◀	–	=	M	è	m	ù	a	⇧	_	⇩				
1 1 1 0	E	ê	¯	·	>	N	à	n	¨								
1 1 1 1	F	ô	~	/	?	O	_	o	¦								

IBM Typewriter, National Group D (IBMCG1D0)

Bits 7 6 5 4	Hex	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	É	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q								
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	—	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t								
0 1 0 1	5	¢	·	%	5	E	U	e	u								
0 1 1 0	6	¤	}	&	6	F	V	f	v								
0 1 1 1	7	Ω	ß	'	7	G	W	g	w								
1 0 0 0	8			(	8	H	X	h	x								
1 0 0 1	9			)	9	I	Y	i	y								
1 0 1 0	A		{	*	:	J	Z	j	z								
1 0 1 1	B			+	;	K	Ä	k	ä								
1 1 0 0	C		¥	,	<	L	Ö	l	ö								
1 1 0 1	D			–	=	M		m									
1 1 1 0	E	↑	¯	.	>	N	Ü	n	ü								
1 1 1 1	F	↓	~	/	?	O	_	o	¦								

IBM Typewriter, Alternate, National Group D (IBMCG1D1)

Bits 7 6 5 4	Hex	0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	É	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q								
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	—	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t								
0 1 0 1	5	¢		%	5	E	U	e	u								
0 1 1 0	6	¤	}	&	6	F	V	f	v								
0 1 1 1	7	Ω	ß	'	7	G	W	g	w								
1 0 0 0	8			(	8	H	X	h	x								
1 0 0 1	9			)	9	I	Y	i	y								
1 0 1 0	A		{	*	:	J	Z	j	z								
1 0 1 1	B			+	;	K	Ä	k	ä								
1 1 0 0	C		¥	,	<	L	Ö	l	ö								
1 1 0 1	D			–	=	M		m									
1 1 1 0	E	↑	¯	.	>	N	Ü	n	ü								
1 1 1 1	F	↓	~	/	?	O	_	o	¦								

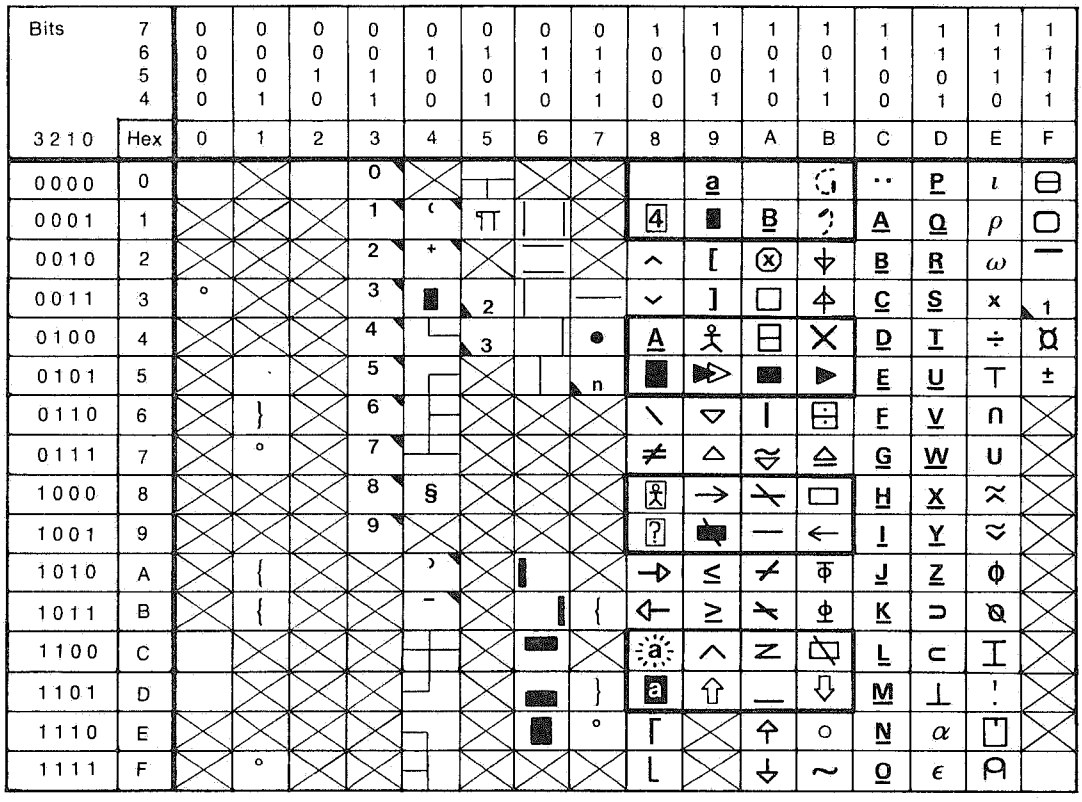
IBM Typewriter, Non-diacritic/Diacritic, National Group E (IBMCG1E0)

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	ë	P	é	p		a						
0 0 0 1	1	^	|	!	1	A	Q	a	q	4		B					
0 0 1 0	2	§	\	"	2	B	R	b	r								
0 0 1 1	3	ú	`	#	3	C	S	c	s								
0 1 0 0	4	£	¬	$	4	D	T	d	t	A							
0 1 0 1	5	¢	·	%	5	E	U	e	u								
0 1 1 0	6	í	}	&	6	F	V	f	v								
0 1 1 1	7	ò	ì	′	7	G	W	g	w								
1 0 0 0	8	°	ü	(	8	H	X	h	x		→						
1 0 0 1	9	~	▶	)	9	I	Y	i	y	?		—	←				
1 0 1 0	A	¡	{	*	:	J	Z	j	z								
1 0 1 1	B	[	]	+	;	K	Ñ	k	ñ								
1 1 0 0	C		¥	,	<	L	Pt	l	ç	à	^	~					
1 1 0 1	D		◀	–	=	M	è	m	ù	a	⇧	_	⇩				
1 1 1 0	E	á	¯	·	>	N	à	n	¨								
1 1 1 1	F	ó	¿	/	?	O	__	o	¦								

IBM Typewriter, Diacritic, National Group E (IBMCG1E1)

Bits 7 6 5 4		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
3 2 1 0	Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 0 0 0	0		@		0	ë	P	é	p		a						Á
0 0 0 1	1	^	|	!	1	A	Q	a	q	4		B					É
0 0 1 0	2	§	\	"	2	B	R	b	r								Í
0 0 1 1	3	ú	`	#	3	C	S	c	s								Ó
0 1 0 0	4	£	¬	$	4	D	T	d	t	A							Ú
0 1 0 1	5	¢		%	5	E	U	e	u								À
0 1 1 0	6	í	}	&	6	F	V	f	v								È
0 1 1 1	7	ò	ì	′	7	G	W	g	w								Ì
1 0 0 0	8	°	ü	(	8	H	X	h	x		→						Ò
1 0 0 1	9	~	▶	)	9	I	Y	i	y	?		—	←				Ù
1 0 1 0	A	¡	{	*	:	J	Z	j	z								Ü
1 0 1 1	B	[	]	+	;	K	Ñ	k	ñ							¨	Ï
1 1 0 0	C		¥	,	<	L	Pt	l	ç	à	^	~				¯	·
1 1 0 1	D		◀	–	=	M	è	m	ù	a	⇧	_	⇩				Ç
1 1 1 0	E	á	¯	·	>	N	à	n	¨								
1 1 1 1	F	ó	¿	/	?	O	__	o	¦								

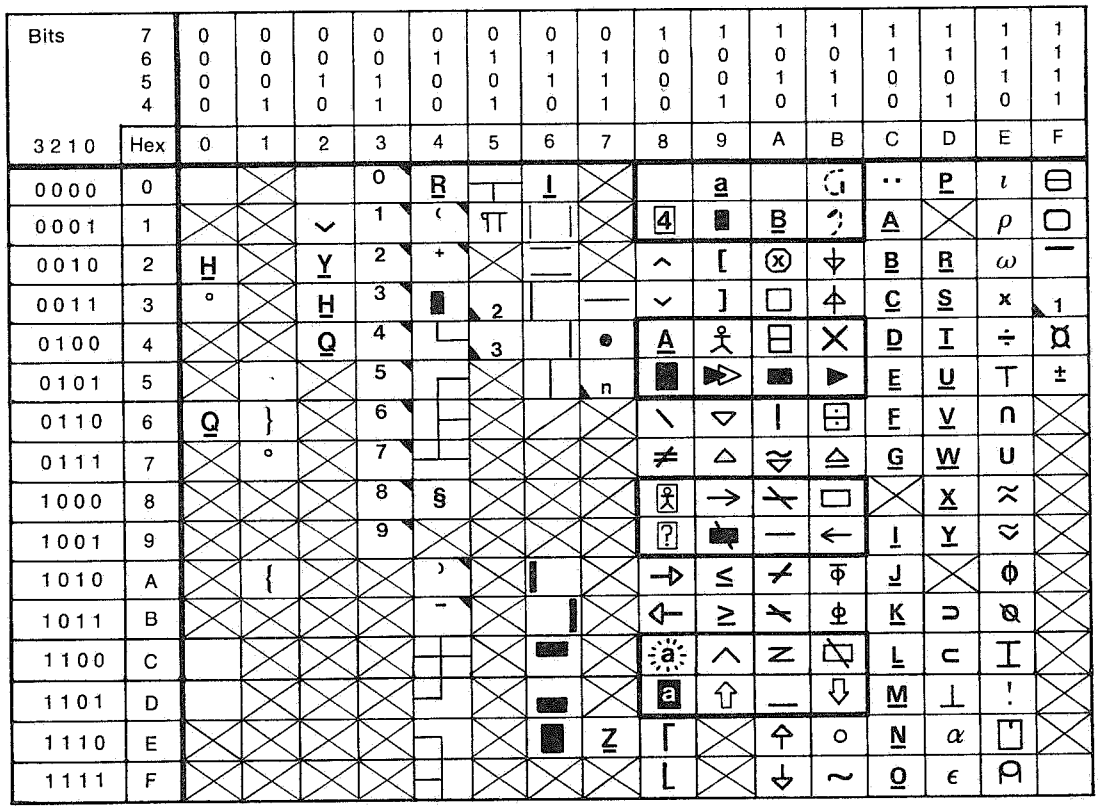
IBM APL/Text, Typewriter, National Group A (APLCG1A0)



National digressions:

	{	}
SE/FI	7B	7D
DK	1B	7D
NO	1B	7D
GB	1A	16
US	1A	16

IBM APL/Text, Typewriter, Alternate, National Group A (APLCG1A1)

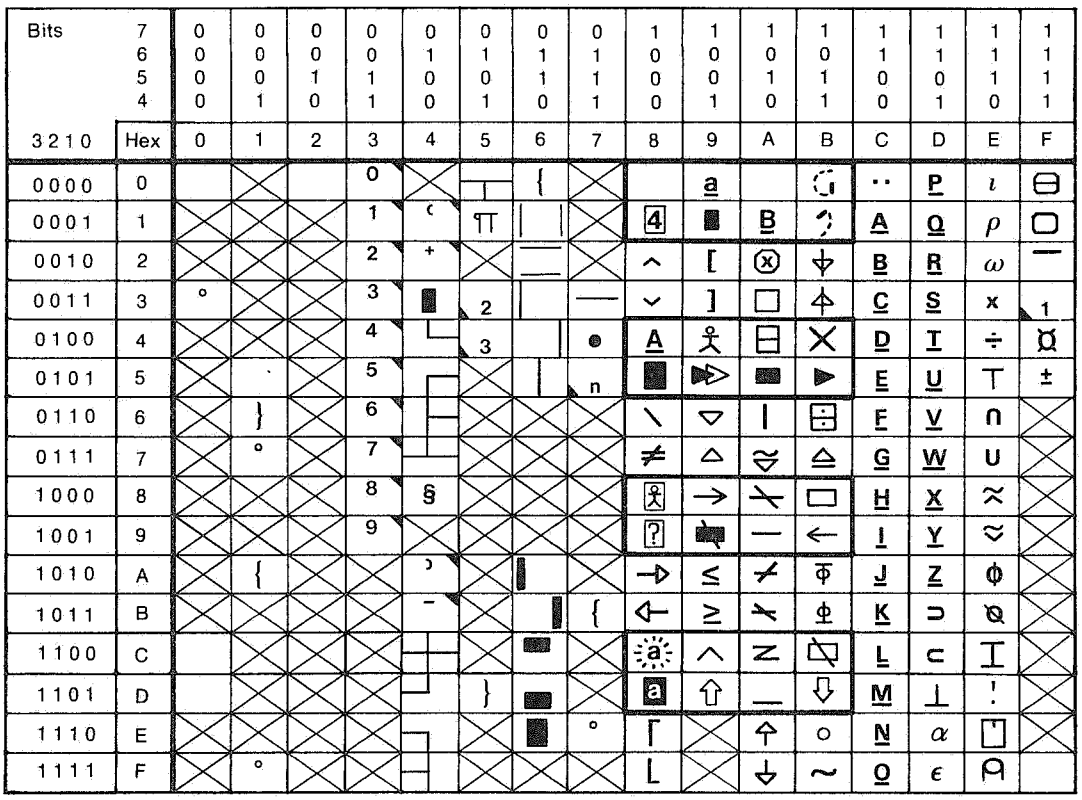


National digressions:

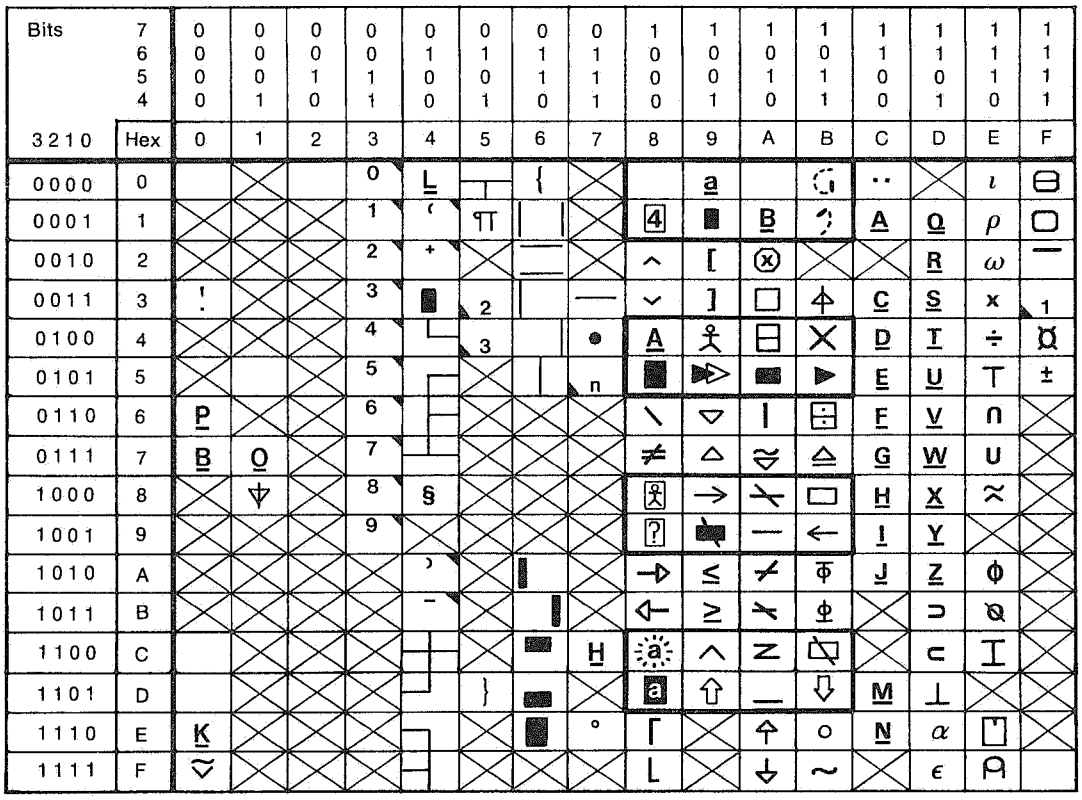
	SE/FI	DK/NO
H	23	02
Q	24	06
}	16	16
{	1A	1A
~	21	83
Y	22	22
R	40	D2
Z	7E	7E
I	60	C9

	{	}
SE/FI	7B	7D
DK	1B	7D
NO	1B	7D

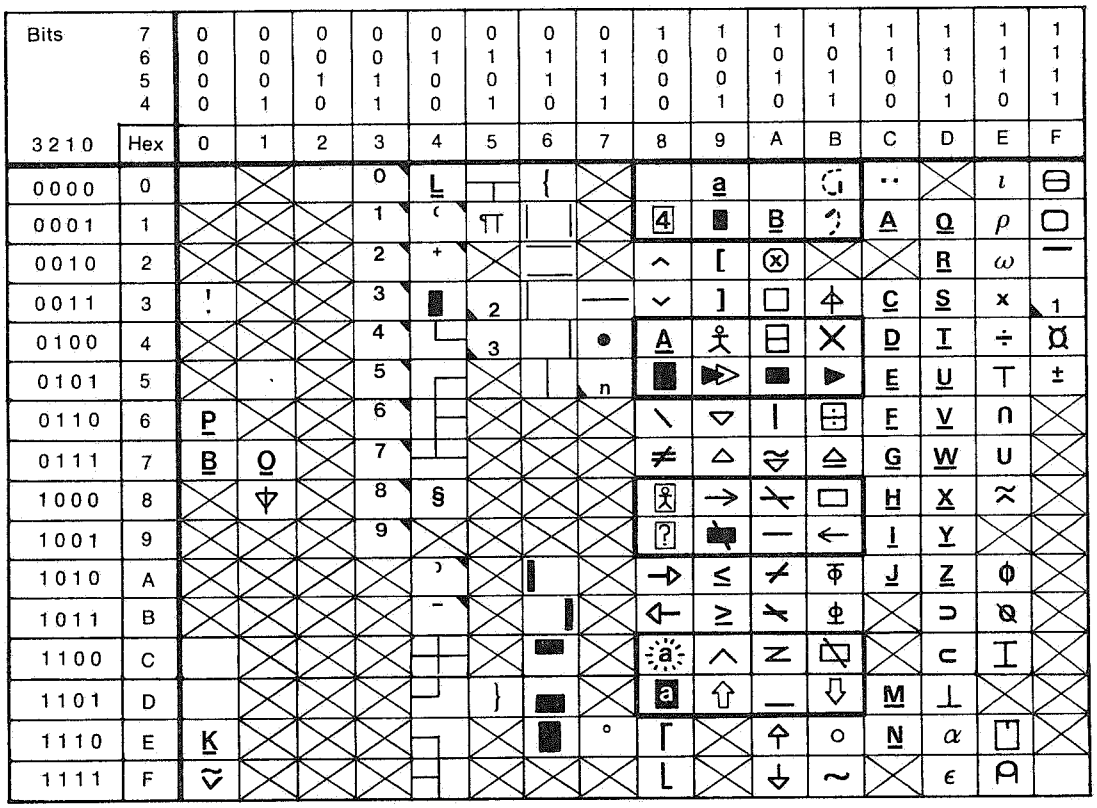
IBM APL/Text, Typewriter, National Group B (APLCG1B0)



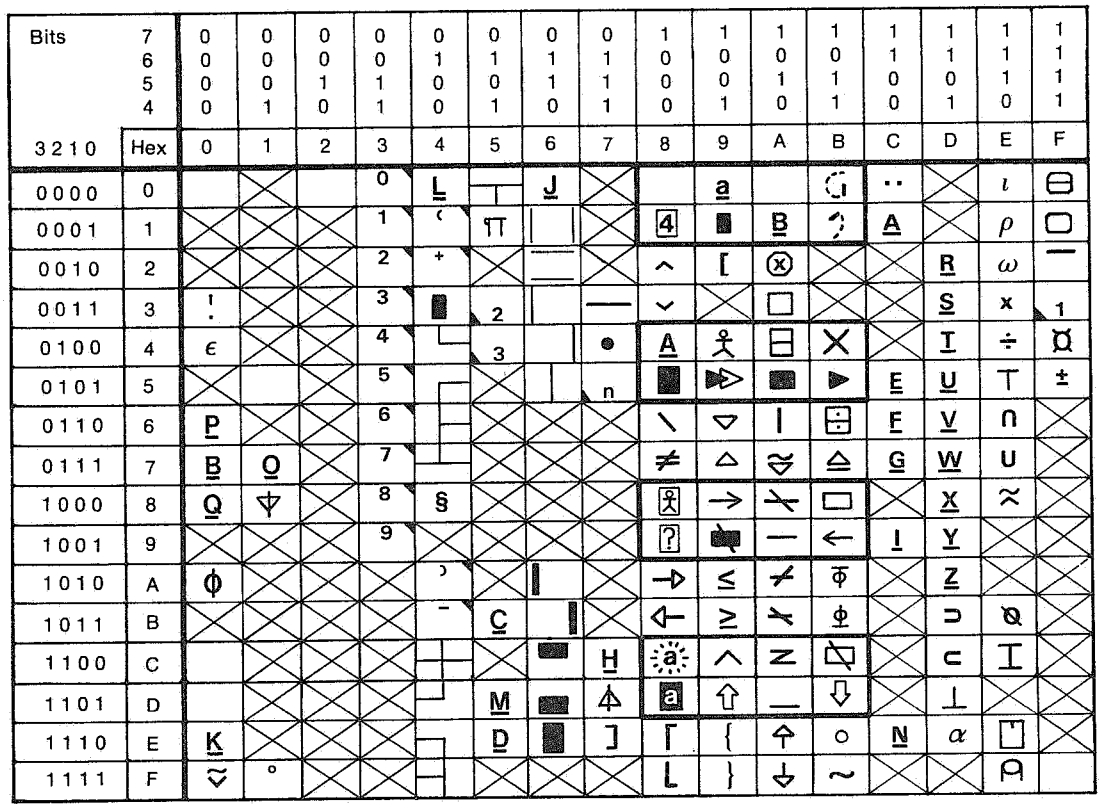
IBM APL/Text, Typewriter, Diacritic, National Group B (APLCG1B1)



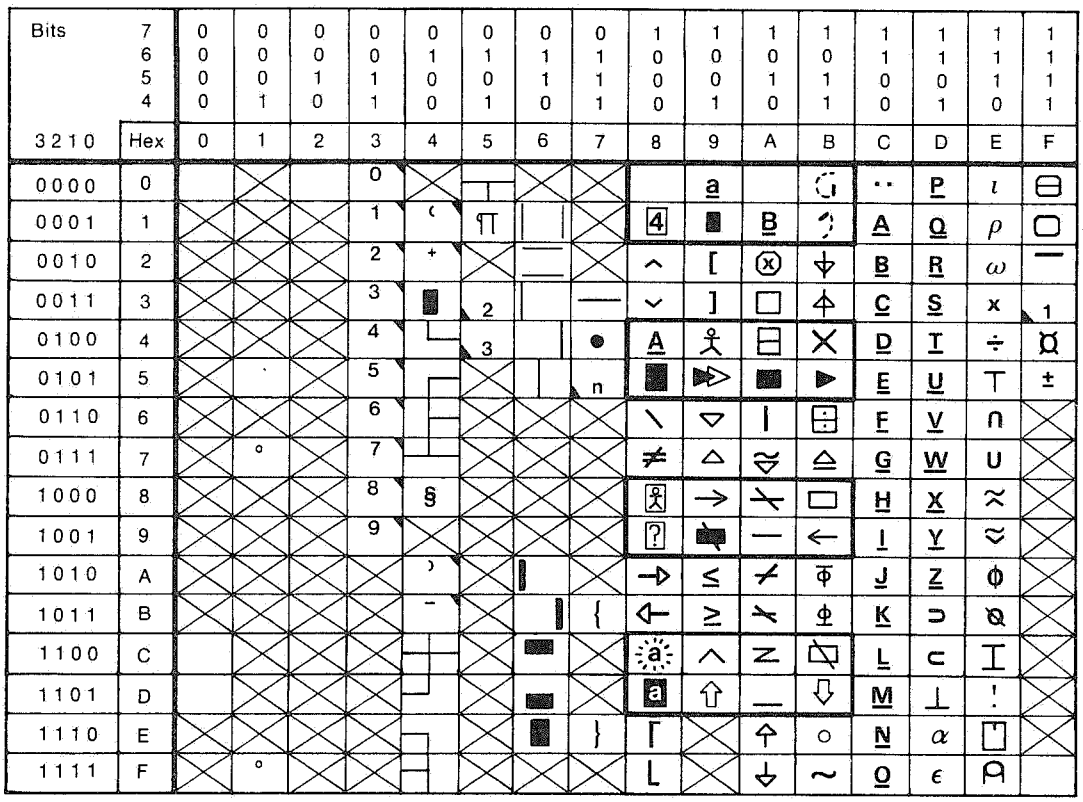
IBM APL/Text, Typewriter, Diacritic, National Group B (APLCG1B2)



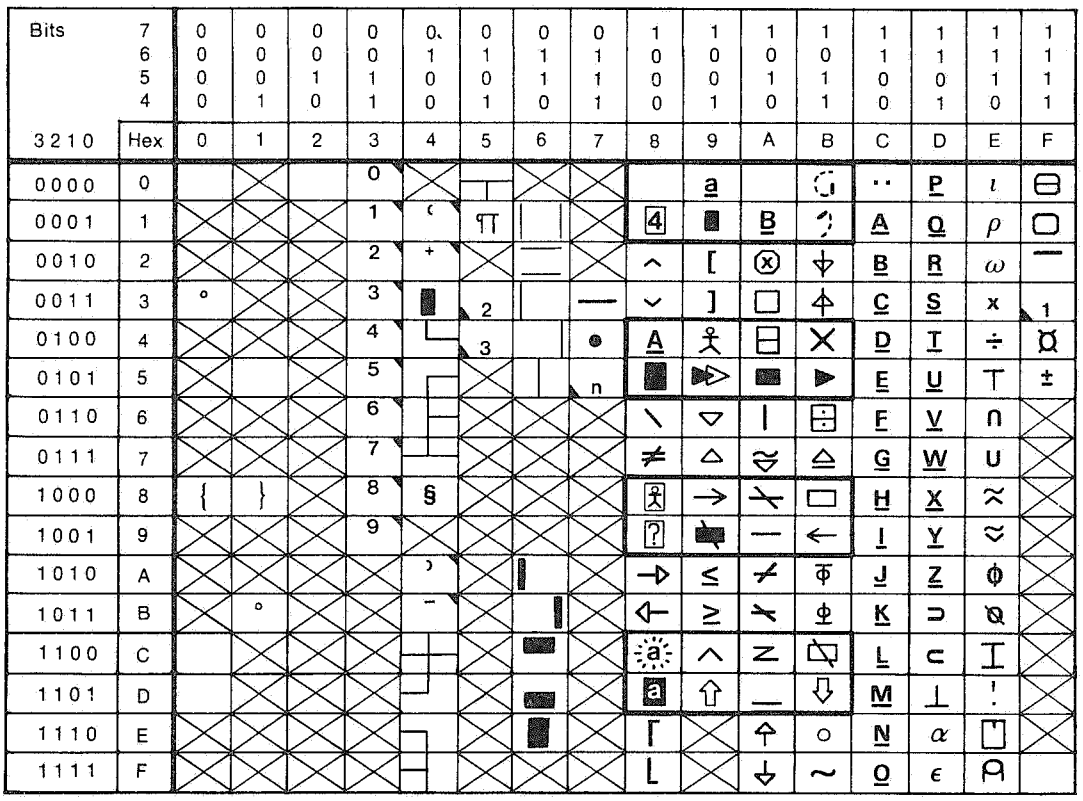
IBM APL/Text, Typewriter, National Group C (APLCG1C0)



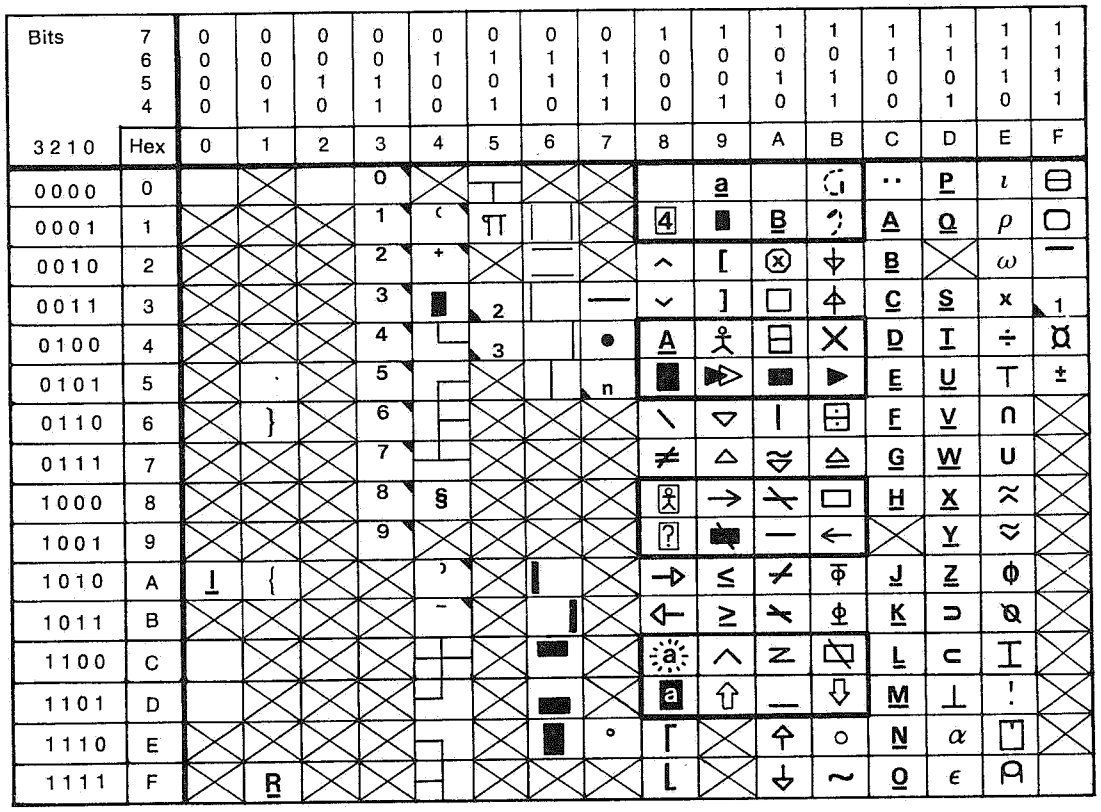
IBM APL/Text, Typewriter, National Group D (APLCG1D0)



IBM APL/Text, Typewriter, Alternate, National Group D (APLCG1D1)



IBM APL/Text, Typewriter, National Group E (APLCG1E0)



DATASAAB

Keyboard

Contents

Figures

EE361-810
1978-11-13

General

The keyboard consists of the keyboard unit, KBU and, mostly, the keyboard expansion unit, KXU.

The keyboard unit, KBU, contains a key matrix, a loudspeaker for certain audible signals, indicator lamps and connectors for magnetic identification device and keyboard expansion unit, KXU. Its logic contains a two-chip microprocessor. For maintenance purpose the logic is accessible with microprocessor control panel, MCP, via a 40 pin clip and a 12-pin connector (P4).

The purpose of the keyboard logic is

- to scan the key matrix regarding depressed keys and to store the code number of such keys.
- to read ID data when interrupts are fed from the identification device and to check continuously if the ID card is in or out.
- to maintain communication with the display terminal controller, DTC.

All measures taken by the keyboard logic are under supervision of the microprocessor and its firmware program.
The expansion unit, KXU, is functionally only an extension of the key matrix and its scanning register.

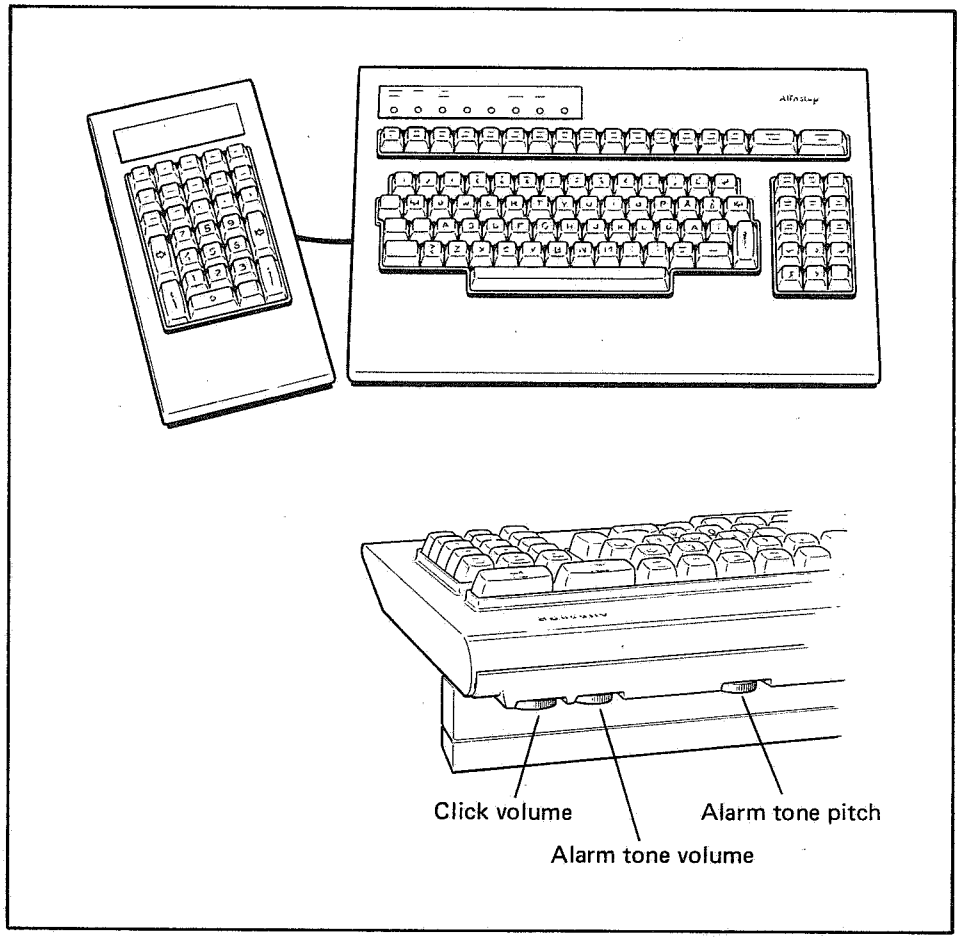


Fig. 1. Keyboard

Brief Outline

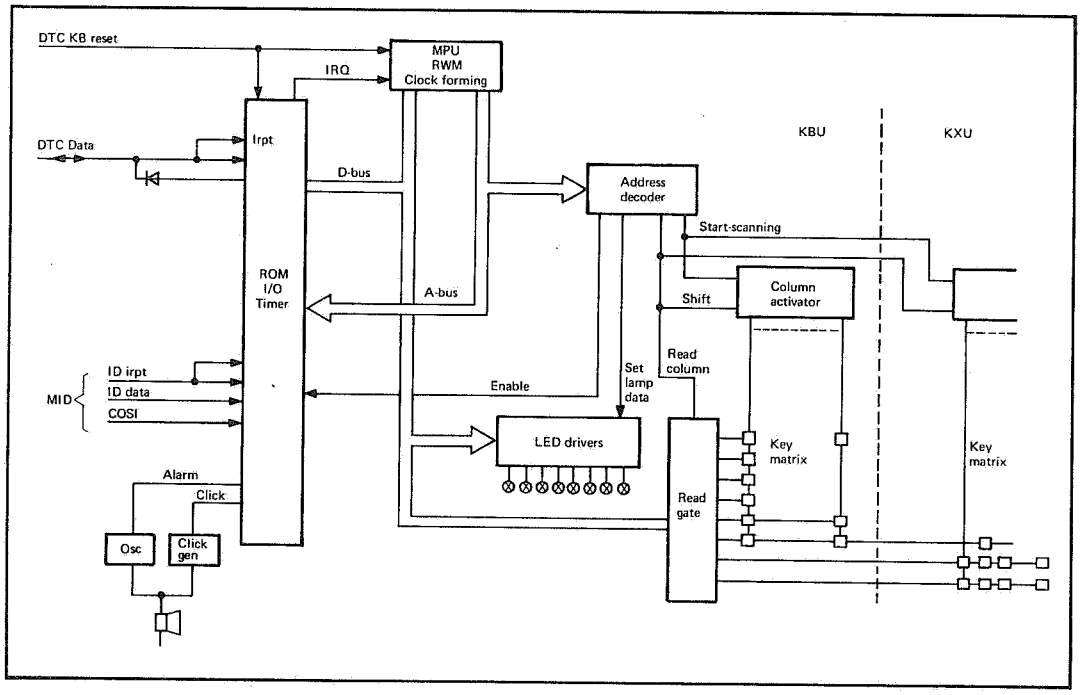


Fig. 2. Keyboard block diagram

Microcomputer

The microprocessor is called 6802 and uses the same instruction repertoire as 6800. However, it contains a 128 bytes RWM area and a clock forming circuit. It is meant to be used together with 6846 and thus forming a two chip microcomputer. The 6846 contains 2 kbytes ROM, a timer circuit and an eight bit parallel I/O register.

The register in 6846 interfaces the magnetic identification device, MID, the DTC and the discrete circuits controlling the alarm and click loudspeaker. The ROM contains the keyboard program.

Address Decoding

Four high order address bits from the MPU are fed to an address decoder. The output from this decoder provides control pulses for matrix scanning and lamp drivers and circuit enable for 6846 (ROM, I/O and timer). The eleven low order address bits address the program memory.

Lamp Control

The eight LED lamps are lit according to orders from DTC. A lamp used as "shift" lamp, may also be lit by KBC. Current lamp status is stored in a register in the microprocessor RWM area. The content of this register is duplicated in the lamp driving register every program loop.

Scanning of the Key Matrix

The key matrix is arranged in 16 columns by 8 rows (including the expansion unit). The eight rows are connected to the data bus via tri-state read gates, thus forming a byte output. The matrix is scanned from a 16 bits shift register, through which a zero is shifted every scan cycle. The 16 outputs of the shift register correspond to the 16 key columns. Thus the shifted zero indicates the columns one at a time, and the column is thereby activated. At every state of the zero the read gates are opened and the "row byte" is fetched by the microprocessor.
Note that the keyboard expansion unit has its own column activating shift register, but shares the tri-state read gates with the KBU.
As seen from the microprocessor the KBU key matrix and KXU key matrix only appear as one matrix.
The result of each scan cycle is compared with the results of the previous cycle. Should they differ, indicating depressed keys, the change in state is transferred to DTC as key numbers.
The DTC program then converts these numbers into suitable information (e g ASCII characters).

MID Interface

The magnetic identification device, MID, feeds three signals to the keyboard. When drawing the ID card through the reader slit, ID interrupt and ID data are applied via two twisted pairs.
The interrupt to the ROM-I/O-timer circuit will be distributed to the microprocessor, which thereby senses the state (one or zero) of the databit. Data is fed in series from MID and is applied to one of the I/O register inputs of 6846. ID data is stored in the RWM area of the microprocessor, wherefrom it can be transferred to DTC if requested.

When the ID card has reached the bottom of the reader, and thus all ID information is stored in the keyboard, the signal Card out sensor interface, COSI, indicates that the card remains in the reader. This signal is continuously scanned by the program in the keyboard and as soon as the state changes, this is transmitted as status information to DTC.

Power Distribution

The keyboard unit is fed with +5 V/0 V from the terminal power supply via the cable from DTC. This voltage is also fed from KBU to MID and KXU.

Communication Procedures

The keyboard logic continuously communicates with DTC according to a certain protocol. DTC transfers one order byte and one or more data bytes to the keyboard. The keyboard replies with one or more status bytes and sometimes one or more data bytes.

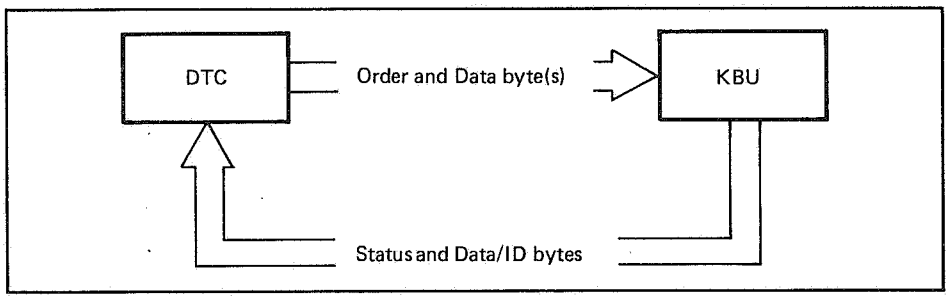
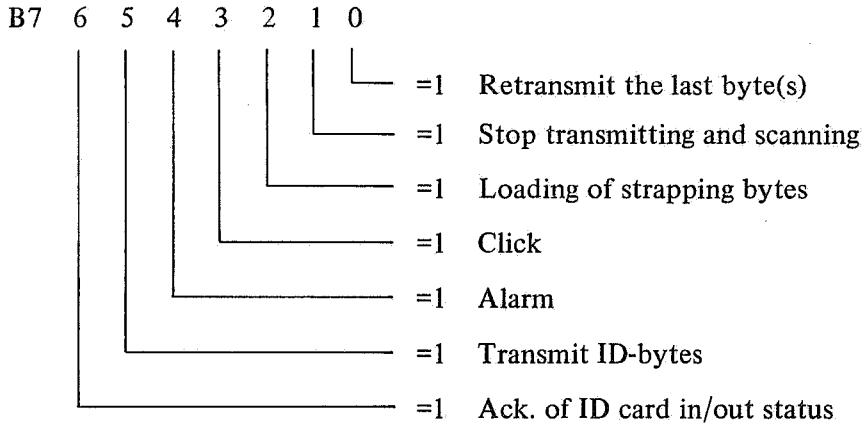


Fig. 3. Communication sequence

The first data bit in a byte is defined as bit 0 and the last one as bit 7. A parity bit follows, making even parity.
The communication is asynchronous with start and stop bits.
The bit rate is 1200 Hz.

Order Byte

The first byte in the procedure is defined as the order byte transferred to the keyboard. If bit 7 in this byte is zero, it contains the following information.



Bit 0 implies, when set, an order to the keyboard to retransmit the previously transmitted data byte or data bytes e. g. if some kind of fault has been detected by DTC.

Bit 1 The order byte from DTC is always considered as a poll, upon which a status byte should be answered. However, when bit 1 is set transmitting of status and data is stopped as well as scanning of the key matrix.

Bit 2 indicates that the succeeding byte(s) contains strapping data, see paragraph "Strapping Bytes".

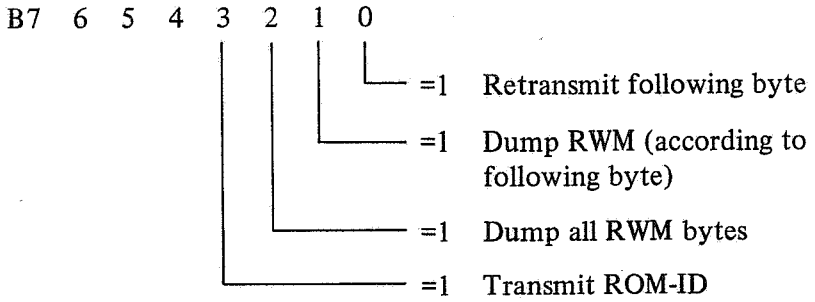
Bit 3 always orders the keyboard to induce a click sound via the loudspeaker, e.g. as a tellback of a key depression. The keyboard can be assigned to click by itself at the detection of a depressed key, see paragraph "Strapping Bytes".

Bit 4 orders the keyboard to induce an alarm via the loudspeaker.

Bit 5 orders the keyboard to transfer ID data which are stored in the ID buffer in the microprocessor RWM area.

Bit 6 implies an acknowledgement that ID status (card in/card out) has been received. The keyboard keeps sending ID status until this bit is received as a one.

If bit 7 is a one, the keyboard system is set into test mode and thus the order byte contains the following information.



Bit 0 indicates that the succeeding data byte from DTC shall be "echoed" back by the keyboard logic so that DTC can check that no transmission faults are at hand.

Bit 1 implies that the succeeding data byte should be considered as an address to the RWM area and that the content of this memory cell shall be transferred to DTC.

Bit 2 means that the content of the whole RWM area shall be sent to DTC and thus that the succeeding byte should be ignored.

Bit 3 All firmware (stored in ROM) has an identification sequence of 30 bytes programmed in a certain area. When bit 3 is set, this area is addressed by the microprocessor and valid ROM ID data is fed to DTC. This makes it possible to check if correct program is used in the keyboard. The succeeding byte is ignored.

During test mode bits 4, 5 and 6 are not valid in the order byte.

Data Byte from DTC

When the order byte bits 7 and 2 are equal to zero, the data byte is considered as lamp data and the lamps are lit accordingly. A one in a bit position corresponds to a lit lamp.
The meaning of the lamps are software dependent.
Bit 7 corresponds to the leftmost lamp-position on the keyboard panel and bit 0 to the rightmost position.

Test mode is defined by order byte bit 7 = 1. Depending on the states of bit 0 and 1 (see previous paragraph), the byte is either

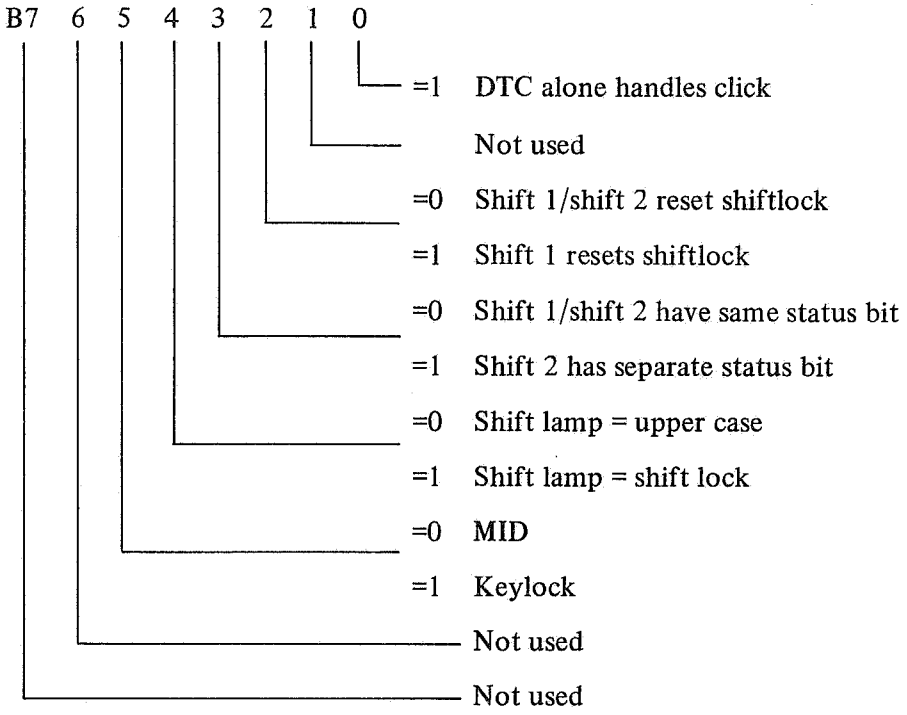
- a byte to be echoed
- an address to RWM
- ignored

Strapping Bytes

In order to avoid designing with hardware strappings to alter certain functions in the keyboard, "software strappings" are used.
These strapping bytes are fed from DTC program mainly after a system reset.
The strapping bytes are stored in the microprocessor RWM area.

When the order byte bit 7 = 0 and bit 2 = 1 up to eight strapping bytes can be transferred from DTC to the keyboard logic. This string of strapping bytes ends with a stop byte with the hex code FF. Thus no strapping byte may use the code FF.

Strapping byte 1



Bit 0 decides if the DTC alone shall handle the click function or if the keyboard shall handle the tellback click function at the depression (not at repetition) of a key.

Bit 1 Not used

Bit 2 decides if both shift 1 function and shift 2 function or only shift 1 function shall reset shift lock (see further on).

Bit 3 decides whether or not shift 1 and shift 2 shall have the same status bit (also see status byte from keyboard).

Bit 4 decides if the shift lamp shall indicate upper case or shift lock.

Bit 5 indicates if a magnetic identification device is used or if a keylock is mounted.

Strapping byte 2

Defines which lamp that shall be used as shift lamp. If this byte is 00 no lamp is handled by KBC. If bit 7 = 1 the leftmost lamp functions as shift lamp and may be handled by KBC etc.

Strapping bytes 3–8 indicate key numbers, i.e. key positions on the keyboard, which are assigned to certain functions such as shift lock. Thus it is possible to assign any key in the matrix to such a function.
There are four different "shift" possibilities that can be strapped in the keyboard RWM. Shift 1 (which always releases the shift lock function) is usually used as an ordinary upper/lower case shift key similar to a typewriter keyboard.
Shift 2 can be used as a shift. However, it is possible to give shift 2 another function. In that case the shift 2 key puts the system into another mode. This could be used e.g. as a data entry ALPHA SHIFT key. If shift 1 and shift 2 are given different meanings, each key on the keyboard can have four different meanings i.e. normal upper, normal lower, shift 2 upper and shift 2 lower.
Shift 3 and shift 4 provide two more different shift modes.

Strapping byte 3

Bits 6–0 indicate the number of the key that shall handle the shift lock function (also see strapping byte 1 bits 2 and 4).

Strapping byte 4

Bits 6–0 indicate the key number for shift 1.

Strapping byte 5

Bits 6–0 indicate the key number for shift 2 (also see strapping byte 1 bits 2 and 3).

Strapping byte 6

Bits 6–0 indicate the key number for shift 3.

Strapping byte 7

Bits 6–0 indicate the key number for shift 4.

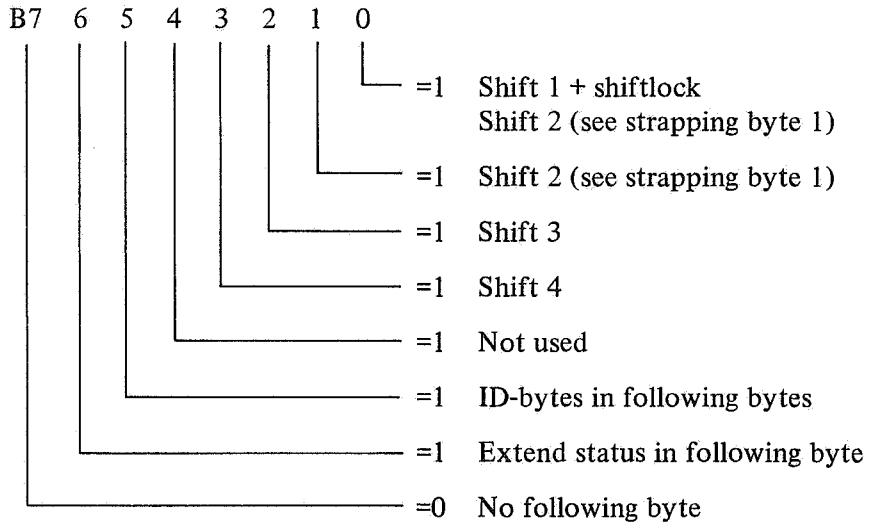
Strapping byte 8

Bits 6–0 indicate the key number for the repeat key. If this is used, the "typeamatic" function is omitted, i.e. the automatic repeat function that normally is activated when any key is depressed more than 0,5 seconds. The repetition frequency is about 25 Hz.

Note that the keyboard only notes the key-number of the various shift and repeat functions. The measures to be taken when any of these keys is depressed is decided by the DTC program.

Status Byte from Keyboard

When order and data (or strapping data) have been received from DTC, the keyboard replies with status and data unless bit 1 (stop transmitting and scanning) was set in the received order byte.



Bit 0 indicates when shift lock, shift 1 or shift 2 has been depressed. Note that shift 2 in this case must have the same function as shift 1.

Bit 1 indicates that shift 2 has been depressed. This status bit is only used when shift 1 and 2 have different meanings.

Bits 2 and 3 indicate that keys assigned to shift 3 and 4 have been depressed.

Bit 5 indicates that ID data succeeds the status byte.

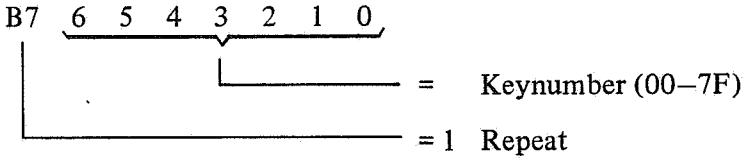
Bit 6 indicates that an extended status byte succeeds the status byte.

Bit 7 = 0 implies that no data succeeds in this transfer.

= 1 implies that key-number, ID data or extended status succeeds the status byte.

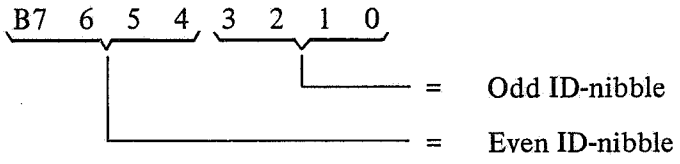
Data Bytes from Keyboard

When the status byte has bit 7 = 1 and bits 6–4 = 0 a data byte containing the number of a depressed key is transferred after the status byte.



ID Bytes from Keyboard

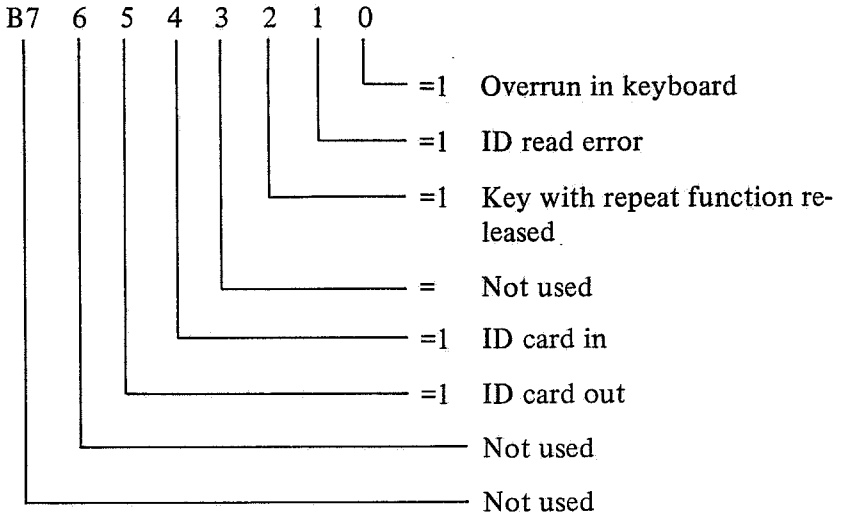
When bits 5 and 7 = 1 and the rest = 0 in the status byte, ID data succeeds the status byte.



The ID bytes include start-, stop-, LRC-byte. The string of ID bytes ends with a stop byte (code "AA").

Extended Status from Keyboard

When bits 7 and 6 = 1 and bits 5–0 = 0 an extended status byte succeeds the status byte.



Bit 0 indicates that the keyboard has lost one or more key depressions due to full FIFO buffer. This buffer is in the RWM area and can hold four key numbers before overrun.

Bit 1 indicates that the keyboard has detected a parity fault or a fault at the longitudinal redundancy check while reading the ID data from the identification device. (Bit 5 is also set).

Bit 2 indicates that a key with typeamatic repeat function or the repeat key (see strapping byte 8 above) has been released.

Bit 3 Not used.

Bit 4 is set if an ID card is inserted in the MID and the ID bytes have been correctly read by KBC. This bit is reset after KBC gets an acknowledge of ID card in/out status from DTC.

Bit 5 is set when the ID card has been taken out of the MID or if a fault is detected during the reading (see bit 1). Also reset at acknowledge. See below.

Note that the keyboard transfers bit 4 or 5 until DTC acknowledges with bit 6 in the order byte.

Detailed Description

See keyboard logic diagram.

Keyboard RWM Test

At KB reset, the keyboard program begins with a test of the 128 byte RWM area in the microprocessor. This program is called KB RAMT. Thereby every two bytes are fed with AA (hex) and the other with the inversion, 55. This means that the memory contains AA, 55, AA, 55 etc.
The program then waits for 4 ms and then reads the memory to check that the information remains and is correctly stored in the memory. If no error is detected, the program stores the inverted information, i.e. 55, AA, 55, AA etc.
After another 4 ms delay, the memory is checked again and if no error is detected the program writes 00 into cell 7F, 01 into cell 7E etc. After another 4 ms delay, the memory is checked again and if no error is detected the program proceeds with testing the program memory.
Should, however, an error appear during the test, the program enters a loop which forces the eight lamps of the panel to blink simultaneously with a frequency of 5 Hz. At the same time the program sets pin 2 at connector P1 to low level, indicating error and pin 4 in connector P4 to high level, indicating that the error is an RWM error. This loop can only be broken by a KB reset.

Keyboard Program Memory Test

When no error is found in the RWM area, the program proceeds by checking the program memory. This test is called KB CRCT.
The program thereby reads all 2 kbytes of the ROM and at the same time accumulates a 16 bit block check character, BCC. This character is generated according to the cyclic redundancy check, CRC–16 method.
When all 2 kbytes have been examined in this way, and thus a 16 bit word has been accumulated, the word is compared with a previously calculated word. This previously calculated word is stored in two successive addresses in the ROM.

Should the two words match, this means that the program memory is correct and without faults.
However, if the patterns do not match, the program enters a loop which forces the eight lamps on the panel to blink sequentially from left to right with a "jump" frequency of 5 Hz. The program also sets pin 2 in connector P1 to low level, indicating error and pin 4 at connector P4 to low level, indicating fault in the program memory. The loop can only be broken by a KB reset.

Initiation after KB Reset

If, after a KB reset, no RWM or ROM errors have been detected, the program proceeds with setting certain circuits into initial state.
These circuits are

- The peripheral control register, data direction register and peripheral data register in ROM-I/O-timer.
  (Also see section "Microcomputer" – paragraph "PIA").
- The RWM area is cleared, FIFO pointers are set to start address and the "stop transmitting and scanning" bit in the order byte is set.
- Presumed strapping bytes regarding shift and repeat are stored in RWM.
- The column activating shift register is cleared.

Main Loop

When the memory tests and the initiation have been made, the keyboard is ready to receive order from DTC.
When an order appears having the "stop transmitting and scanning" bit reset, the program enters the main loop or working loop, which is called KB scan.
This part of the program has the succeeding main functions:

- to light the lamps on the panel according to lamp data from DTC. These data are stored in the RWM area.
- to check whether or not an ID card remains in the reader and indicate this to the DTC the succeeding communication cycle.

- scan all keys of the keyboard and compare the present state of the keys with the state of the previous scanning.

- decide if the code for a depressed key should be transferred repeatingly to DTC.

- finally the program handles two interrupts. One appears at each bit serially fed from MID while reading an ID card. The other one appears at the start bit at every byte fed from DTC.

Thus the main loop begins with the microprocessor addressing the RWM cell containing current lamp data. If the keyboard handles the shift lamp, lamp data from DTC is OR:ed with the shift lamp bit.
Lamp data is then applied to the data bus while address bits 15–11 affect the address decoder so that the lamp driving register is clocked whereby lamp data is set into this register.

When lamp data is set, the program proceeds with testing the signal COSI from the MID. This signal tells whether there is an ID card resting in the reader or not.
This part of the program checks if the signal has changed state regarding the previous scan. If so, DTC is informed of the change in the succeeding extended status byte (bit 4 or 5) from the keyboard. The keyboard keeps on sending this status until DTC acknowledges it with bit 6 in the order byte. However, if the state of the signal has changed to "ID card out" the program checks if MID presently reads a card before informing DTC.

The keyboard matrix, defined by 16 columns and 8 rows, is scanned every main loop in the following manner. The program sets a zero in the first stage of the column activating shift register. This implies that the first key column of Hall switches i.e. the first eight keys, are activated.

Then the program enables the eight tri-state buffers, whereby the state of the eight activated Hall switches are applied to the data bus. At this time the column activating shift register is clocked by the program and the zero proceeds to the second stage of the register.
The byte on the data bus is fetched and stored in a 16 byte RWM buffer called "new key values". The zero in the shift register now activates the second column of eight Hall switches, which are fetched in the same manner as the first. The zero is stepped again to indicate the third column and so on until all 16 columns are fetched by the program and stored successively in the "new key values" buffer.

The program then starts to compare the 16 bytes, one by one, with the corresponding 16 bytes in another buffer called "old key values" which contains the states of the Hall switches from the previous scanning.
Should a difference appear at the comparison, this means that a key has been depressed, or that a depressed key has been released. If a key has been depressed, the program starts to determine the number of the current key by comparing each bit until a difference is found. At this state the program

knows the number of the byte i.e. matrix column and also the number of the bit i.e. matrix row. Thus the key at hand is identified and the key number, consisting of column nr (bits 6 through 3) and row nr (bits 2 through 0), is fed to the output FIFO buffer together with the status byte at hand. The status byte thereby indicates if shift is depressed, which also is determined by the program.

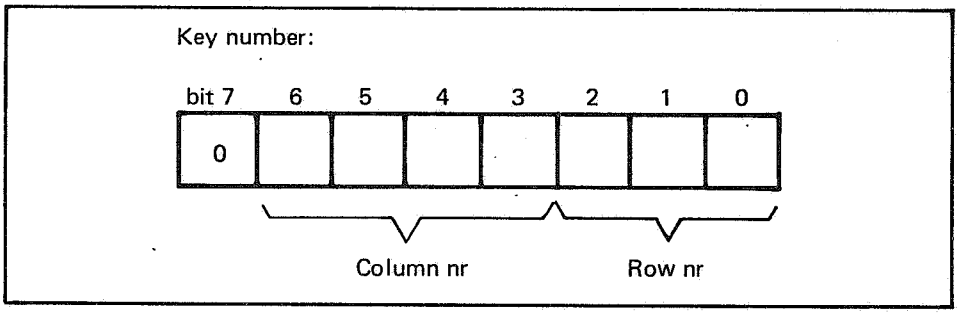


Fig. 4. Key code bit designation

The FIFO buffer holds eight bytes, which means that four key numbers can be stored. In the improbable case of more than one hit at a scan cycle, all keys will be stored in the FIFO in the order they were found.
Any byte in the "new key values" buffer differing from the corresponding one in the "old key values" buffer is transferred to the corresponding location in the "old key values" buffer. The matrix is scanned at least every 2 ms.

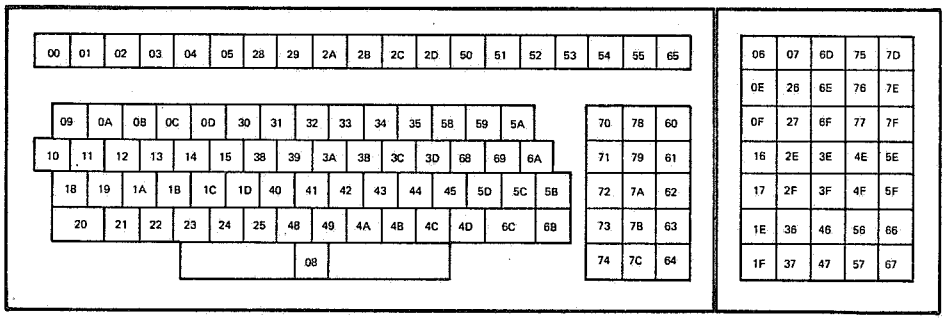


Fig. 5. Key codes

When the program has finished the matrix scanning, it proceeds with the repeat function.
Thereby the strapping bytes are checked to find if any key is assigned as repeat key. If so, the program checks if this key is depressed. If it is depressed and the FIFO is empty, the number of the last depressed key is set into the output FIFO buffer once again. This is made until the repeat key is released.
If no key is assigned as repetition key, the "typeamatic" function is valid. This means that when any key on the keyboard is depressed longer than 0.5 s its number will be set into the output FIFO buffer as described above. The repetition ceases when the key is released.
After the end of repetition, bit 2 of the extended status byte from KB is set.

Interrupts from DTC

The main loop can at any time be interrupted by the DTC having decided to start a communication sequence. The transfer is made asynchronous with one start (low) and one stop bit (high) for each byte.
When the data line from DTC drops to low at the beginning of a communication sequence, this is sensed by inputs CP2 and P0 of the ROM, I/O, timer. P0 is assigned as data input and CP2 causes an IRQ to the microprocessor. Thereby the program sets the timer to measure half the decided bit time, which is 1/1200s. When this time has elapsed, the timer causes an IRQ. The state of the data line is tested and if it still is low, it is considered as a start bit.

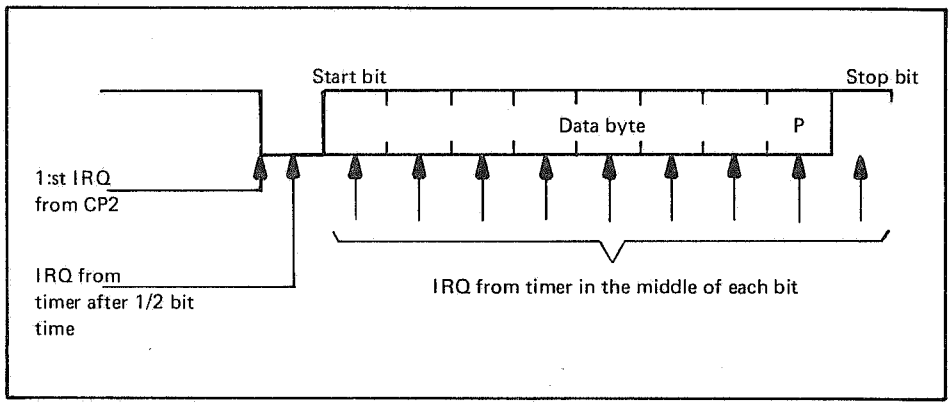


Fig. 6. Bit timing, received data

The timer is then set to measure the whole bit time which means that the succeeding IRQ appears in the middle of each bit. Thus the program can strobe the databit at the most suitable time. When the whole message (at least two bytes) is received in this manner, the program changes to transmit mode.
The first byte from DTC is always an order byte. Every bit in this byte must be examined by the program in order to find out what the keyboard is expected to perform.

Transmission to DTC

The timer in the 6846 circuit is also used during transmission from the keyboard to DTC. The program then applies a start bit to output P1 and starts the timer to measure one bit time. When the time has elapsed, IRQ is fed to the microprocessor, whereby the program serially feeds the rest of the message.
For each bit the timer measures its length.

Interrupt from MID

When an ID card is drawn through the slit of the MID, a train of clock pulses and of data bits are fed to the 6846 circuit. Serial data is applied to input P2 and clock pulses to CPl input. The clock pulse causes the ROM–I/O–timer to feed IRQ to the microprocessor. The program thereby fetches the current data bit and this is repeated until the stop code from the ID card is detected and the clock pulses cease.

Program Summary

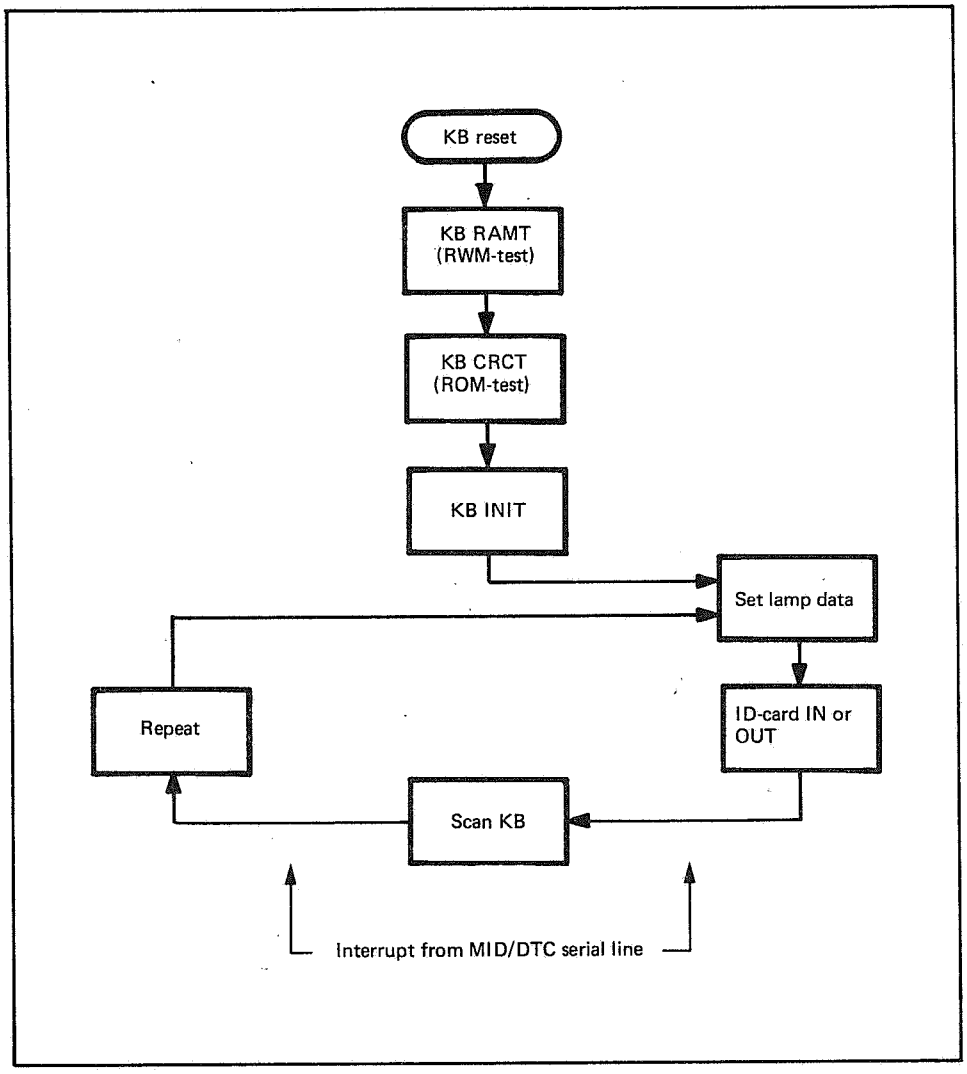


Fig. 7. Coarse flowchart

Keyboard, KBU 4143, KXU 4146, MSR 4136

Contents

General

This chapter describes the keyboard unit KBU 4143 and its auxiliary units the keyboard expansion unit KXU 4146 and the magnetic stripe reader MSR 4136. The KBU is intended for connection to a display unit but might be connected to other devices in some applications. When used, the KXU and MSR are connected to the KBU.

Functions

The KBU and, if connected, the KXU and the MSR are controlled by the display unit via a serial interface. On command from the display unit, the KBU performs:

- Scanning of the KBU and KXU key matrices.
- Storing of key codes for pressed keys.
- Processing of data from the MSR.
- Storing of MSR data.
- Sending of stored information to the display unit.
- Control of a LED display.

Mechanical Design

The main parts of the KBU are, as shown in Fig. 1:

- Case, formed by two parts.
- Switchboard with key matrix and LED display.
- Logic board with microprocessor (CPT), memory, interfacing circuitry and loudspeaker. The board also contains contact springs for a battery mounted through the case bottom and the board.

The switchboard is connected to the logic board by means of a cable with connector. The logic board is equipped with two connectors for connecting of external equipment. The connectors are accessible at the rear side of the case.

The main parts of the KXU are, as shown in Fig. 2:

- Case, formed by two parts.
- Switchboard, with key matrix and decoding logic.

The KXU is equipped with a cable with connector intended for connection to the corresponding KBU connector.

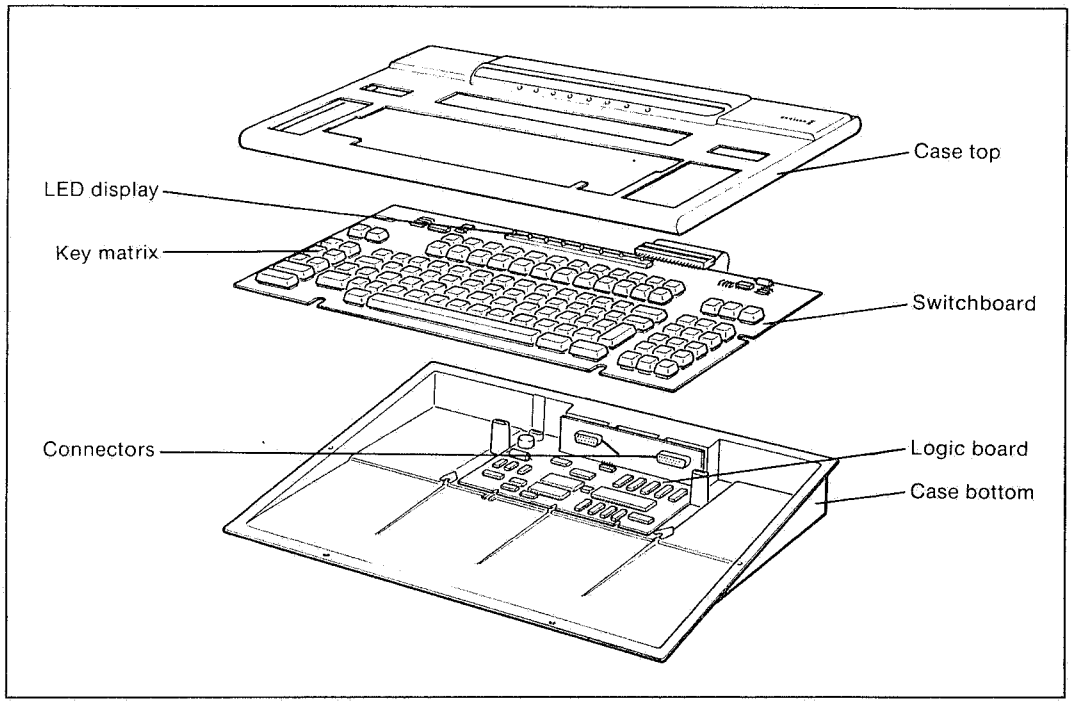


Fig. 1. Mechanical layout of KBU

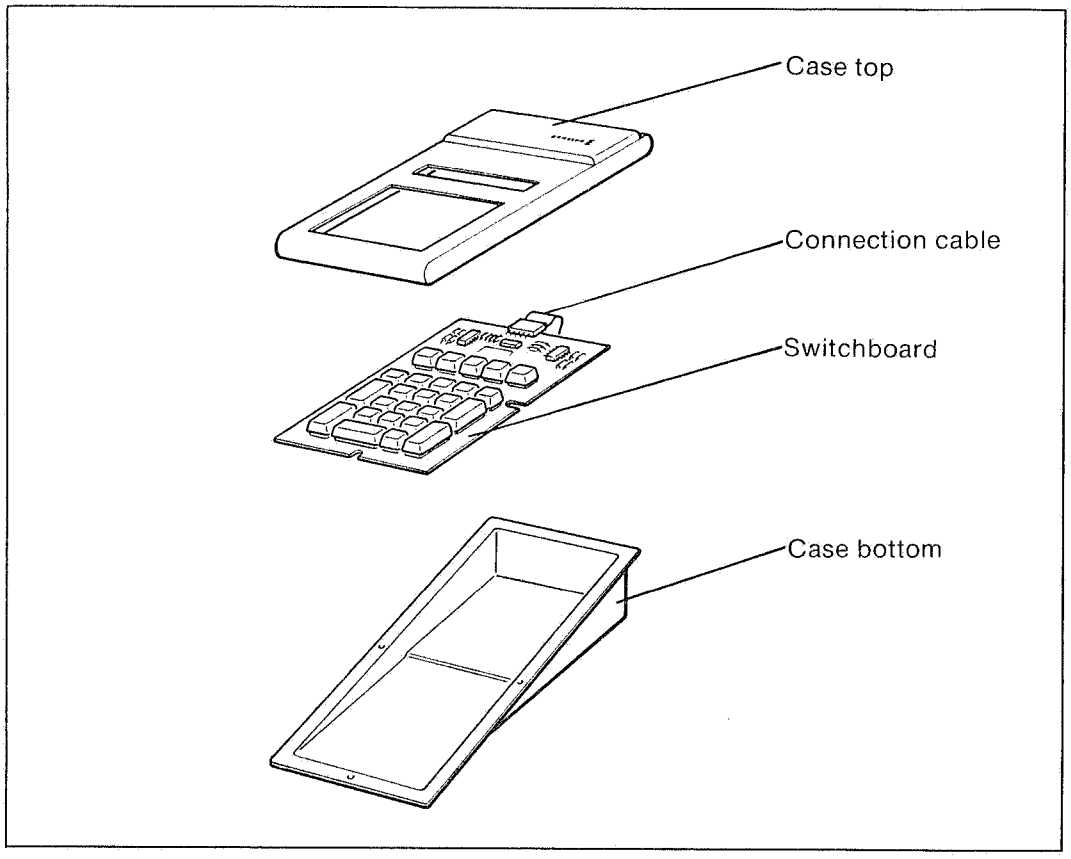


Fig. 2. Mechanical layout of KXU

The MSR is a unit intended for installation on the top of the KBU, see Fig. 3. The MSR is provided with a cable with connector for connection to the KBU logic board.

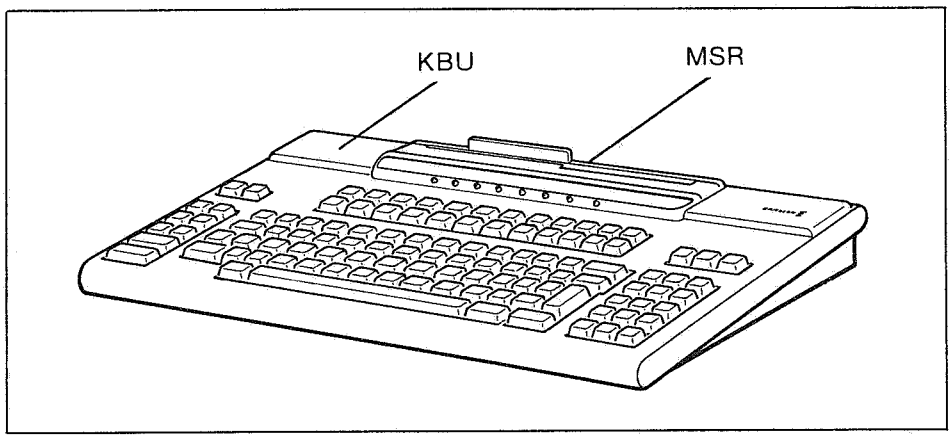


Fig. 3. Placement of MSR

Brief Outline

Hardware Functions

Main Parts

The main parts of the keyboard are, according to function:

- Microprocessor, CPT, with address decoder, ROM and RWM.
- Key matrices of KBU and KXU with decoding logic and input gates.
- LED display and loudspeaker.
- MSR with logic.

See block diagram, Appendix 5.

The CPT has two outputs for address and data signals. One of the outputs is an address output, while the other output is a combined address output and data input/output. An address register captures the address signals to separate these from the data signals.

The most significant bits of the address are fed to an address decoder which feeds enabling signals to:

- Output gates.
- ROM.
- RWM.
- Battery sense circuits.

The least significant address bits are fed to the ROM and the RWM to point out the memory cells. The LED display and the loudspeaker are controlled by means of the output gates. Signals from these gates also control the keyboard scanning, input gates and an analog section of the logic.

The scanning of the keyboard matrices is controlled by two signals from the output gates and a keyboard column activator. Information regarding pressed keys is fed to the CPT inputs via input gates, controlled by a signal from the output gates.

The MSR signals are fed to the CPT via the analog section. If an external MSR is used (not used in S41), its signals are also fed to the CPT the same way. A battery sense circuit is also connected to this analog section. The analog section, which is controlled by the output gates, evaluates the analog signals and converts them into digital levels. The analog section output signals are fed to the CPT interrupt inputs.

Interface

The keyboard is connected to the display unit via three signals:

- SVK, on/off signal to the display unit controlled by the keyboard on/off switch (not used in S41)
- KB-RESET, signal from the display unit for reset of the CPT and associated logic
- KB-DATA, asynchronous bidirectional data line between the KBU and the display unit.

Alternatively, a synchronous three line-connection can be used (not used in S41). In this case the KB-RESET and KB-DATA signals are replaced by the signals:

- DO, data from the display unit to the KBU.
- DI, data from the KBU to the display unit.
- CPKB, clock signal from the display unit to the KBU.

Function

The CPT controls the function of the KBU according to a program in the ROM. The CPT fetches an instruction from the ROM, executes the instruction, fetches the next instruction etc.

Addressing of memories and output gates takes place in two steps. First, part of the address is fed out via the AD output of the CPT. This part is saved in the address register. The rest of the address is then fed out via the A output of the CPT. The AD output/input is then used as data input/output from or to the circuit pointed out by the address decoder.

The sensing of the keyboard matrix is achieved by means of a shift register (column activator) activating one key column at a time. Each key connects, when pressed, one column to one row in the matrix. The CPT senses the keyboard rows via the input gates and evaluates which key is pressed from the column/row combination. The key codes are stored in a buffer area in the RWM while the KBU is waiting for transmission order from the display unit.

The MSR signal is amplified, filtered and converted into digital levels in the analog section. From the analog section interrupt signals are fed to the CPT. The CPT evaluates the received information and stores the MSR code in a buffer area in the RWM.

Data output is made via the output gates. The output signals control:

- Keyboard matrix scanning.
- Input gates.
- LED display.
- Loudspeaker.
- Analog section.

The loudspeaker control circuit consists of a three-bit digital-to-analog converter with amplifier. The program performs frequency as well as amplitude control of the acoustic signal. Two types of signals are used:

- Click signal, which is the acoustic feedback on a pressed key.
- Alarm signal, which is an alert signal to the operator.

Power Supply

The keyboard is fed with 5 V DC from the display unit. A battery is provided to keep the RWM powered when the 5 V supply is switched off. The battery voltage is sensed by a supervision circuit in the analog section. In case of low battery voltage, an interrupt signal is fed to the CPT.

Firmware Program Functions

See Fig. 4.

Power On

At power on, a test sequence is executed. The sequence consists of:

- ROM test.
- RWM test.
- CPT test.
- Battery voltage test.

If the test sequence results in an error, an error message is shown on the LED display. If the ROM-, the RWM- or the CPT test results in an error, the keyboard function is blocked. The LEDs and the alarm are also activated during the test sequence.

If no blocking errors are found, an initiation procedure is executed. The procedure sets the circuits in their initial states.

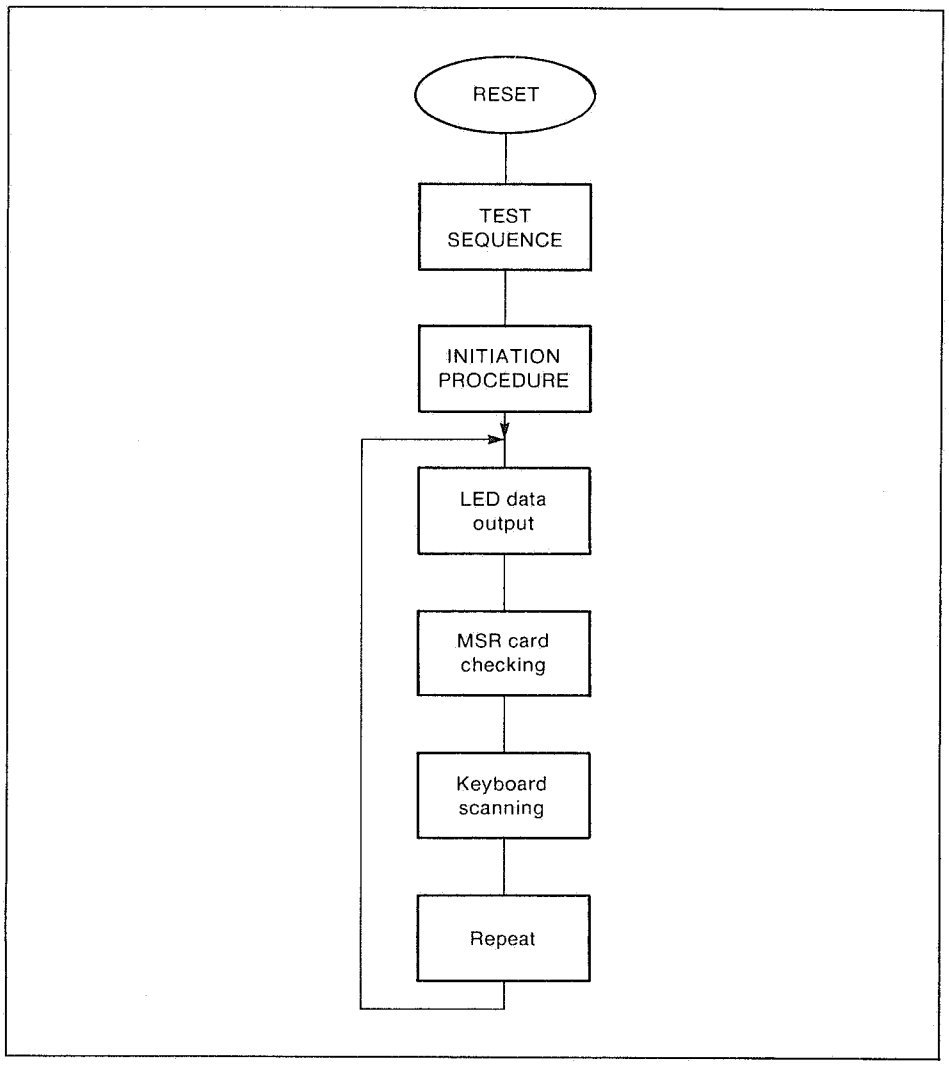


Fig. 4. Summary of program sequence

Main Sequence

When the tests and the initiation have been made, the keyboard is ready to receive orders from the display unit. When an order is received where the "stop transmitting and scanning" bit is not set, the program enters the main sequence. This part of the program performs the following functions:

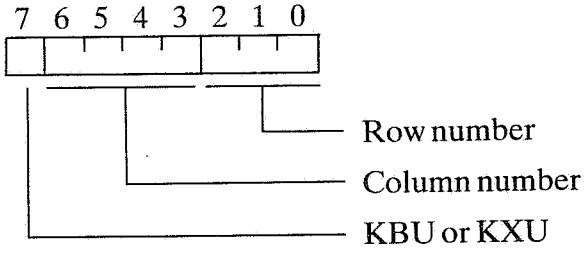
- Lights the LEDs on the panel according to LED data from the display unit.
- Checks whether or not a card is inserted in the MSR.
- Scans all the keys of the keyboard.
- Decides if the code for a pressed key should be transferred repeatingly to the display unit.
- Handles two interrupts. One appears at each bit fed from the MSR. The other one appears at every byte fed from the display unit.

The main loop begins with the microprocessor addressing the RWM cell containing current LED data. Data is applied to the LED driving register via the data bus.

When LED data is set, the program proceeds with testing the signal COSI from the MSR. This signal tells whether there is a card in the reader or not. This part of the program checks if the signal has changed state since the previous scan. If so, the display unit is informed of the change. The keyboard matrix, defined by 16 columns and 8 rows, is scanned every main loop in the following manner. The program sets a zero in the first stage of the column activating shift register. Then the program enables the eight input gates, whereby the state of the eight activated switches are applied to the CPT. The column activating shift register is clocked by the program, the second column of the keyboard is enabled and the state of this is fetched in the same manner as the first. The zero is stepped again to indicate the third column and so on until all 16 columns are fetched and stored in the buffer.

The program then starts to compare the 16 bytes, one by one, with the corresponding bytes from the previous scanning. Should a difference appear at the comparison, this means that a key has been depressed, or that a depressed key has been released. An additional scan is used to filter out contact bounces. If a key has been depressed, the program determines the number of the key consisting of column number (bits 6 through 3) and row number (bits 2 through 0).

Bit 7 in the code states whether the code comes from the keyboard or the expansion keyboard, see the figure below.

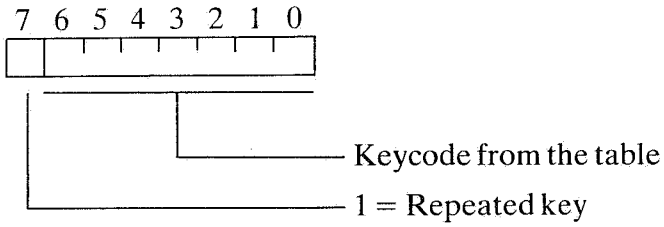


The column/row numbers and the keyboard codes for the keys are shown in Appendix 2.

The codes are translated according to certain tables before they are fed to the output buffer. Two tables are used, one for normal codes (Appendix 3) and one for extended codes (Appendix 4). When extended codes are used, the status byte states, whether the code comes from the keyboard or from the expansion keyboard. The keycode is fed to the output buffer together with the status byte at hand, thus each keycode occupies two bytes in the buffer.

The output buffer holds 16 bytes, which means that 8 key numbers can be stored. The matrix is scanned at least every 2 ms. When the program has finished the matrix scanning, it proceeds with the repeat function. First, the strapping bytes are checked to find if any key is assigned as repeat key. If so, the program checks if this key is depressed. If it is depressed and the output buffer is empty, the number of the last depressed key is set into the buffer once again. This is made until the repeat key is released. If no key is assigned as repetition key, the "typematic" function is valid. This means that when any key on the keyboard is depressed longer than 0,5 s its number will be set into the buffer as described above. The repetition ceases when the key is released.

When repetition occurs, bit 7 in the keycode is set, see the figure below.



Communication

Communication is handled by the interrupt routine according to Fig. 5. The display unit sends messages formed by order bytes and data bytes. The KBU answers with status bytes and data bytes.

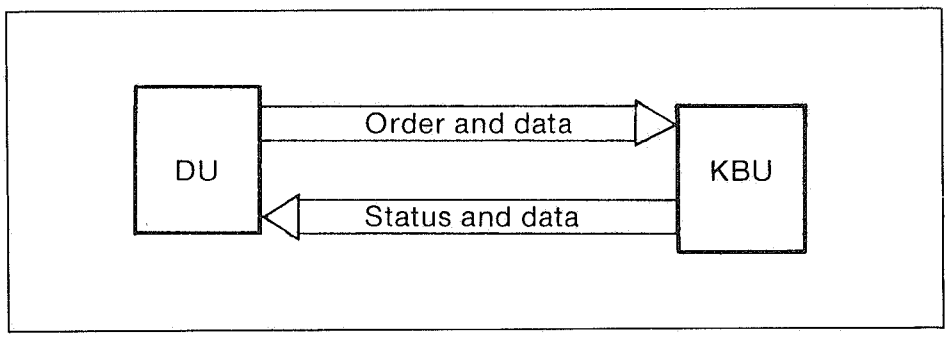


Fig. 5. DU – KBU transfers

Each byte is built up according to Fig. 6.

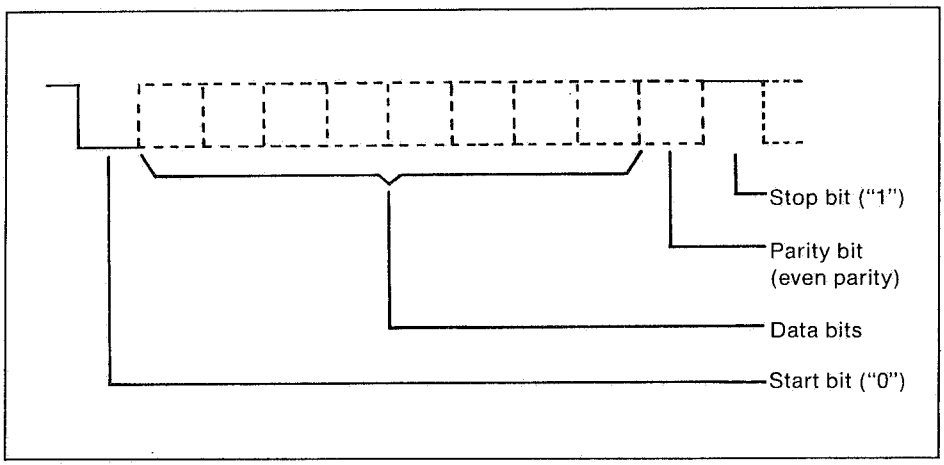


Fig. 6. Word format of DU – KBU transfers

Communication Procedures

Display Unit Messages

Message Format

The five different types of messages sent from the display unit are shown in Fig. 7. Communication examples are found in Appendix 6.

The first sent data bit in a byte is defined as bit 0 and the last one as bit 7.

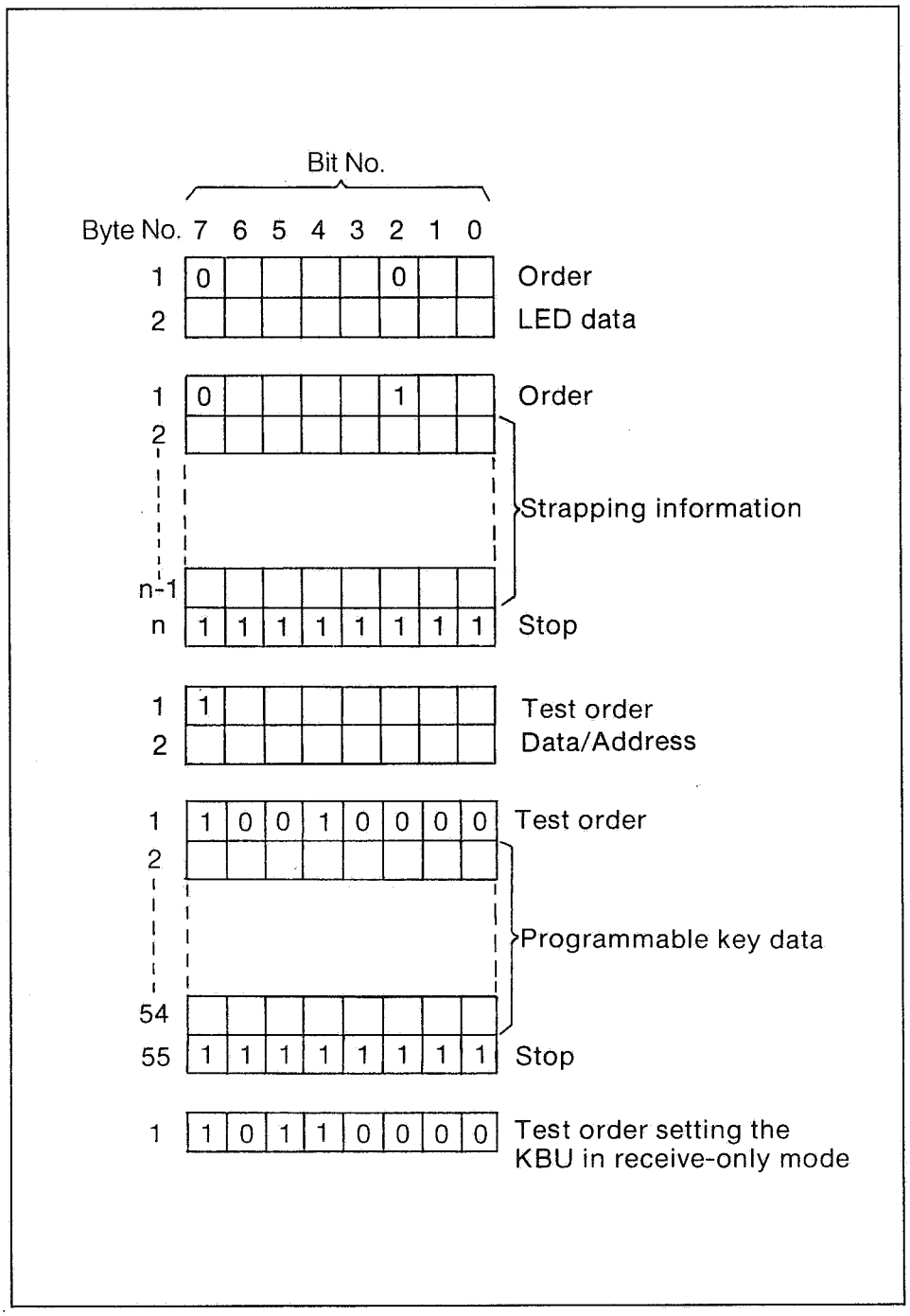
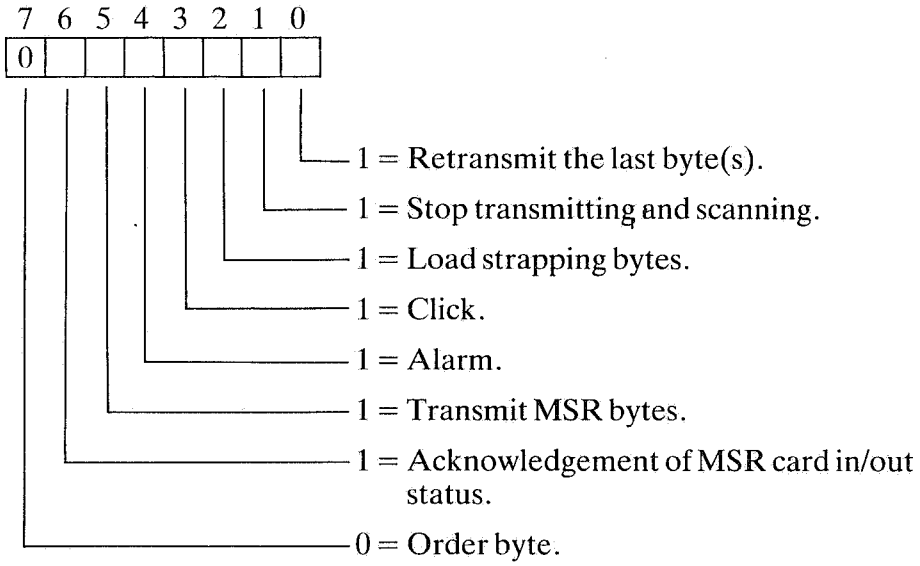


Fig. 7. DU to KBU message formats

Order byte

The first byte is defined as the order byte transferred to the keyboard. Bit 7 of the byte defines if it is an order byte or a test order byte. If bit 7 = 0 the byte contains the following information:



The bits cause, with the values stated above, the following functions:

Bit 7 defines order byte.

Bit 6 implies an acknowledgement that MSR status (card in/card out) has been received. The keyboard keeps sending MSR status until this bit is received.

Bit 5 orders the keyboard to transfer MSR data which is stored in an RWM area.

Bit 4 orders the keyboard to activate an alarm via the loudspeaker.

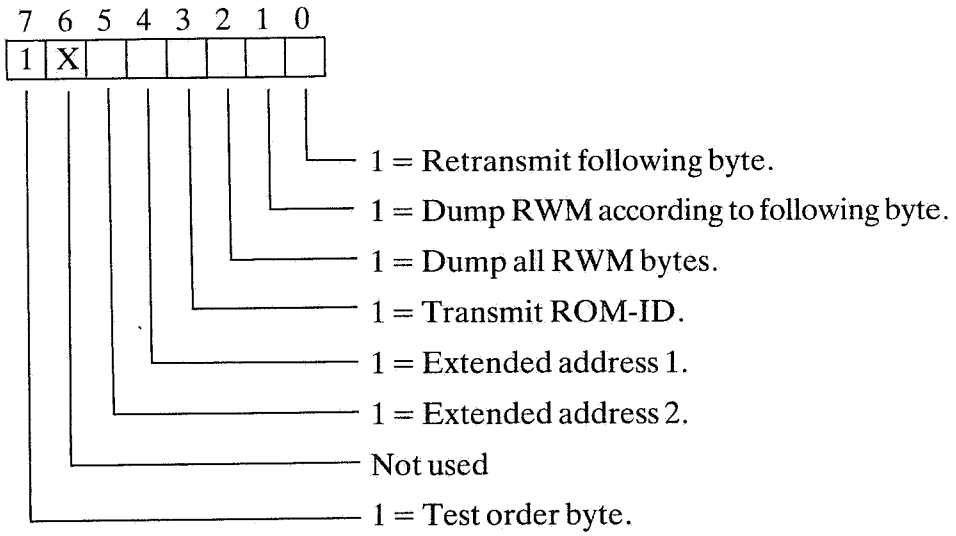
Bit 3 orders the keyboard to induce a click sound via the loudspeaker, e.g. as feedback from a key depression. The keyboard can be assigned to click by itself at the detection of a depressed key, see "Strapping Bytes".

Bit 2 indicates that the subsequent byte(s) contain(s) strapping data, see "Strapping Bytes". If bit 2 = 0 the subsequent byte consists of LED data.

Bit 1 The order byte from the display unit is always considered as a poll, upon which a status byte should be answered. However, when bit 1 is set transmission of status and data is stopped as well as scanning of the key matrix.

Bit 0 orders the keyboard to retransmit the previously transmitted data byte(s), e.g. if some kind of fault has been detected by the display unit.

If bit 7 = 1 the keyboard is set into test mode and then the order byte contains the following information:



The bits cause, with the values stated above, the following functions:

Bit 7 Test mode is defined by order byte bit 7 = 1. Depending on the states of bit 0 and 1, the subsequent byte is either:

- A byte to be echoed.
- An address to the RWM.
- Ignored.

Bit 6 Not used.

Bit 5, 4 These bits are used to address devices which may be connected to the communication link between the display unit and the keyboard. The bits are used for loading of programmable key data and to order IPL of the keyboard.

Bit 5	Bit 4	Function
0	0	Address 1
1	0	Address 2
0	1	Programmable key data
1	1	IPL

Bit 3 All firmware (stored in ROM) has an identification sequence programmed in a certain area. When bit 3 is set, this area is addressed by the microprocessor and valid ROM ID data is sent to the display unit. This makes it possible to automatically choose the keyboard table corresponding to the keyboard program. The subsequent byte is ignored.

Bit 2 means that the contents of the whole RWM area will be sent to the display unit and thus that the subsequent byte should be ignored.

Bit 1 implies that the subsequent byte should be considered as an address to the RWM area and that the contents of this memory cell will be transferred to the display unit.

Bit 0 indicates that the subsequent byte from the display unit shall be "echoed" back by the keyboard logic so that the display unit can check that no transmission faults have occurred.

Programmable Key Data

Programmable key data is sent in a block formed by 55 bytes.

Byte 1 Order byte ($90_{(16)}$).

Byte 2 Key value for the PF key in question.

Byte 3–54 Key value and status for the keycodes in the string (2 bytes each).

Byte 55 Stop code ($FF_{(16)}$).

IPL

The IPL order byte ($B0_{(16)}$) sets the keyboard to receive-only mode with the transmission speed 4 800 bits per second to receive program code.

Data Byte

When the order byte bit 7 = 0 and bit 2 = 0, the subsequent data byte is considered as LED data and the LEDs are lit accordingly. A bit position set to one corresponds to a lit LED.

The meaning of the LEDs is software-dependent. Bit 7 corresponds to the leftmost LED position on the keyboard panel and bit 0 to the rightmost position.

Strapping Bytes

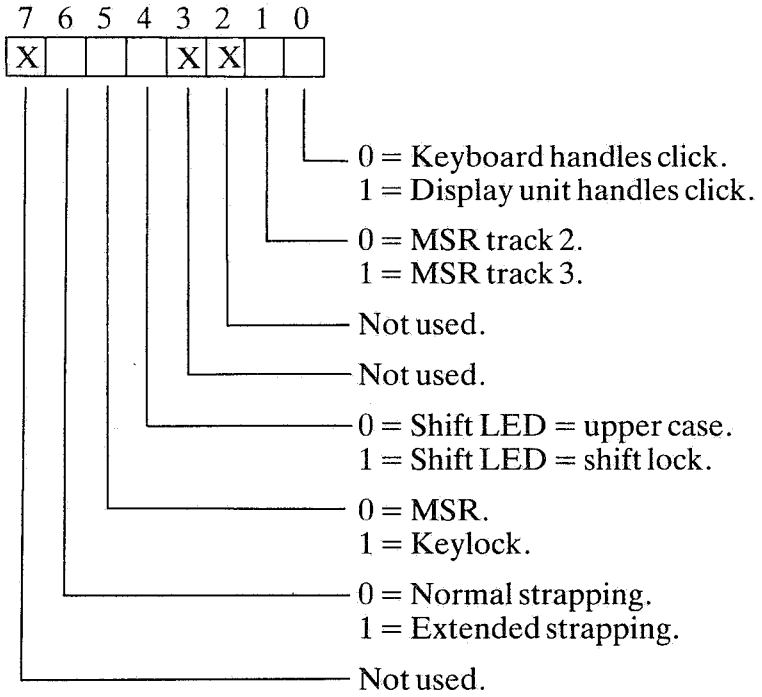
In order to avoid hardware strappings for altering certain functions in the keyboard, "software strappings" are used.

These strapping bytes are fed from the display unit program mainly after a system reset.

The strapping bytes are stored in the microprocessor RWM area.

When the order byte bit 7 = 0 and bit 2 = 1 up to 17 strapping bytes can be transferred from the display unit to the keyboard logic. This string of strapping bytes ends with a stop byte with the code $FF_{(16)}$. Thus no strapping byte may use the code $FF_{(16)}$.

Strapping byte 1



The bits are further explained below:

Bit 7 Not used.

Bit 6 indicates if normal strapping (9 bytes) or extended strapping (17 bytes) is used.

Bit 5 indicates if a magnetic stripe reader or a keylock is used.

Bit 4 decides if the shift LED shall indicate upper case or shift lock.

Bit 3 Not used.

Bit 2 Not used.

Bit 1 indicates if MSR track 2 or 3 is to be selected.

Bit 0 decides if the display unit alone shall handle the click function or if the keyboard shall handle the feedback click function at the depression (not at repetition) of a key.

Strapping byte 2

Defines which LED is to be used as shift LED. If this byte is $00_{(16)}$ no LED is handled by the KBU program. If bit 7 = 1 the leftmost LED functions as shift LED etc.

Strapping bytes 3 – 16

Strapping bytes 3 – 16 indicate key numbers, i.e. key positions on the keyboard, which are assigned to certain functions such as shift lock See. Appendix 2. Thus it is possible to assign any key in the matrix to such a function.

Strapping byte 13

Strapping byte 13 indicates the key number for the repeat key. If this is used, the "typematic" function is omitted, i.e. the automatic repeat function that is normally activated when any key is depressed more than 0.5 seconds. The repetition frequency is determined by the display unit program.

Note that the keyboard only notes the keynumber of the various shift and repeat functions. The measures to be taken when any of these keys is depressed are decided by the display unit program.

Normal Strapping

Byte	Function
3	Not used
4	Not used
5	Not used
6	Shift 3
7	Shift 4
8	Repeat key
9	SEMAC key

Strapping bytes 3 – 5 are not used, as shift lock 1, shift 1 and shift 2 are assigned by the keyboard program. Shift 1 always releases the shift lock function and is used as an ordinary upper case/lower case shift key.

Extended Strapping

Byte	Function
3	Shift lock 1
4	Shift 1a
5	Shift 2
6	Shift 3
7	Shift 4
8	Repeat key
9	SEMAC key
10	Shift 1b
11	Shift 1c
12	Shift lock 2
13	Shift lock 3
14	Shift lock 4
15	Alarm volume
16	Alarm tone
17	Click volume

Shift lock 1 is reset by shift 1a – c and shift lock 2 – 4 is reset by shift 2 – 4. $80_{(16)}$ in a strapping byte means "not used".

KBU messages

Message format

The four different types of messages sent from the KBU are shown in Fig. 8. Communication examples are found in Appendix 6.

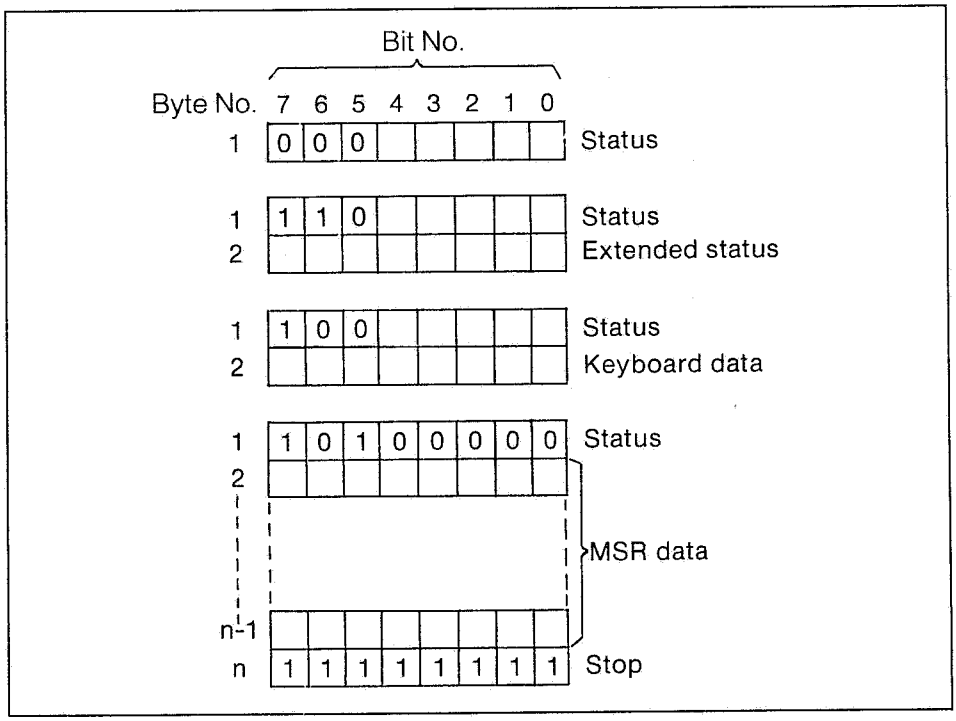
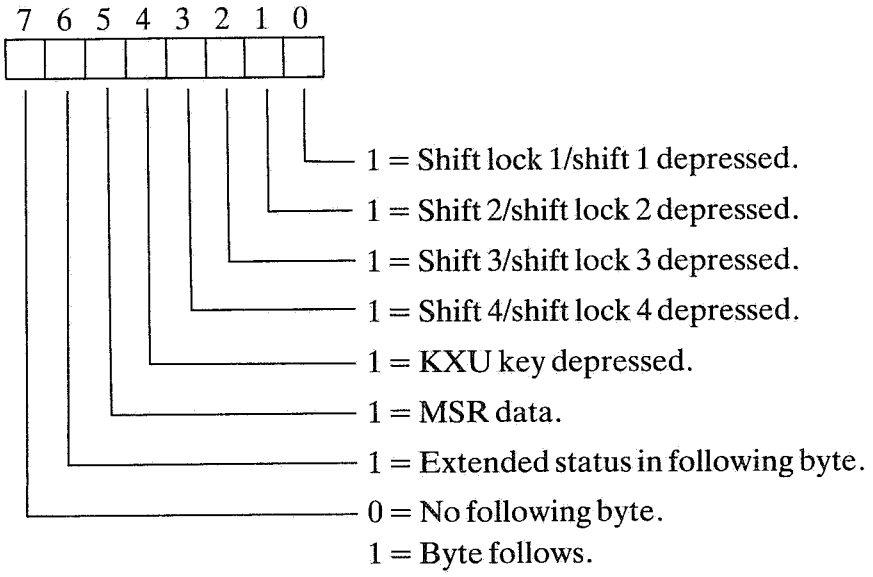


Fig. 8. KBU to DU message formats

Status byte

When order and data (or strapping data) have been received from the display unit, the keyboard replies with status and data unless bit 1 (stop transmitting and scanning) was set in the received order byte. The status byte has the following format:



The bits are further explained below:

Bit 7 0 implies that no data follows in this transfer.

1 implies that key-number, MSR data or extended status follow the status byte.

Bit 6 indicates that an extended status byte follows the status byte.

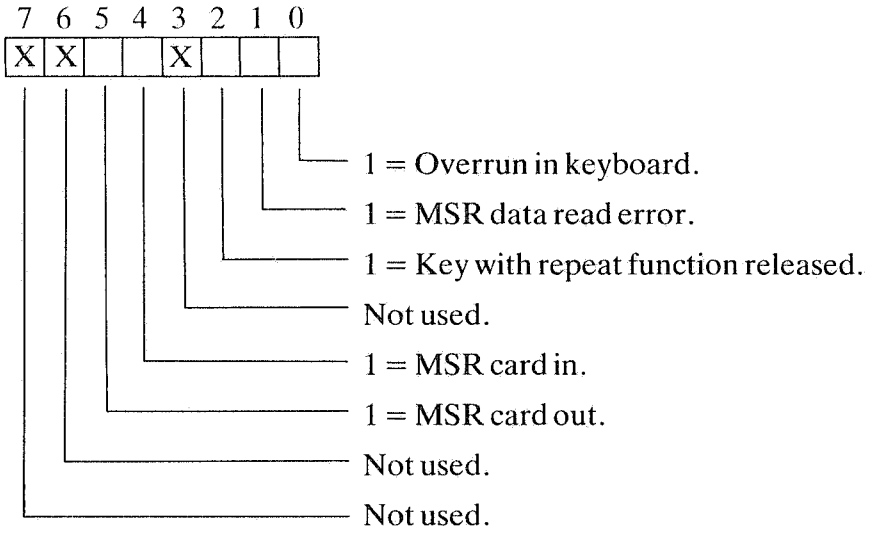
Bit 5 indicates that MSR data follows the status byte.

Bit 4 indicates that a KXU key with extended code function has been depressed.

Bits 3 – 0 indicate that keys assigned to shift or shift lock have been depressed.

Extended status

When bits 7 and 6 = 1 and bits 5 – 0 = 0 in the status byte, an extended status byte, as below, follows:



The bits are further explained below:

Bits 7, 6 Not used.

Bit 5 is set when the MSR card has been taken out of the MSR or if a fault is detected during reading (see bit 1). Also reset at acknowledge. See below.

Bit 4 is set if a card is inserted in the MSR and the bytes have been correctly read by the KBU. This bit is reset after that the KBU has received an acknowledge of MSR card in/out status from the display unit.

Bit 3 Not used.

Bit 2 indicates that a key with "typematic" repeat function or the repeat key (see "strapping byte 8") has been released.

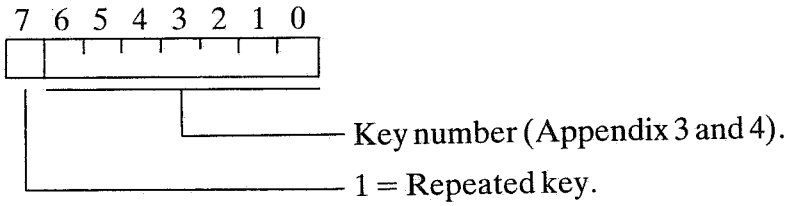
Bit 1 indicates that the keyboard has detected a parity fault or a fault at the redundancy check while reading the MSR data. Bit 5 is also set.

Bit 0 indicates that the keyboard has lost one or more key depressions due to full output buffer. This buffer can hold 8 key numbers before overrun.

Note that the keyboard transfers extended status when bit 4 or 5 are set until the display unit acknowledges with bit 6 set in the order byte.

Data bytes

When the status byte bit 7 = 1 and bits 6 and 5 = 0 a data byte containing the number of a depressed key is transferred after the status byte:



When bits 7 and 5 = 1 and the rest = 0 in the status byte, data from the MSR follows the status byte.

Detailed Description

This section only describes more complicated circuits and components. Please refer to the block diagram in Appendix 5 when reading this text.

Keyboard Decoding

The shift registers forming the column activators are controlled by the SHIFT COLUMN and SCANNING signals according to the timing diagram, Fig. 9. A logic 1 signal is applied to one column at a time. If a key is pressed, the corresponding ROW signal is active. The ROW signals are input to the CPT via the input gates when the SEL-EXT-Y signal is active.

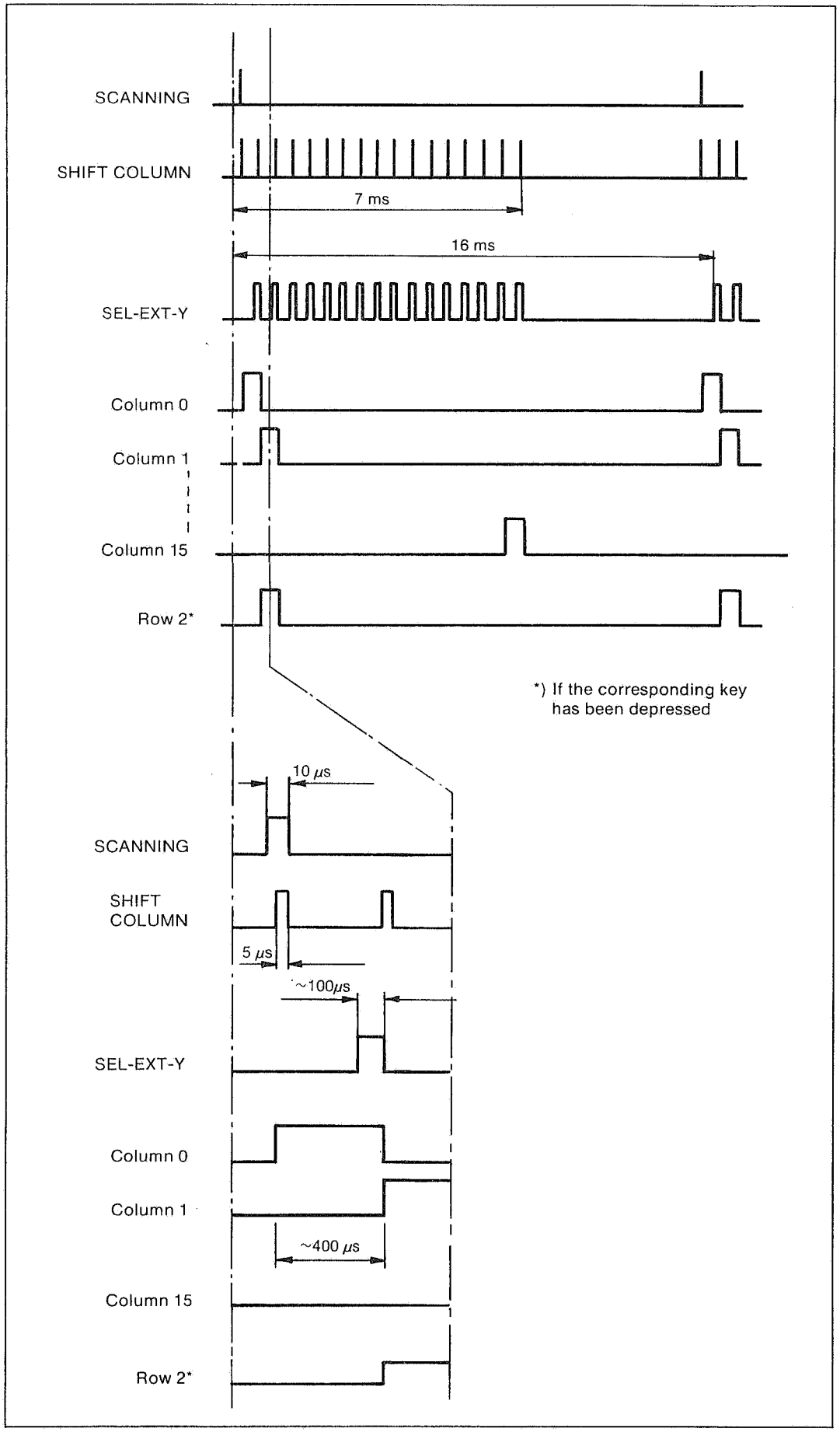


Fig. 9. Timing of key scanning

Address Decoding

The used addresses are shown in the table below. Address decoding for ROM, battery supervision and output gates is made via an address decoder (IC9).

Address (hex)	Function
0000–07FF	RWM (IC11) enable
1000–1FFF	ROM (IC10) enable
4XXX	Battery supervision enable
8XXX	Battery supervision disable Output gate (IC4) enable
CXXX	Output gate (IC3) enable

The output gates are used in the following way:

Output gate IC3		Output gate IC4	
Bit	Used for	Bit	Used for
0	LED 0	0	} MSR track select
1	LED 1	1	
2	LED 2	2	Keyboard select
3	LED 3	3	SCANNING
4	LED 4	4	SHIFT COLUMN
5	LED 5	5	
6	LED 6	6	} Loudspeaker control
7	LED 7	7	

The RWM is enabled when an S 180 signal is at logic 0 level. The S 180 signal is controlled by the address A11 signal via the transistor Q4, see Fig. 10.

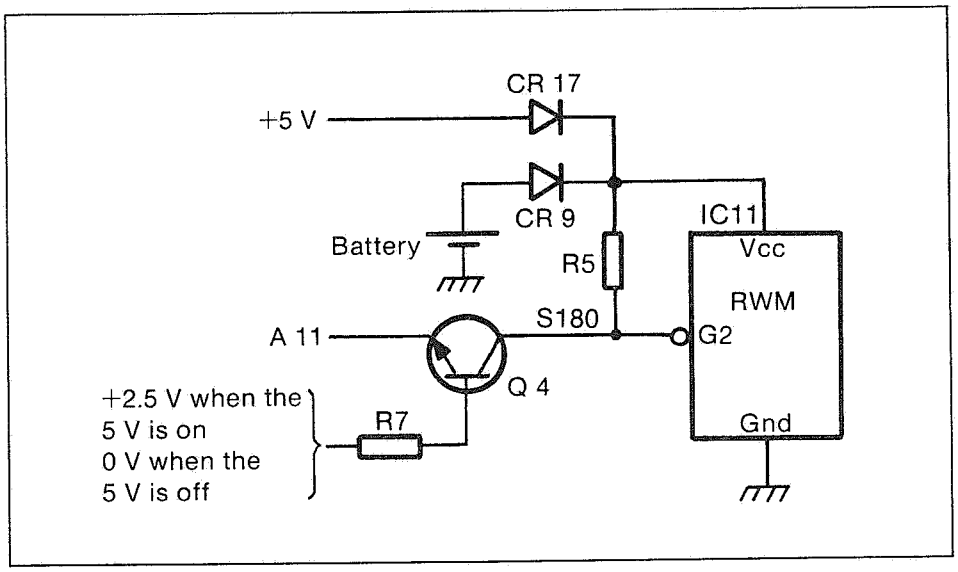


Fig. 10. RWM power and enable

RWM Power Supply

The RWM power supply and enable signal (G2) control are shown in Fig. 10. When the 5 V power supply is on, the RWM is powered via the diode CR17. When the 5 V power supply is off, the RWM is instead powered by the battery via the diode CR9. The transistor Q4 is cut-off, when the 5 V supply is off, to keep the RWM-enable signal deactivated.

Battery supervision is made in the battery sense and analog sections, see Fig. 11. When the flip-flop, formed by the IC20:C and IC20:D gates, is set current flows via the diode CR14 to the battery and via the diodes CR15 and CR16, the resistor R44 and the flip-flop to ground. The voltage at the CR15/CR16 junction is 0.2 V lower than the battery voltage. The level sensing circuit, formed by the amplifier IC15, outputs a logic 1 level (3.2 V) if the battery voltage is lower than 2.7 V.

The reference voltage for the supervision circuit has a level of 2.5 V. The sensing circuit is switched off when the flip-flop is reset.

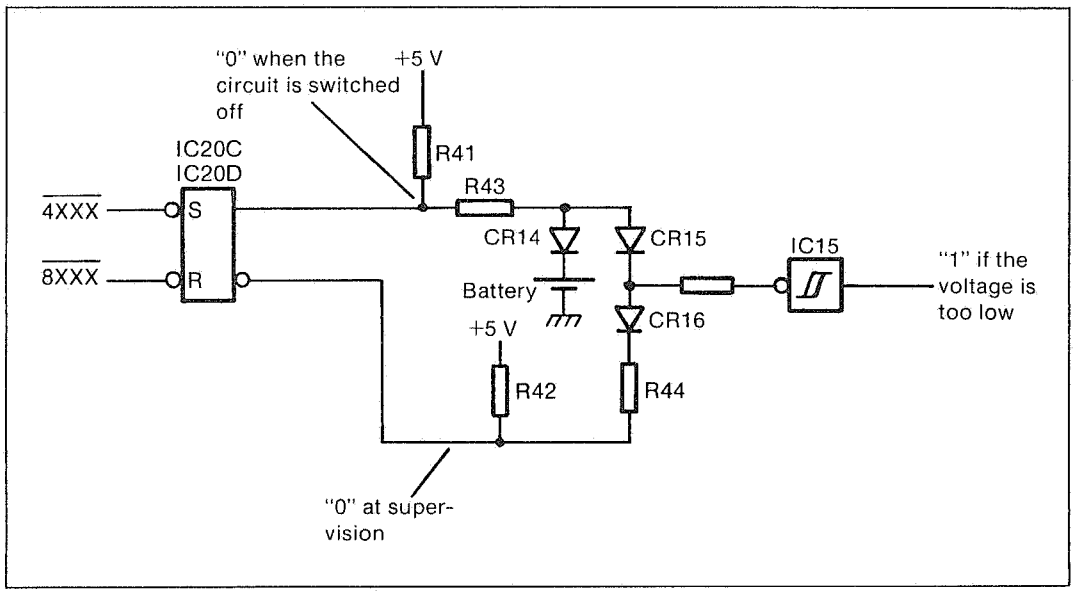


Fig. 11. Battery supervision

MSR Analog Circuit

The signal from the MSR is processed in three stages, see Figs 12 and 13. The first stage is the amplifier IC15:A-C which has a gain of 50. One amplifier per track is used. The selector IC17 connects the amplifier for the used track to the second stage. The second stage consists of C8, R45 and the amplifier IC15:D. The stage is, according to function:

- A derivating stage (C8/R45).
- A threshold detector (R32/R35).
- An LP filter (R33/R45/C13).
- An amplitude limiter (CR4-CR7).

The third stage consists of the gate IC14 which feeds the CPT interrupt input. Each flank in the output signal from IC15:D induces a negative flank to the interrupt input. The CPT acknowledges the interrupt by changing the state of the INTERRUPT ACKNOWLEDGE signal, thereby resetting the interrupt signal.

Signals from an external MSR are fed to the other CPT interrupt input in a similar manner.

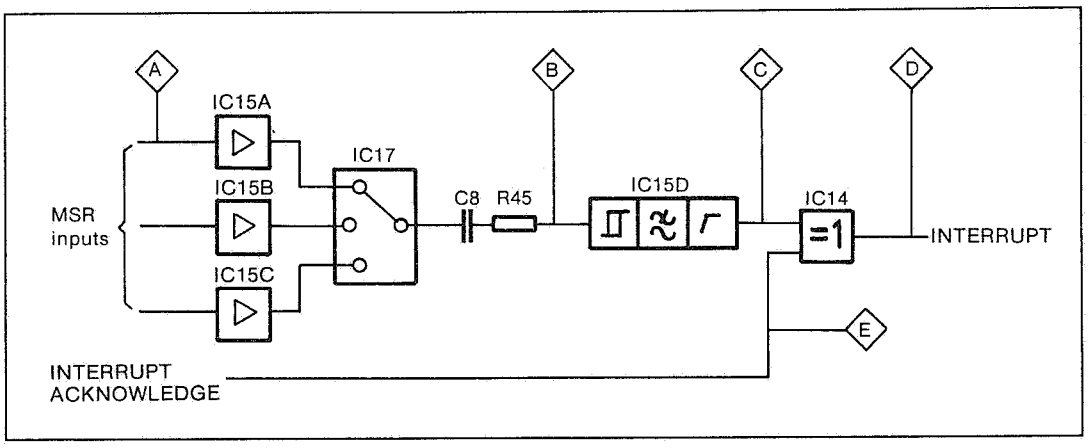


Fig. 12. MSR interface

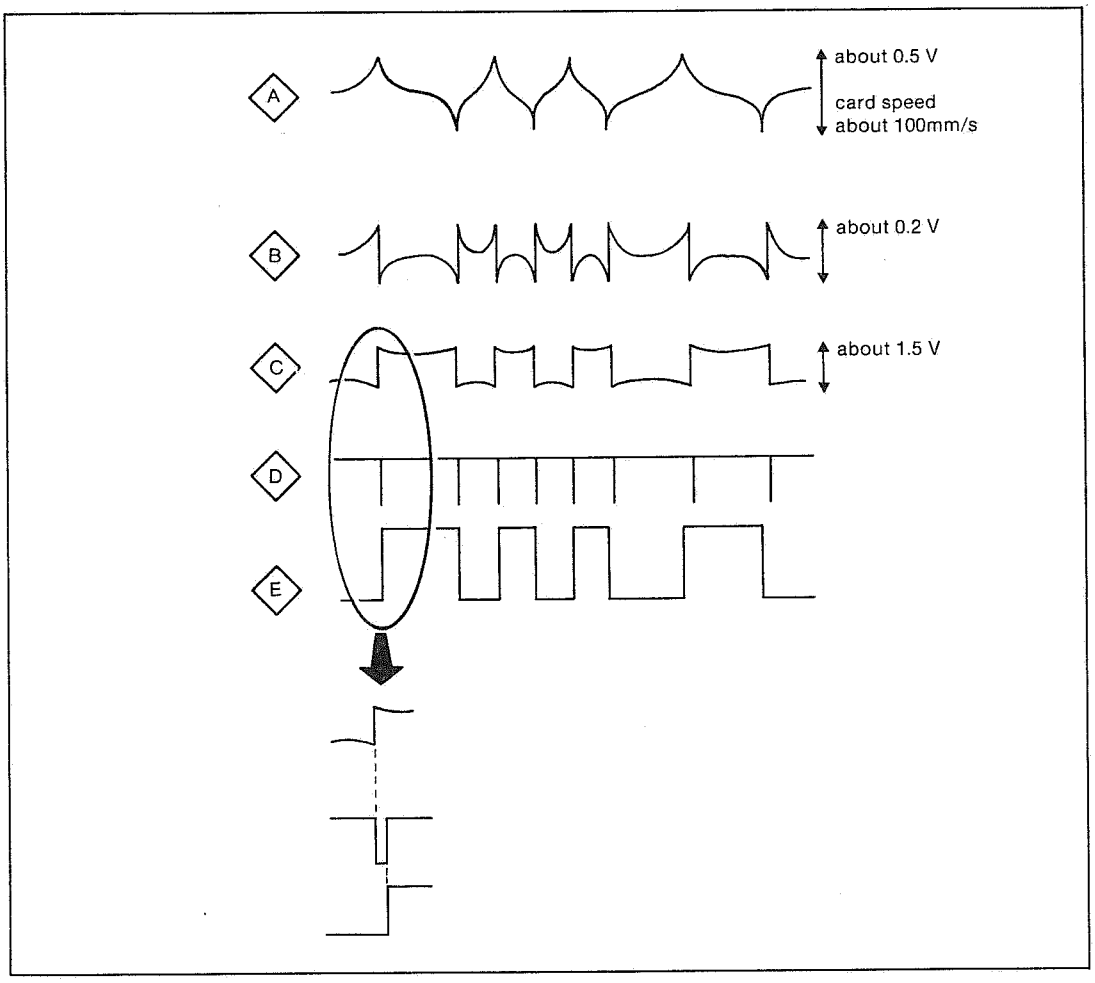


Fig. 13. MSR interface timing

Hardware Strappings

Please refer to the keyboard adapter circuit diagrams. If a CPT with built-in ROM is used (not used in S41), the strapping W10 is cut-off. In this case, IC10 can be changed to a RWM if the strappings W9, W13 and W20 are cut off while W1 and W21 are strapped.

The interface signal designations can be altered using strappings. When the bidirectional data line KB-DATA is used, the W3 strap is cut off and W4 is strapped. The KB-DATA signal is connected to the logic via the W2 strap. If two lines (DI and DO) are used, the strappings W2 and W4 are cut off. The interface signal can be inverted by changing the W6/W11 strapping for the signal to the keyboard and the strappings W5/W12 for the signal from the keyboard. The normal strappings for S41 are shown in the diagrams.

Microcomputer

The microcomputer used is called CPT 8051. It combines CPU, memory, and serial as well as parallel I/O-ports. The microcomputer is also available in a version with built-in program memory (ROM). The oscillator (at XTAL inputs) is running at 12 MHz and the cycle time is thus 83.33 ns.

The CPT has four 8-bit I/O ports. When external memories are used, three of the ports are used as address, data and control bus. See Fig. 14. The CPT connection pin functions are:

- P0/D/A: 8-bit I/O port or multiplexed address/data bus.
- P1: 8-bit I/O port.
- P2: 8-bit I/O port or address bus.
- P3: 8-bit I/O port or control bus.
- RST: Reset input.
- ALE: Strobe for output of the multiplexed address via port 0.
- PSEN: Output for enabling of the external program memory.
- EA: Input for selection of internal (if applicable) or external ROM.
- XTAL: Connections for the oscillator crystal.

The control bus signals are:

- RXD/D: Serial port input data (asynchronous) or bi-directional data (synchronous).
- TXD/CL: Serial port output data (asynchronous) or clock output (synchronous).
- INT0/INT1: Interrupt input or gate control input for counter.

- T0/T1: Input to counter.
- WR: Strobe for output of data via P0 (write).
- RD: Strobe for input of data via P0 (read).

The following is a brief description of the microcomputer in this application. Even more detailed information can be found in the INTEL MCS-51 manual.

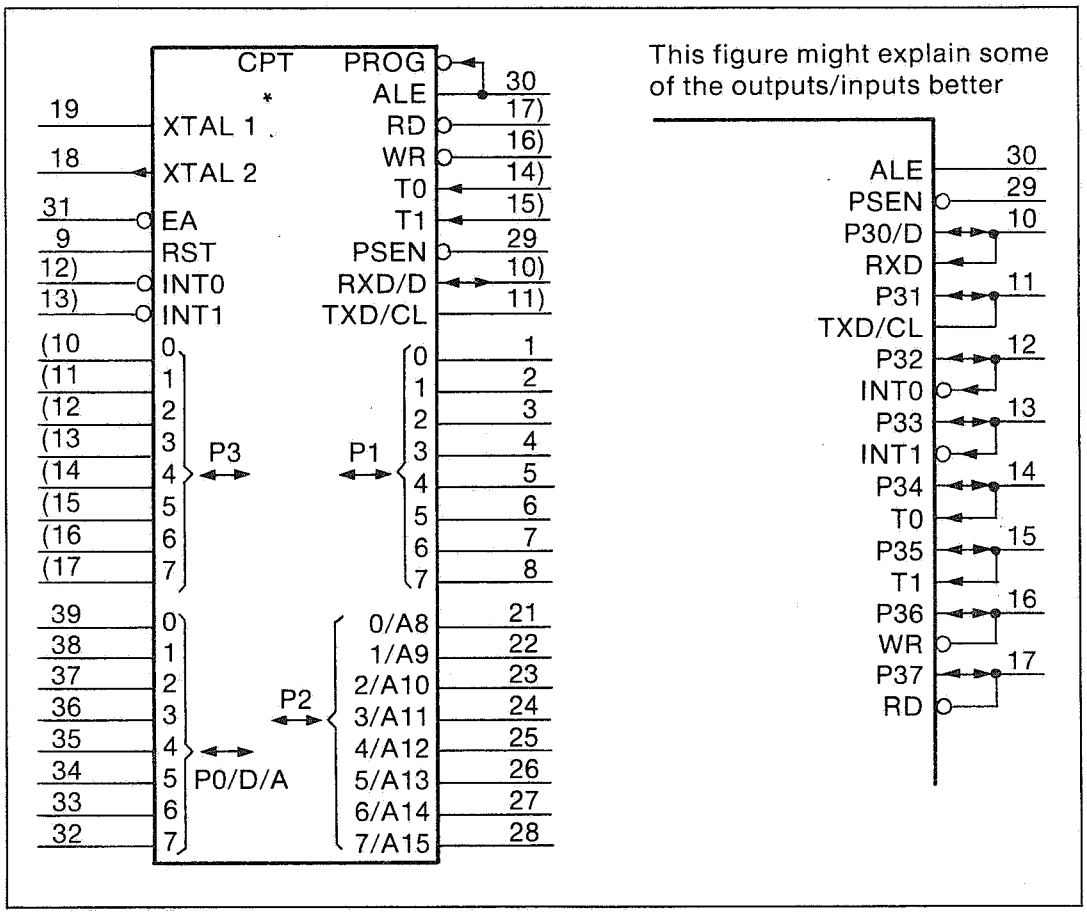


Fig. 14. CPT drawing symbol

Hardware Structure

The structure of the CPT is shown in Fig. 15. The main functional blocks are:

- CPU.
- Oscillator and timing circuit.
- 4k bytes ROM (optional).
- 128 bytes RWM.
- Two timers/counters.
- Serial I/O port (UART, Universal Asynchronous Receiver-Transmitter).
- Parallel I/O ports.

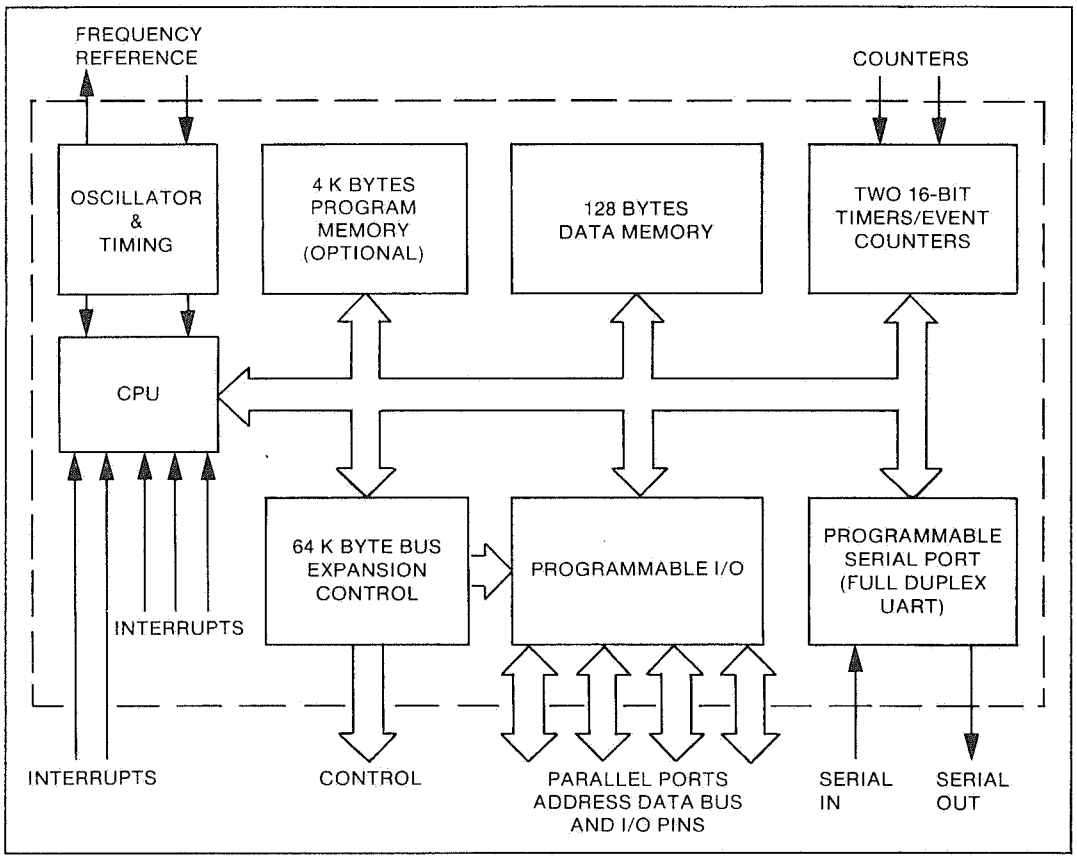


Fig. 15. CPT block diagram

The CPT can address up to 64k bytes of external program and/or data memory, see the timing diagram, Fig. 16.

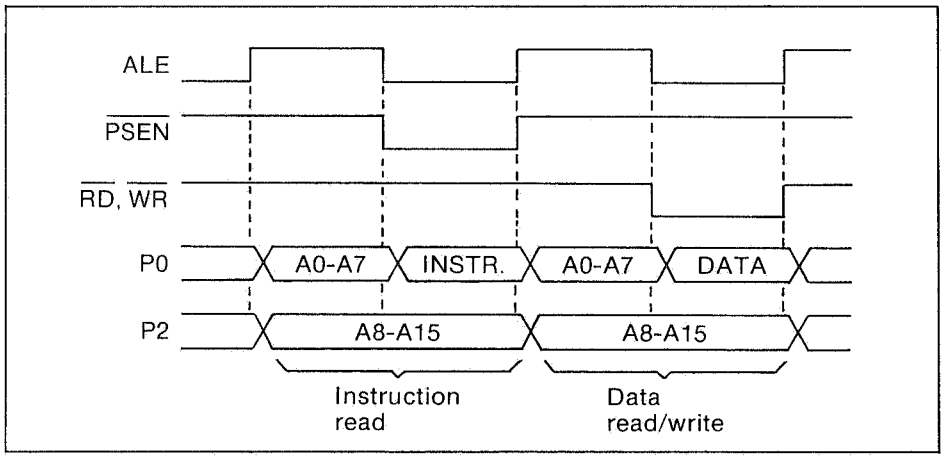


Fig. 16. CPT timing

The timers/counters can be used for measuring time intervals, measuring pulse widths, counting events and generating interrupts, see Fig. 17.

In this application one of the timers/counters is divided into two 8-bit timers. One part controls the keyboard scanning and the LED output while the other part controls the loudspeaker and the MSR input. In this case, the other timer/counter is used to control the UART transmission rate.

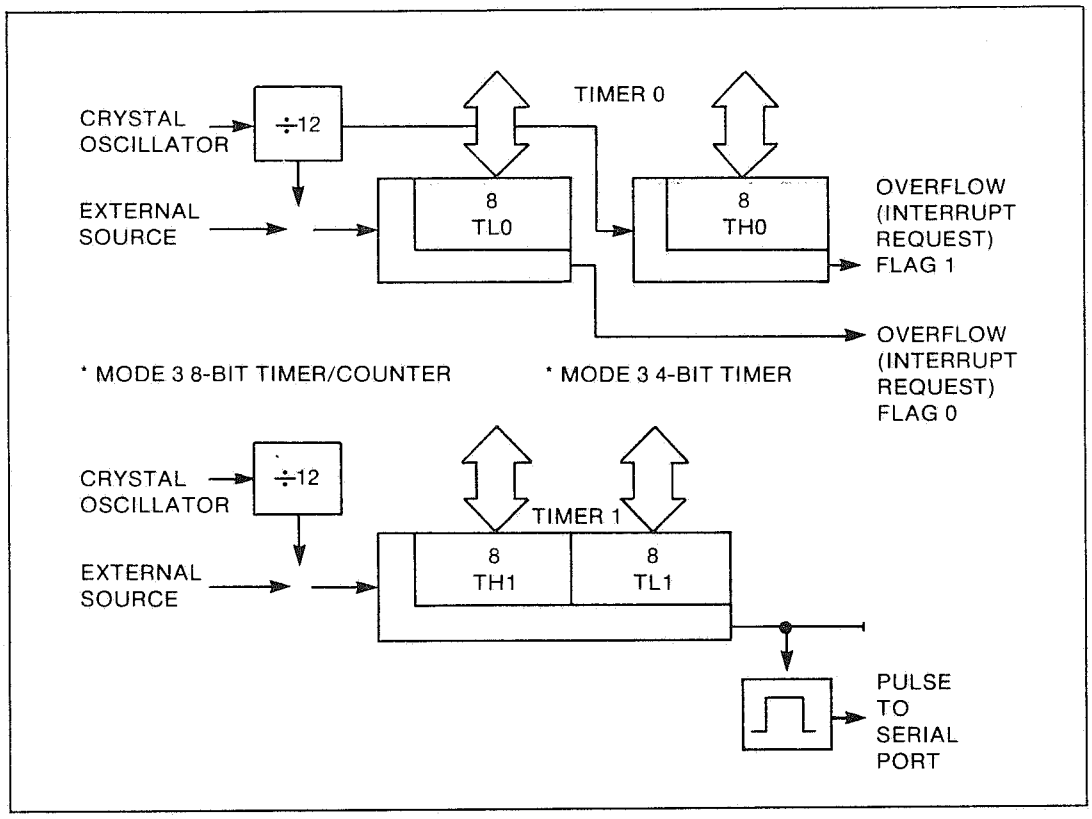


Fig. 17. CPT timers/counters

The serial port can work in synchronous mode or, as in this application, as a full-duplex asynchronous UART, see Fig. 18. The timing can be controlled from the oscillator via one of the timers. The data byte can contain 10 – 11 bits.

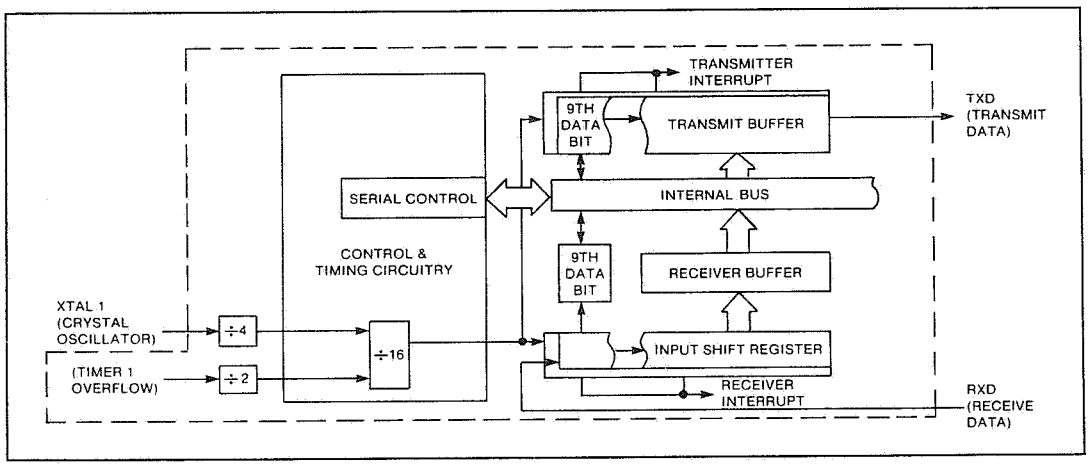


Fig. 18. CPT serial port

The interrupt system is a multiple source, two level system. The CPT acknowledges interrupts from five sources. Each of the five sources can be assigned to either of two levels and can be independently enabled. The interrupt system is shown in Fig. 19.

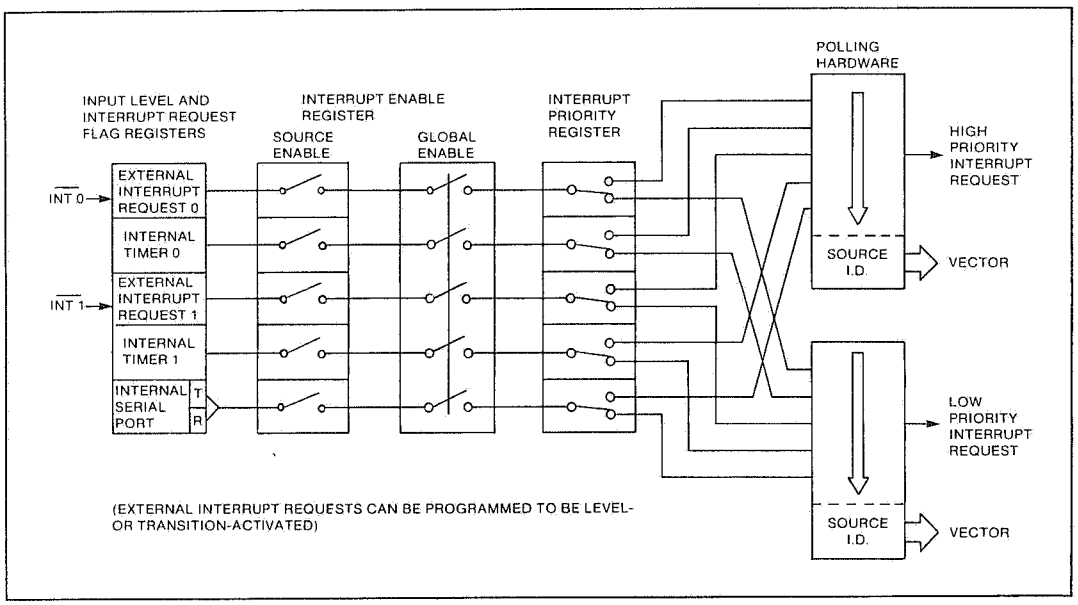


Fig. 19. CPT interrupt system

Software Structure

The CPT can be used as a controller as well as an arithmetic processor. It works in three memory spaces, see Fig. 20.

- The program memory.
- The internal data memory.
- The external data memory.

The internal data memory is divided into the internal data RAM and the special function register (SFR). The data RAM contains four eight-register banks, 128 addressable bits and the stack. The SFR contains arithmetic registers, pointers, I/O ports, interrupt system registers, timers and the serial port. 128 bit locations in the SFR are addressable as bits. The instruction set consists of 49 one-byte, 45 two-byte and 17 three-byte instructions. The instruction set is listed in Appendix 1.

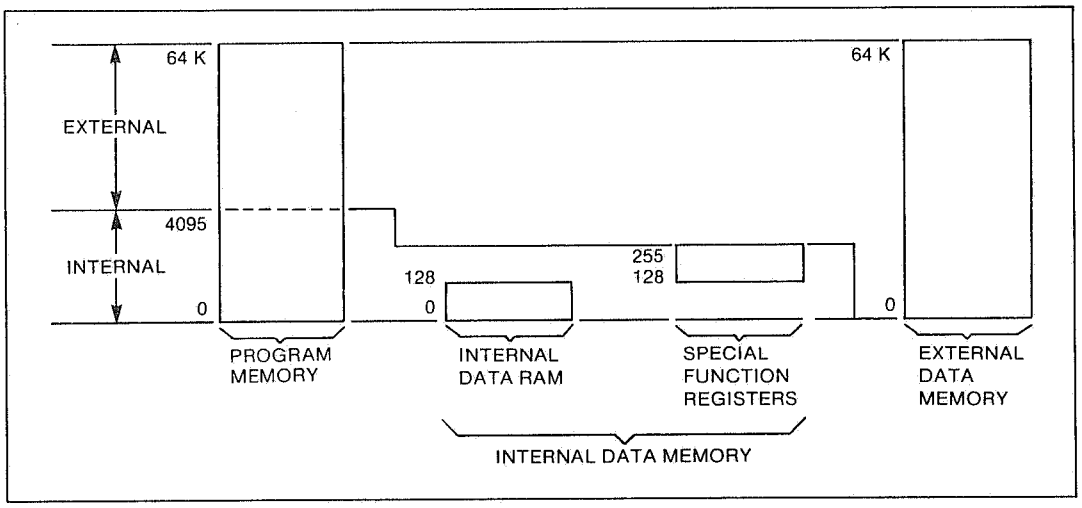


Fig. 20. Memory layout

Appendix 1

Instruction Set Summary

Notes on Instruction Set and Addressing Modes

Rn	Register R7 – R0 of the currently selected register bank.
data	8-bit internal data location's address. This could be an internal data RAM location (0 – 127) or an SFR (i.e. I/O port control register, status register, etc. (128–225).
@Ri	8-bit internal data RAM location (0 – 225) addressed indirectly through register R1 or R0.
#data	8-bit constant included in instruction.
#data 16	16-bit constant included in instruction.
addr 16	16-bit destination address. Used by LCALL & LJMP. A branch can be anywhere within the 64k-byte program memory address space.
addr 11	11-bit destination address. Used by ACALL & AJMP. The branch will be within the same 2k-byte page of program memory as the first byte of the following instruction.
rel	Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. The range is −128 to +127 bytes relative to first byte of the following instruction.
bit	Direct addressed bit in internal data RAM or SFR.

Interrupt Response Time

To finish execution of current instruction, respond to the interrupt request, push the PC and to vector to the first instruction of the interrupt service program requires 38 to 81 oscillator periods (3 to 7 μs).

Instructions that Affect Flag Settings

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	0		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C, bit	X		
MUL	0	X		ANL C,/bit	X		
DIV	0	X		ORL C, bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C, bit	X		
RLC	X			CJNE	X		
SETB C	1						

Note that operations on SFR byte address 208 or bit addresses 209 – 215 (i.e. the PSW or bits in the PSW) will also affect flag settings.

Arithmetic Instructions

Mnemonic	Description	Bytes	Cycles[1)]
ADD A, Rn	Add register to A.	1	12
ADD A, data	Add direct byte to A.	2	12
ADD A, @Ri	Add indirect RAM to A.	1	12
ADD A, #data	Add immediate data to A.	2	12
ADDC A, Rn	Add register and carry flag to A.	1	12
ADDC A, data	Add direct byte and carry flag to A.	2	12
ADDC A, @Ri	Add indirect RAM and carry flag to A.	1	12
ADDC A, #data	Add immediate data and carry flag to A.	2	12
SUBB A, Rn	Subtract register and carry flag from A.	1	12
SUBB A, data	Subtract direct byte and carry flag from A.	2	12
SUBB A, @Ri	Subtract indirect RAM and carry flag from A.	1	12
SUBB A, #data	Subtract immediate data and carry flag from A.	2	12
INC A	Increment A	1	12
INC Rn	Increment register	1	12
INC data	Increment direct byte	2	12
INC @Ri	Increment indirect RAM	1	12
DEC A	Decrement A	1	12
DEC Rn	Decrement register	1	12

Mnemonic	Description	Bytes	Cycles[1)]
DEC data	Decrement direct byte	2	12
DEC @Ri	Decrement indirect RAM	1	12
INC DPTR	Increment data pointer	1	24
MUL AB	Multiply A times B	1	48
DIV AB	Divide A by B	1	48
DA A	Decimal add adjust of A	1	12

Logic Instructions

Mnemonic	Description	Bytes	Cycles[1)]
ANL A, Rn	AND register to A.	1	12
ANL A, data	AND direct byte to A.	2	12
ANL A, @Ri	AND indirect RAM to A.	1	12
ANL A, #data	AND immediate data to A.	2	12
ANL data, A	AND A to direct byte.	2	12
ANL data, #data	AND immediate data to direct byte.	3	24
ANL C, bit	AND direct bit to carry.	2	24
ANL C,/bit	AND complement of direct bit to carry.	2	24
ORL A, Rn	OR register to A.	1	12
ORL A, data	OR direct byte to A.	2	12
ORL A, @Ri	OR indirect RAM to A.	1	12
ORL A, #data	OR immediate data to A.	2	12
ORL data, A	OR A to direct byte.	2	12
ORL data, #data	OR immediate data to direct byte.	3	24
ORL C, bit	OR direct bit to carry.	2	24
ORL C,/bit	OR complement of direct bit to carry.	2	24
XRL A, Rn	Exclusive-OR register to A.	1	12
XRL A, data	Exclusive-OR direct byte to A.	2	12
XRL A, @Ri	Exclusive-OR indirect RAM to A.	1	12
XRL A, #data	Exclusive-OR immediate data to A.	2	12
XRL data, A	Exclusive-OR A to direct byte.	2	12
XRL data, #data	Exclusive-OR immediate data to direct byte.	3	24

Mnemonic	Description	Bytes	Cycles[1)]
SETB C	Set carry.	1	12
SETB bit	Set direct bit.	2	12
CLR A	Clear A.	1	12
CLR C	Clear carry.	1	12
CLR bit	Clear direct bit.	2	12
CPL A	Complement A.	1	12
CPL C	Complement carry.	1	12
CPL bit	Complement direct bit.	2	12
RL A	Rotate A Left.	1	12
RLC A	Rotate A Left through carry.	1	12
RR A	Rotate A Right.	1	12
RRC A	Rotate A Right through carry.	1	12
SWAP A	Rotate A Left four (exchange nibbles within A).	1	12

Data Transfer Instructions

Mnemonic	Description	Bytes	Cycles[1)]
MOV A, Rn	Move register to A.	1	12
MOV A, data	Move direct byte to A.	2	12
MOV A, @Ri	Move indirect RAM to A.	1	12
MOV A, #data	Move immediate data to A.	2	12
MOV Rn, A	Move A to register.	1	12
MOV Rn, data	Move direct byte to register.	2	24
MOV Rn, #data	Move immediate data to register.	2	12
MOV data, A	Move A to direct byte.	2	12
MOV data, Rn	Move register to direct byte.	2	24
MOV data, data	Move direct byte to direct byte.	3	24
MOV data, @Ri	Move indirect RAM to direct byte.	2	24
MOV data, #data	Move immediate data to direct byte.	3	24
MOV @Ri, A	Move A to indirect RAM.	1	12
MOV @Ri, data	Move direct byte to indirect RAM.	2	24
MOV @Ri, #data	Move immediate data to indirect RAM.	2	12
MOV DPTR #data 16	Move 16-bit constant to data pointer.	3	24

Mnemonic	Description	Bytes	Cycles[1)]
MOV C, bit	Move direct bit to carry.	2	12
MOV bit, C	Move carry to direct bit.	2	24
MOVC A, @A + DPTR	Move program memory byte addressed by A+DPTR to A.	1	24
MOVC A, @A+PC	Move program memory byte addressed by A+PC to A.	1	24
MOVX A, @Ri	Move external data (8-bit address) to A.	1	24
MOVX A, @DPTR	Move external data (16-bit address) to A.	1	24
MOVX @Ri, A	Move A to external data (8-bit address).	1	24
MOVX @DPTR, A	Move A to external data (16-bit address).	1	24
PUSH data	Move direct byte to stack and increment SP.	2	24
POP data	Move direct byte from stack and decrement SP.	2	24
XCH A, Rn	Exchange register with A.	1	12
XCH A, data	Exchange direct byte with A.	2	12
XCH A, @Ri	Exchange indirect RAM with A.	1	12
XCHD A, @Ri	Exchange indirect RAM's least significant nibble with A's least significant nibble.	1	12

Control Transfer (Subroutine) Instructions

Mnemonic	Description	Bytes	Cycles[1)]
ACALL addr 11	Absolute subroutine call.	2	24
LCALL addr 16	Long subroutine call.	3	24
RET	Return from subroutine call.	1	24
RETI	Return from interrupt call.	1	24

Control Transfer (Branch) Instructions

Mnemonic	Description	Bytes	Cycles[1)]
AJMP addr 11	Absolute jump.	2	24
LJMP addr 16	Long jump.	3	24
SJMP rel	Short jump.	2	24
JMP @A+DPTR	Jump indirect relative to the DPTR.	1	24
JZ rel	Jump if A is zero.	2	24
JNZ rel	Jump if A is not zero.	2	24
JC rel	Jump if carry is set.	2	24
JNC rel	Jump if carry is not set.	2	24
JB bit, rel	Jump relative if direct bit is set.	3	24
JNB bit, rel	Jump relative if direct bit is not set.	3	24
JBC bit, rel	Jump relative if direct bit is set, then clear bit.	3	24
CJNE A, data, rel	Compare direct byte to A and jump if not equal. See Note a).	3	24
CJNE A, #data, rel	Compare immediate to A and jump if not equal. See Note a).	3	24
CJNE Rn, #data, rel	Compare immediate to register and jump if not equal. See Note a).	3	24
CJNE @Ri, #data rel	Compare immediate to indirect RAM and jump if not equal. See note a).	3	24
DJNZ Rn, rel	Decrement register and jump if not zero.	2	24
DJNZ data, rel	Decrement direct byte and jump if not zero.	3	24

Note a) Set C if the first operand is less than the second operand. Else clear.

Other Instructions

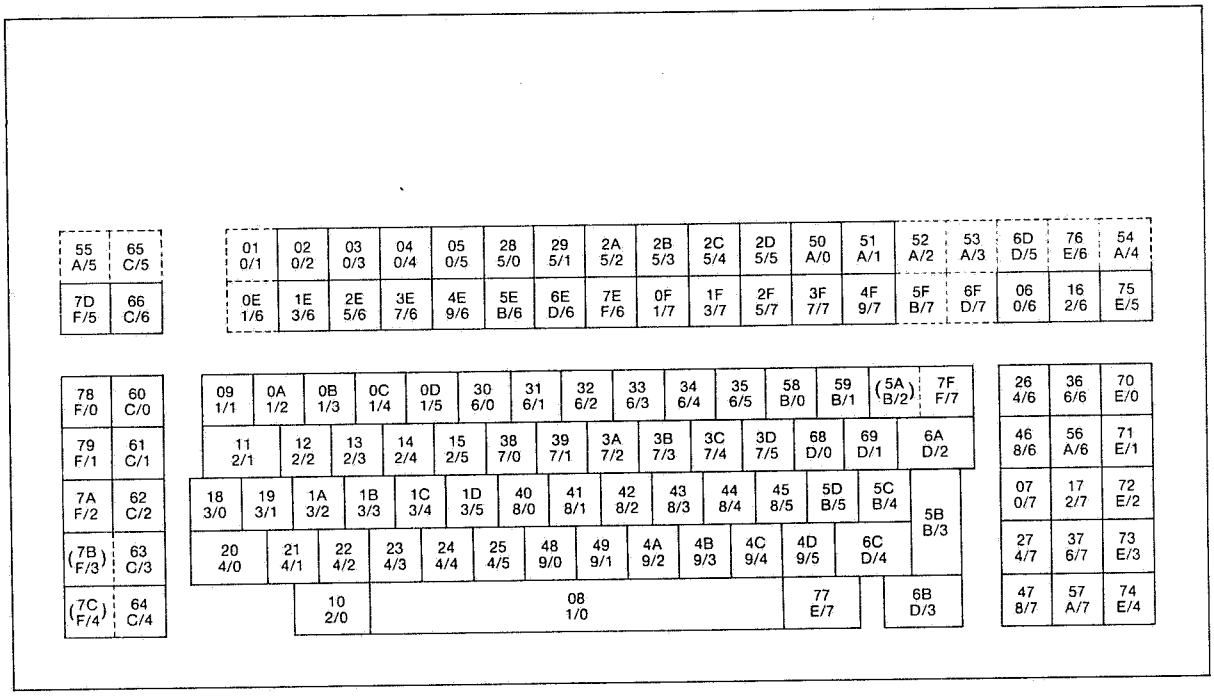
Mnemonic	Description	Bytes	Cycles[1)]
NOP	No Operation.	1	12

[1)] The cycle time is 83.33 ns.

Appendix 2

Keyboard Codes and Column/Row Numbers

KBU

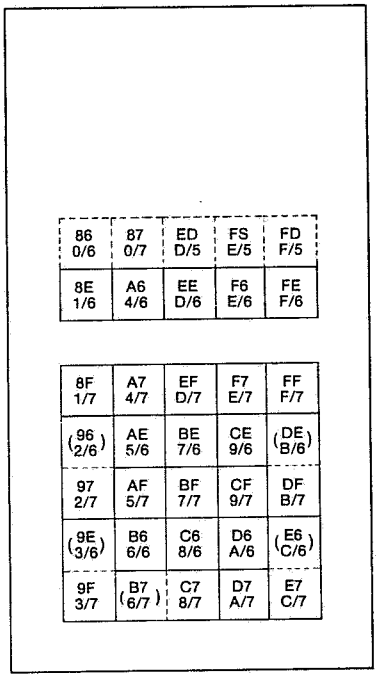


XX
X/Y

Key code (B7 = 0)

Column/row

KXU



XX
X/Y

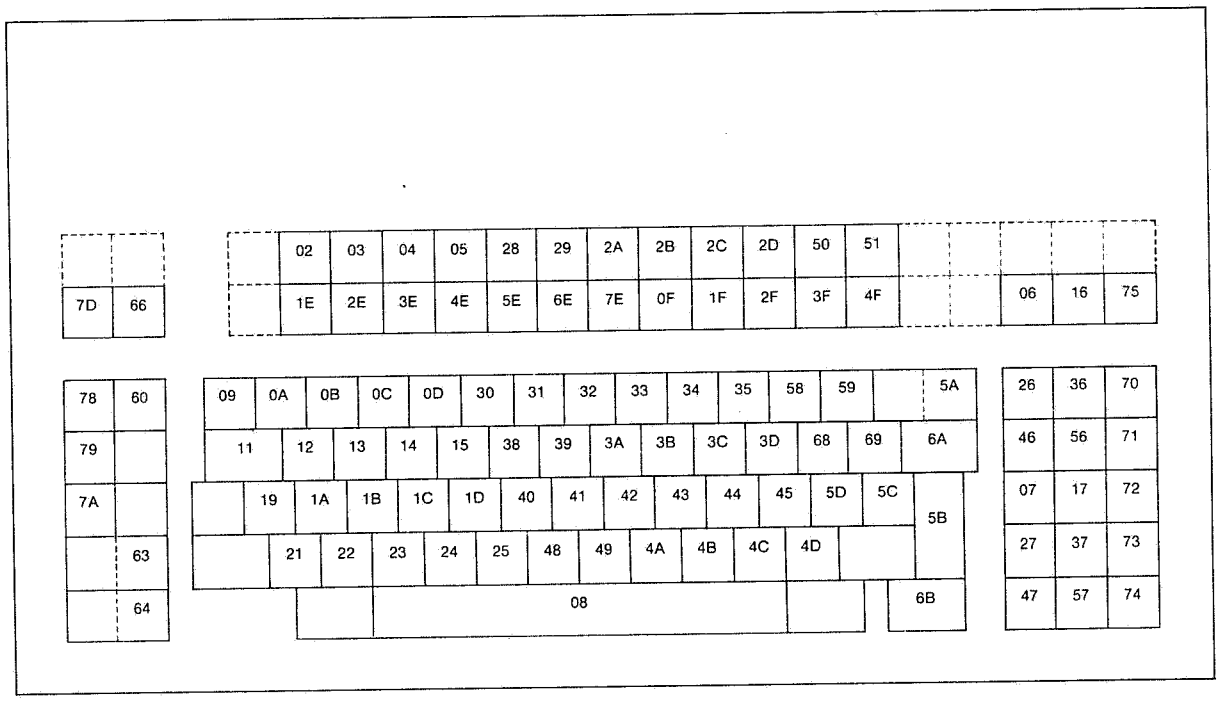
Key code (B7 = 1)

Column/row

Appendix 3

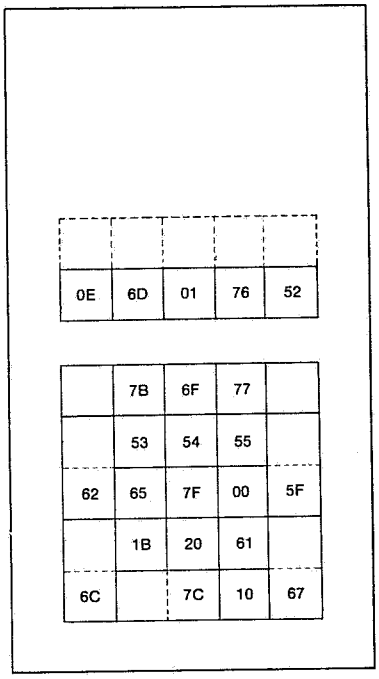
Keyboard Codes (normal codes)

KBU



For repetition, add 80 to the key code

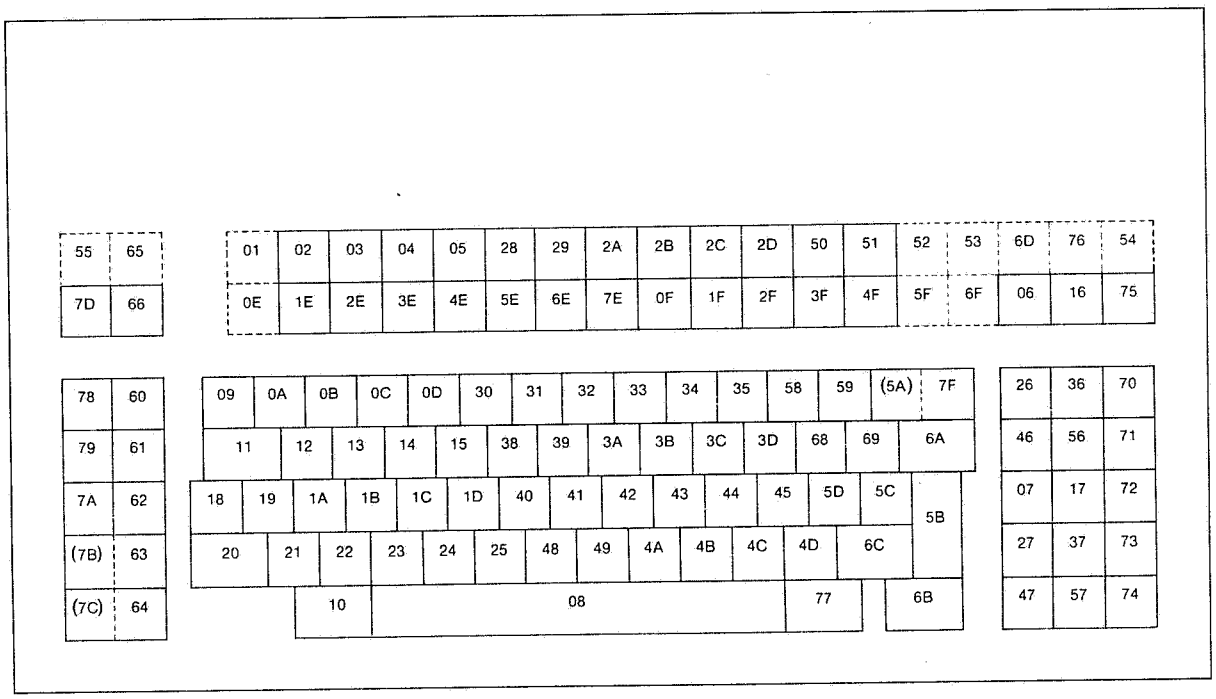
KXU



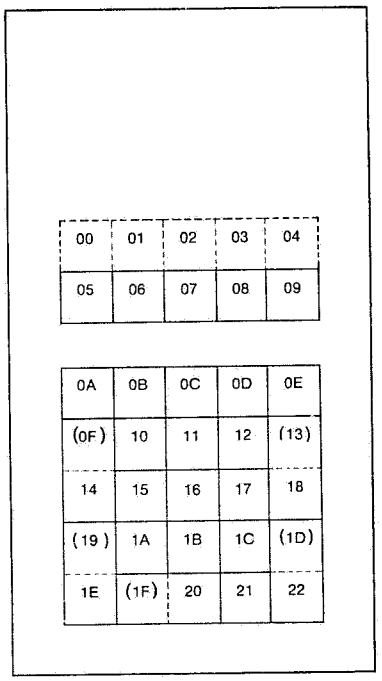
Appendix 4

Keyboard Codes (extended codes)

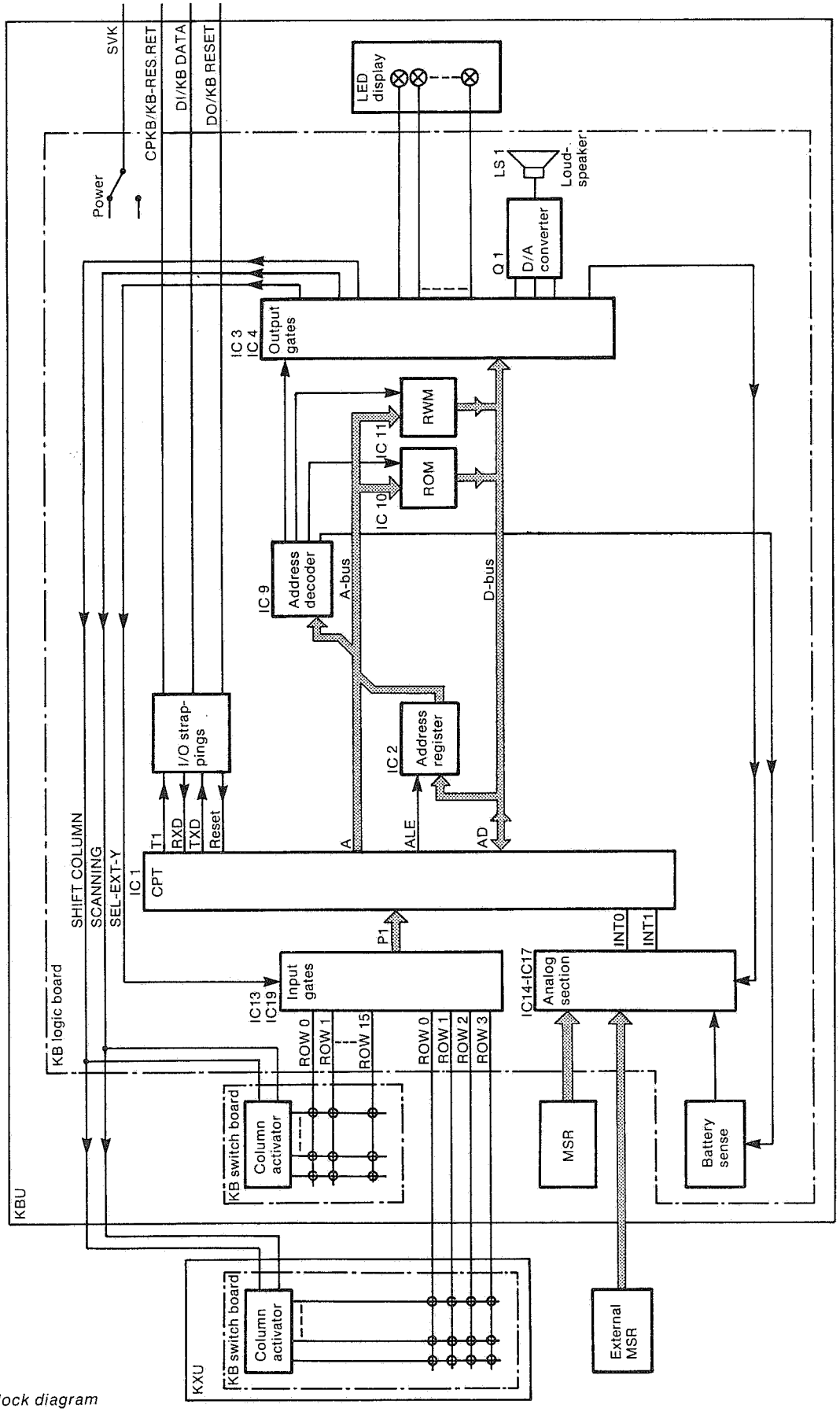
KBU



KXU



Appendix 5

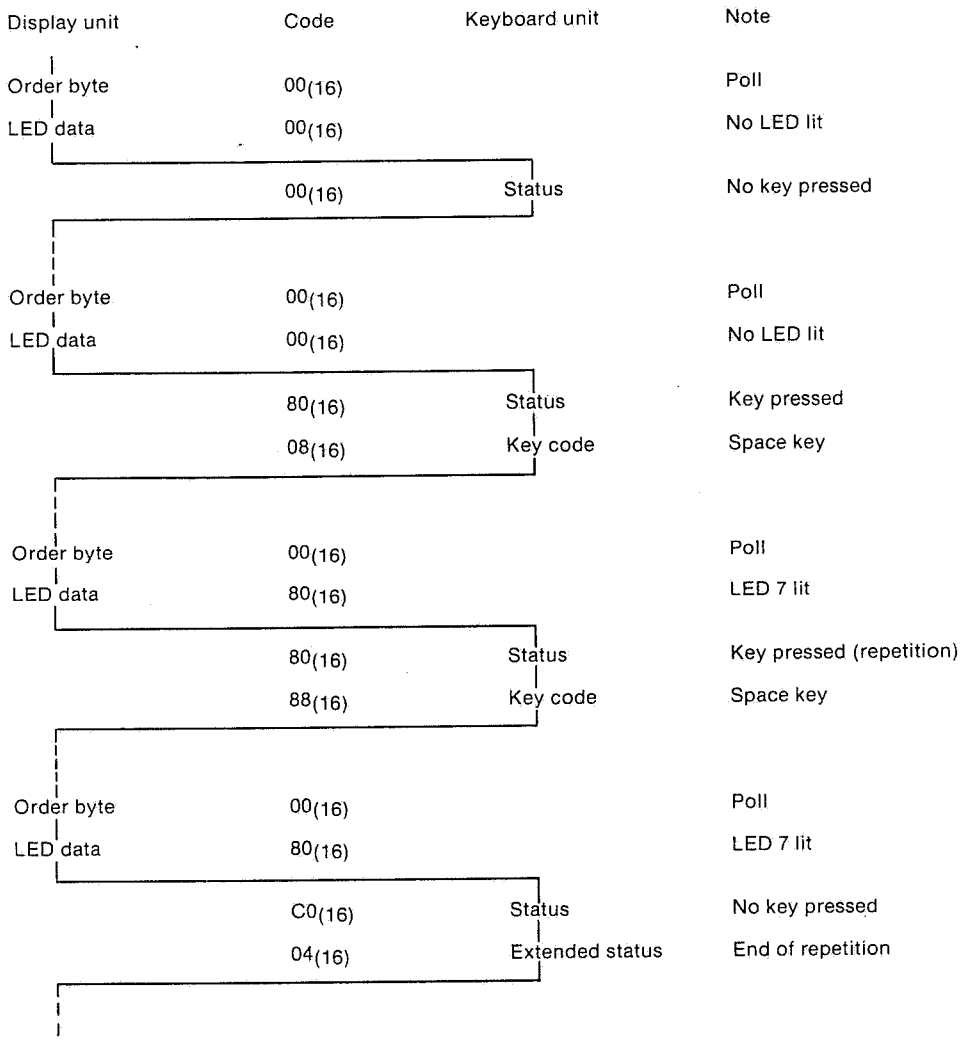


Block diagram

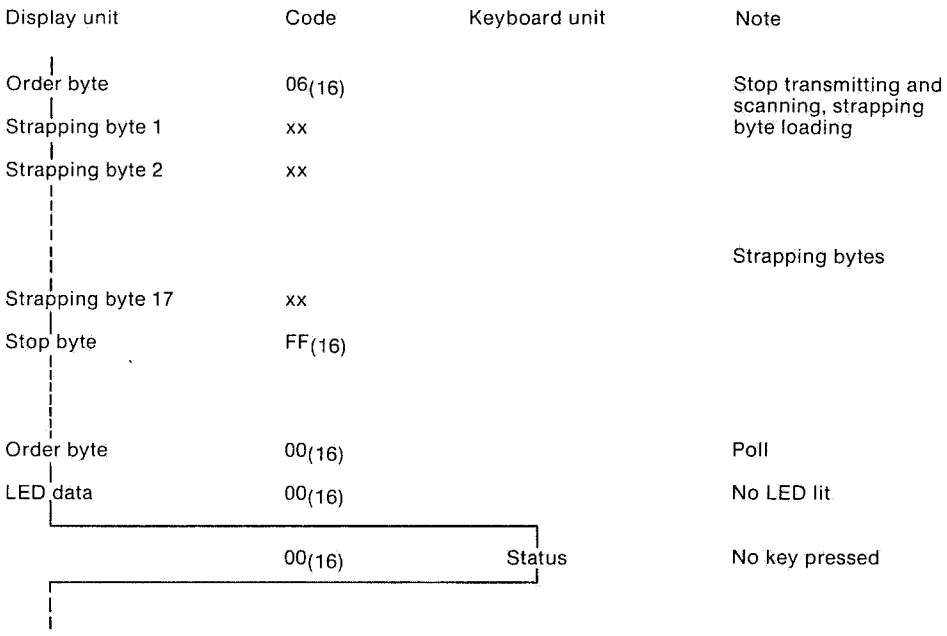
Appendix 6

Examples of Exchange of Messages between the Keyboard and the Display Unit

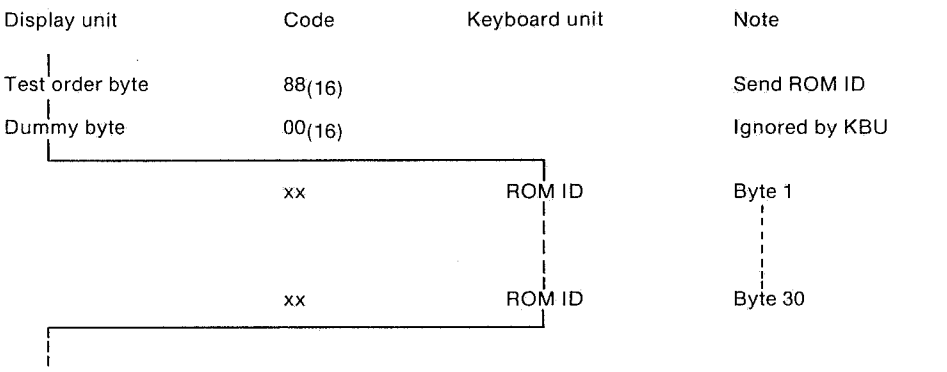
Normal Flow



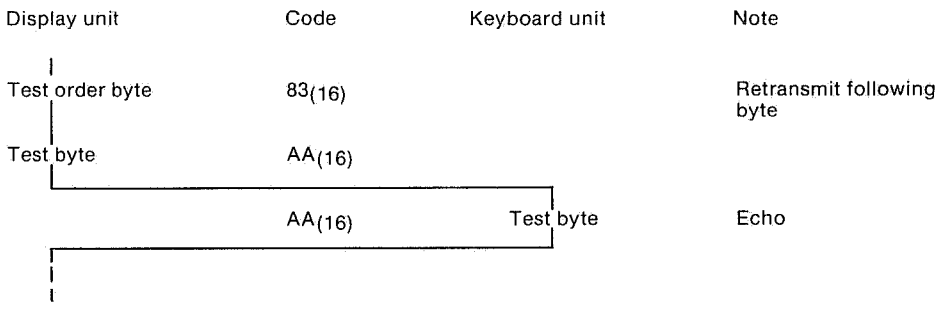
Strapping Byte Receiving



PROM ID Sending



Echo Test



DATASAAB

Magnetic Identification Device

Contents

Figures

EE362-810
1978-10-30

General

The Magnetic Identification Device, MID 4131, is used together with Keyboard 4140, to check the identity of the person using the terminal. The MID is connected to the keyboard via a 1-metre cable. The MID is made to read ID cards conforming to ISO 3554, with magnetically coded characters in track 2.

The MID consists of two main parts. One is the magnetic card sensing unit, MSU, which contains read head, amplifier, a clock pulse/data discriminator and a circuit which detects and indicates when a card passes the read head.

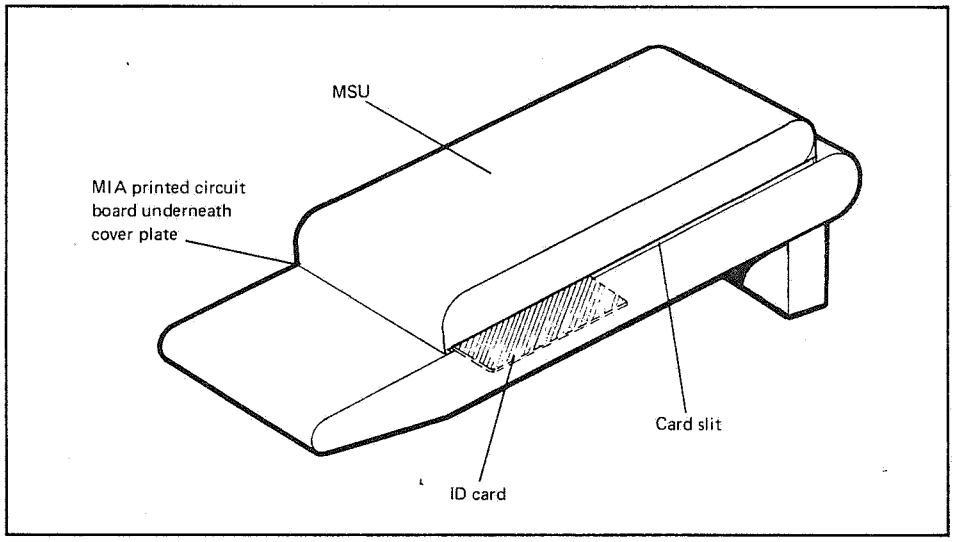


Fig. 1. Magnetic Identification Device, MID 4131

The other part, MID interface adapter, MIA, provides the interface to the keyboard and contains an interrupt generating gate, a data bit buffer and a circuit which senses and indicates whether there is a card in the MID or not. See Figs 1 and 3.

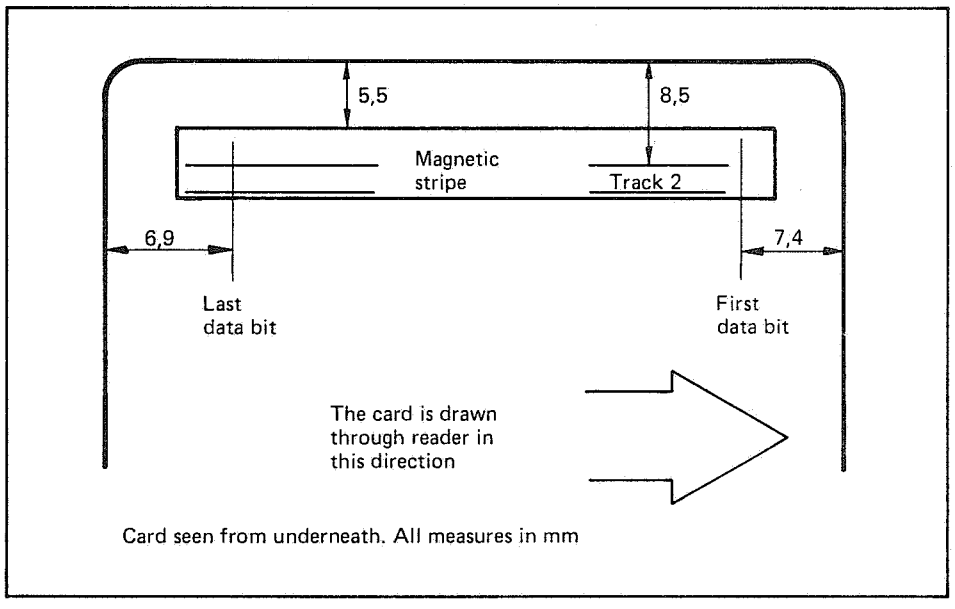


Fig. 2. Location of track 2

The cards have a capacity of 40 characters including three control characters. Each character contains four message bits and one odd parity bit as shown in the succeeding table. The characters are stored on the magnetic tape strip so that bit one is read first and bit four and the parity bit are read as last bits in each character.

Character	Binary Code				
	P	B4	3	2	1
START	0	1	0	1	1
0	1	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	1	0	0	1	1
4	0	0	1	0	0
5	1	0	1	0	1
6	1	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	1	1	0	0	1
STOP	1	1	1	1	1
LRC	X	X	X	X	X
	(1	0	1	0	1)

Note: The LRC character is formed to make the total number of ones in each bit position (B4—B1) in the message even. The p bit of the LRC character is just a parity bit for that character. (The LRC character would thus be 10101 if the above list were a message.)

The magnetic encoding technique is known as two-frequency, coherent-phase recording. This method allows for serial recording of self-clocking data. The data comprise data and clocking bits together. A flux transition occuring between clocks signifies a one, the absence of a flux transition signifies a zero. The whole message on the card begins and ends with a number of zeros.

Function

The magnetic card is drawn through the MID slit towards the operator, the magnetic tape downwards and to the left (nearest the bottom of the slit), at a speed of 8 to 100 cm/s.

When the magnetic tape passes the read head of the MID, its coded information as well as its clock signal is sensed by the head.
The output signal from the head is amplified and fed to a discriminator, which separates the clock pulses from the data bits.
The amplified signal is also applied to a circuit providing the internal signal CLS. This signal indicates that a card presently is read by the head.
The discriminator thus provides serial data, RDD, and clock pulses, RCP.
RCP and CLS are gated and fed to the keyboard as interrupt signals. The gated signal is also used to clock data into an output latch, the state of which is tested by the keyboard logic at each interrupt.
Thus recorded data from the ID card are fetched by the keyboard logic, where the ID characters are stored in an RWM area.

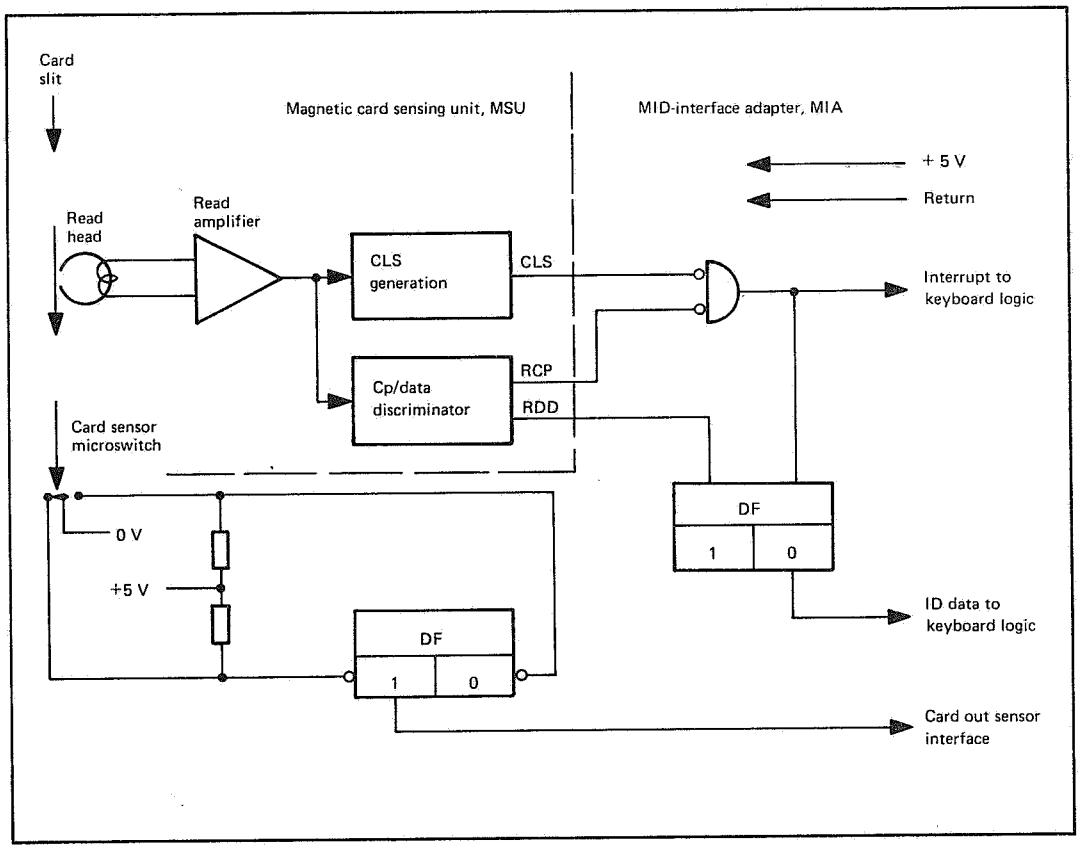


Fig. 3. MID block diagram

When the card has fully passed the read head, it reaches the end of the reader's slit where it is stopped by an edge. At this position the card affects a microswitch, which in turn sets a latch. The output of this latch is applied to the keyboard logic, which continuously scans the state of it. Any change in state is indicated by the keyboard logic to the display unit as "ID card in" or "ID card out" in the status byte.
At power on "ID card out" is indicated by the keyboard logic.

The MID is connected to the keyboard via a wire containing six twisted pairs of which four pairs are used. The MID logic is supplied with power (+5 V, 0 V) from the keyboard via one of the pairs.

DATASAAB

Selector Pen Device

Contents

Figure

EE363-810
1981-04-01

GENERAL

Selector Pen Device 4130 can be connected to the Display Unit 4110. It is used to select selector-pen-detectable fields on the screen, thus permitting the display unit to be used without a keyboard. When the selector pen is used together with a keyboard, selection operations can be carried out using either the selector pen or the keyboard.

The selector pen is activated when the tip of the pen is pressed against the screen, whereupon horizontal lines are displayed, with high brightness. These lines are displayed from left to right in each text line and from the top line to the bottom line.

When such a line hits the tip of the pen the horizontal lines disappear. If the corresponding character position belongs to a selector-pen-detectable field, the field is identified by the system.

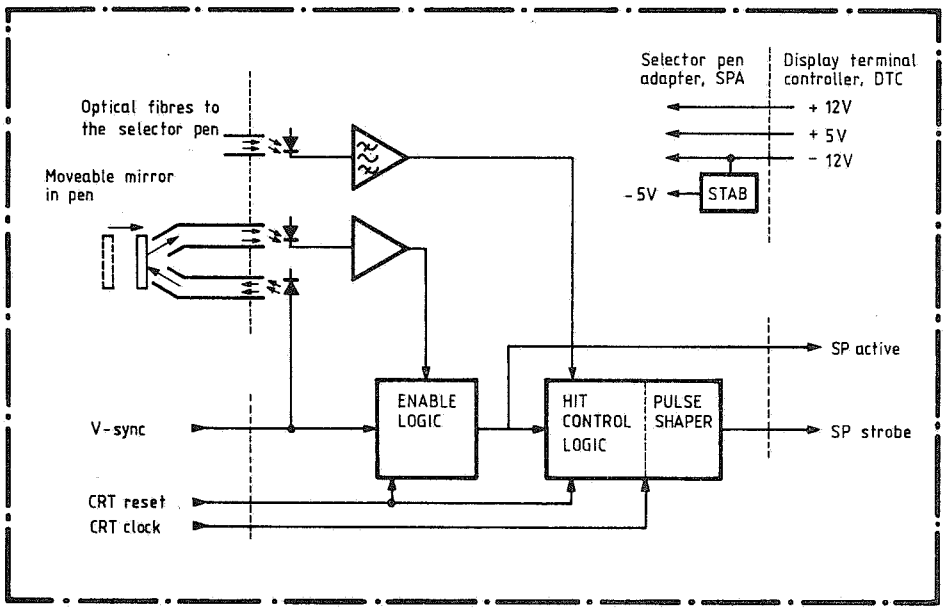


Fig. 1 SPD block diagram

FUNCTION

This section describes the design and construction of the selector pen device and explains how it operates. See Fig. 1.

The Selector Pen Device 4130 can be divided into the following parts:

- Pen
- Fibre optic cable
- Printed circuit board

o Pen and Fibre optic cable

The pen contains an optical lens which detects light emanating from the screen and a mechanical/optical switch used to activate the pen.

The Fibre optic cable contains three optical fibres. One is used for screen detection and the others for activation of the pen.

When the pen is pressed against the screen, the tip of the pen is depressed. This depression moves a mirror inside the pen in such a way that the incoming light from one optical fibre is reflected out into a second optical fibre. Light in this second optical fibre is detected on the printed circuit board, and the selector pen device is activated.

Light emanating from the screen is sent to a third optical fibre via a lens in the tip of the pen and is detected by a photodetector on the printed circuit board.

o Printed circuit board.

The printed circuit board (selector pen adapter, SPA) is located at the rear of the display unit. This board contains circuits that detects light signals from the pen and circuts for hit signal processing and intensity control of the CRT during operation.

- Enable logic

The enable logic is controlled by the "V-sync" signal. This signal clocks the circuit and also controls a light emitting diode (LED). Light from the LED is sent via one of the optical fibres to the selector pen, and if the tip of the pen is depressed the light is reflected back via another optical fibre to a photodetector. The amplified signal from the photodetector then activates the enable logic.

This means that the enable logic output signal "SP active" is generated. This signal is sent to the hit control logic which is activated. Moreover it is sent to the CRT control logic on the display terminal controller (DTC), which responds by presenting horizontal lines on the screen. These lines advance from left to right in each text line and from the top line to the bottom line (too fast for the eye to see).

The enable logic can be reset in two ways:
either by the "CRT reset" signal (When power is switched on to the display unit) or by deactivating the pen (lifting the pen from the screen). The "V-sync" signal in the next picture then resets the enable logic.

- Hit control logic and pulse shaper

  The hit control logic, which is enabled by the "SP active" signal, scans for horizontal lines on the screen.

  When the tip of the pen is illuminated (hit) by a horizontal high-brightness line, the light proceeds via an optical lens and an optical fibre to the photodetector. The rise in signal (light changing form low to high brightness) is fed via a bandpass amplifier to the hit control logic, where it is detected as a hit. The hit control logic is clocked by the CRT clock, so that when a hit is detected the "SP strobe" pulse goes active for a certain number of "CRT clock" pulses. This takes place in the pulse shaper.

  The horizontal lines on the screen advance in response to the incrementation of the content of the character address counter in the DTC. When the "SP strobe" signal becomes active after a hit, the character address counter contains the address of the character position in which the hit was made.

  This address is stored in a register in the CRT-controller, by means of the "SP strobe" pulse, and an interrupt signal is sent to the MPU in the display unit.

- Power supply

  The selector pen device is supplied with +5 V, +12 V and -12 V from the DU. Some circuit on the printed circuit board requires -5 V which is obtained from a stabilizer on the board.

Peripheral Control Unit, PCU 4171

Contents

General

Peripheral Control Unit PCU 4171 connects a printer unit to a communication processor.

The microcomputer chapter ought to be understood before this chapter is studied. A good orientation on the Communication chapter is also recommended; especially the Hardware part under Internal Communication via Two-wire.

This chapter will in principle only treat circumstances not discussed or solved differently in the two chapters mentioned above.

Functions

The peripheral control unit communicates with a communication processor. It handles data transmission from the communication processor to the printer unit.

Communication with the communication processor is synchronous and carried out via two-wire or coaxial cable. Communication with the printer unit is asynchronous and carried out via V24-bus.

A display unit may be connected to the communication processor in chain with the peripheral control unit.

Subunits

The main parts of the peripheral control unit are:

- PCM, peripheral control unit mechanics exclusive of power supply (see below).
- FPS, flexible disk power supply. See chapter Power Supplies.
- GPB, general processor board, containing a microcomputer and a two-wire interface.
- ACA-A board, asynchronous communication adapter which provides a serial interface for printers. See chapter Asynchronous Communication Adapter.

Mechanics

The basic mechanics of the PCU is shown in Fig. 1.

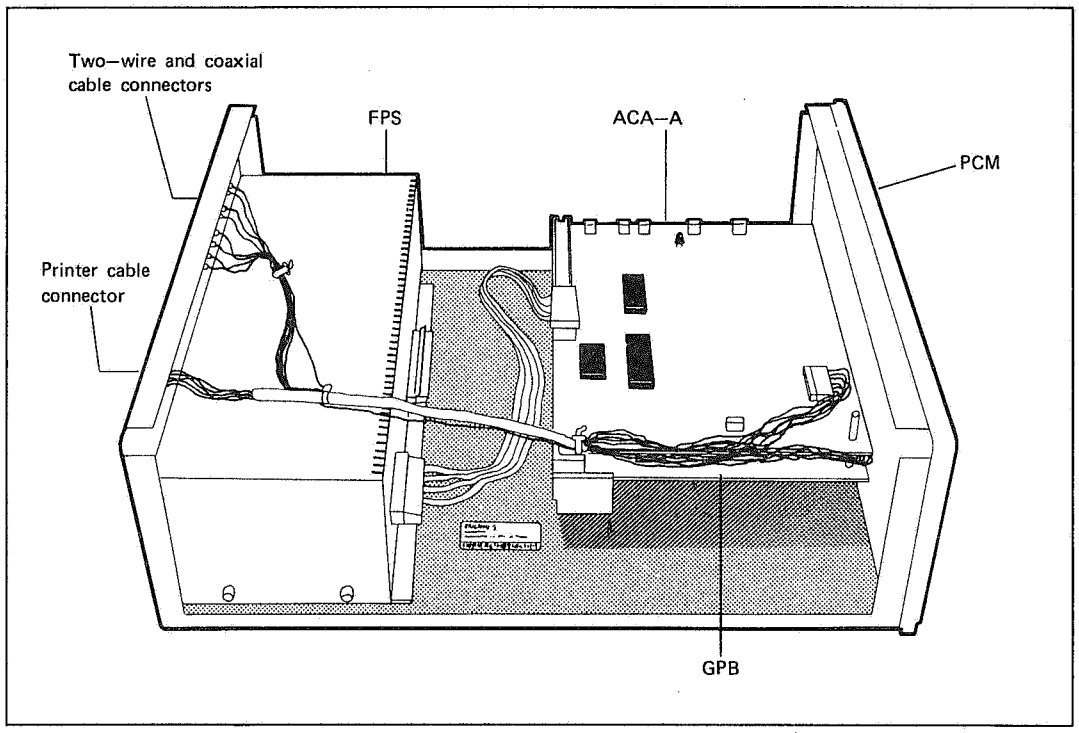


Fig. 1. Basic mechanics of PCU 4171

At the back of the housing, there are connectors for printer cable, two-wire and coaxial cable. At the front, there are a reset button, a DIP-switch and three LED-indicators, which are all part of the GPB board.

The ACA-A board has a LED-indicator, which is hidden inside the PCU.

Brief Outline

Two-wire Interface

A general description of the two-wire interface is found in the Communication chapter. The functions are therefore only summarized here. A comprehensive treatment is found in the Detailed Description in this chapter.

By means of the two-wire interface, data can be transferred to or from a buffer in the basic read/write memory from or to equipment external to the peripheral control unit via a two-wire (or coaxial) line. The transmission on the two-wire is in serial format with a rate of 300 kbit/s.

A display unit may be connected in chain with the peripheral control unit. If not, a terminating resistor must be connected in order to avoid reflexions on the line. Two of the switches at the front are used for this purpose (one for the two-wire and the other for the coaxial line). On the other 8 switches the SS3-address is indicated.

The two-wire interface consists mainly of:

- A direct memory access controller, DMAC, circuit which handles direct memory access, DMA.
- An advanced data link controller, ADLC, which converts 8-bit parallel data to serial data and vice versa. It also supervises the transmission.
- A high speed modem which converts serial data from NRZ format to frequency shifted data and vice versa. It also generates a bit clock and a Clear to send signal.
- A line interface which contains line drivers, line receivers and a line transformer.

The DMAC is connected to work in two channel mode. Channel 0 is used for transmitting and channel 1 for receiving.

Printer Connection

The printer is connected to the ACA-A-board (see that chapter). Communication with the printer unit is carried out via V24-bus, which consists of 8 lines out of which one is used for data transmission.

The ACA-A board is connected to the microcomputer on the GPB board via the following signals:

- Data bus
- Low-order address bits (A0-A7) of the address bus.
- R/W
- Per I/O from the address decoder of the GPB board. Enables the address decoder of the ACA-A board.
- Reset
- IRQ2 from the printer interface of the ACA-A board.
- IRQ4 from the modem interface of the ACA-A board. (This interrupt is called I3 on the GPB board.)
- Ø2 which is used to synchronize the ACA-A board with the microcomputer.

Memory Map

The memory map (see Fig. 2) differs somewhat from the description in the Microcomputer chapter.

The total map is divided into an ordinary read/write memory (RWM) area, an input/output (I/O) area and an initial program load (IPL) area.

The ordinary RWM area is located at the addresses $0000_{(16)}$ – $F6FF_{(16)}$. The physical memory is actually 64 kbytes ($0000_{(16)}$ – $FFFF_{(16)}$).

No parity checking will occur in the RWM and battery backup is not available.

The following circuits and units can be addressed within the I/O area:

- PIA on the GPB board (microcomputer PIA) for several I/O-purposes (e.g. interrupt identifying), PTM for time interrupts and MCP for test purposes.
- ACIA, PIA and PTM on the ACA-A board for printer (or modem) transfers and control. Addresses to printer- and modem-strapped ACA-A boards are separated.
- ADLC and DMAC for two-wire transfers and control.

Detailed I/O addresses are listed in Appendix 1.

The IPL area consists of 2 kbytes of read only memory and contains among other things four interrupt vectors at the addresses $FFF8_{(16)}$ – $FFFF_{(16)}$.

Interrupt Handling

The principles of the interrupt system on the GPB board are quite different from those described in the Microcomputer chapter. They are therefore described below except for the internal MPU functions which of course are the same.

The locations of the interrupt vectors (i.e. the addresses to the interrupt routines) are found in the memory map.

No hardware interrupt prioritizing exists (the interrupt register, mask register and address modifier are thus left out).

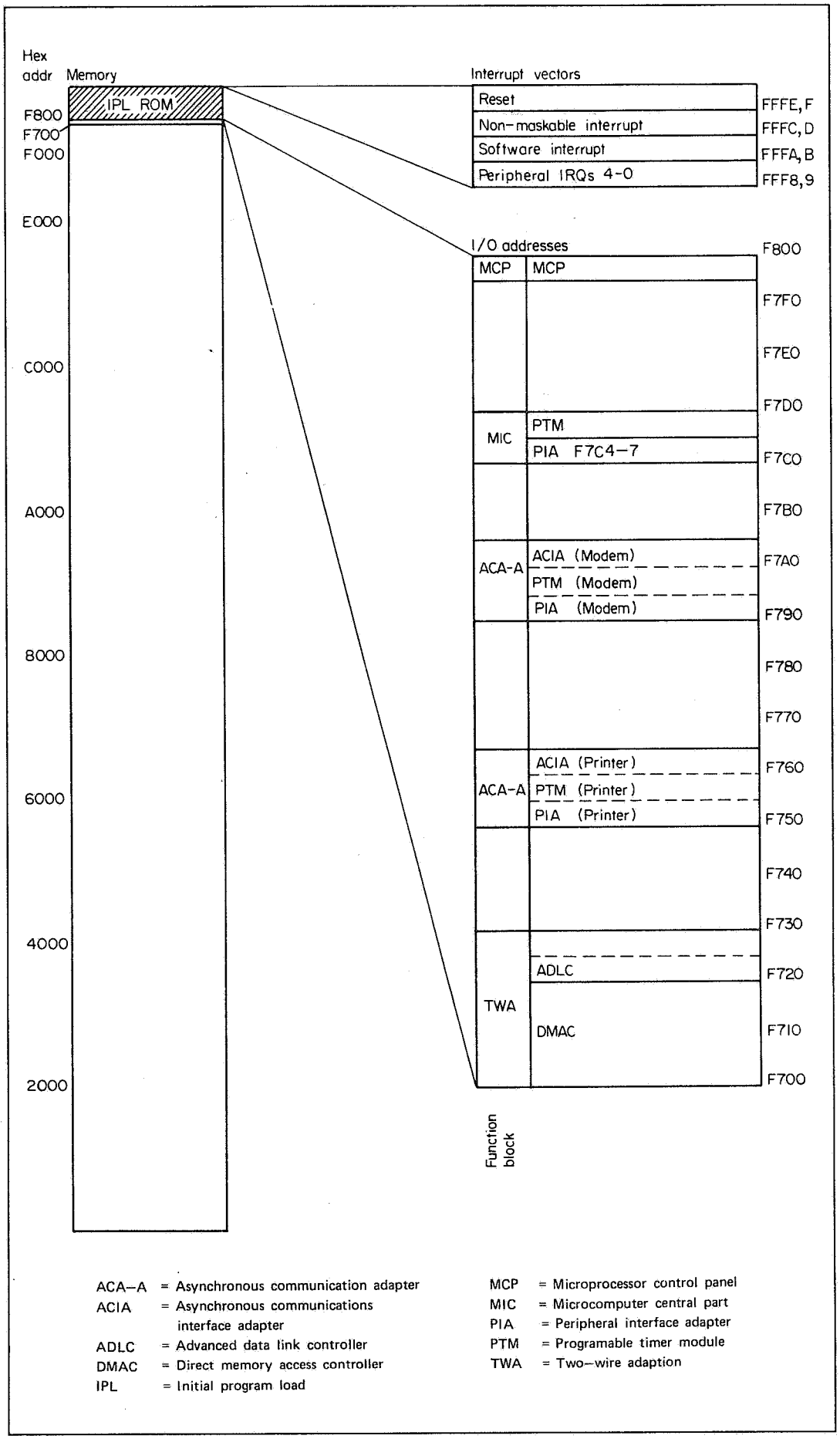


Fig. 2. PCU 4171 memory map

Reset, Non-maskable and Software Interrupts

The MPU and the microcomputer peripheral interface adapter, MIC PIA, are reset at power on. A general reset is also activated.

The MPU is also reset at a depression of the Reset button. The signal from the Reset button resets only the MPU, but is also fed to the control line CB1 of the MIC PIA. The program can thus check the type of reset (by reading the MIC PIA control register B and testing bit 7, the IRQB1 flag) and then, if suitable, generate a general reset (by setting and resetting bit 6 of MIC PIA peripheral register B). Note that the general reset signal does not reset the MPU or the MIC PIA.

The MPU will after a reset fetch the contents of the memory cells $FFFE_{(16)}$ and $FFFF_{(16)}$ to the program counter and execute a reset program.

NMI, non-maskable interrupt can only be activated by the microprocessor control panel, MCP, and causes the MPU to fetch the address of the interrupt routine from the cells $FFFC_{(16)}$ and $FFFD_{(16)}$.

SWI, software interrupt is generated by an MPU instruction and causes the MPU to fetch the address of the interrupt routine from the cells $FFFA_{(16)}$ and $FFFB_{(16)}$.

Interrupt Requests

The interrupt request, IRQ, input of the MPU can be deactivated by the MCP or be activated by five interrupts, I0 – I4. The five interrupts are or-ed together and each one is fed to a specific input of the MIC PIA. The address to the interrupt routine is located at $FFF8_{(16)}$ and $FFF9_{(16)}$. The interrupt routine is able to identify the interrupt by reading the PIA PB-register (see Detailed Description).

The interrupts are further explained below:

I4 originates from the ADLC (advanced data link controller) of the two-wire interface.

I3 originates from the modem interface of the ACA-A board.

I2 originates from the printer interface of the ACA-A board.

I1 originates from the PTM (programmable timer module) on the GPB board.

I0 originates from the CA2 output of the MIC PIA. The program can thus make an interrupt by writing $110_{(2)}$ in bits 5 – 3 of the MIC PIA control register A.

Detailed Description

This section will in principle only cover more complicated circuits or special solutions.

Microcomputer

The microcomputer in PCU 4171 differs mainly from the one described in the Microcomputer chapter regarding the address decoder, the use of the MIC PIA and the interrupt- and DMA-handling.

Basic Timing

A 19.17 MHz clock signal is generated by a crystal clock. From this clock all timing signals, e.g. dynamic memory timing, two-wire transfer bit clock etc are derived. Four basic clocks are generated by means of frequency division and a clear signal (see Fig. 3). Combinations of these are used to define different points of time during each half of the system clock period. The system clock, Ø, (microprocessor instruction clock) has a frequency of 1.065 MHz. It appears in several phases and with different pulse-pause ratios to compensate for propagation delays etc.

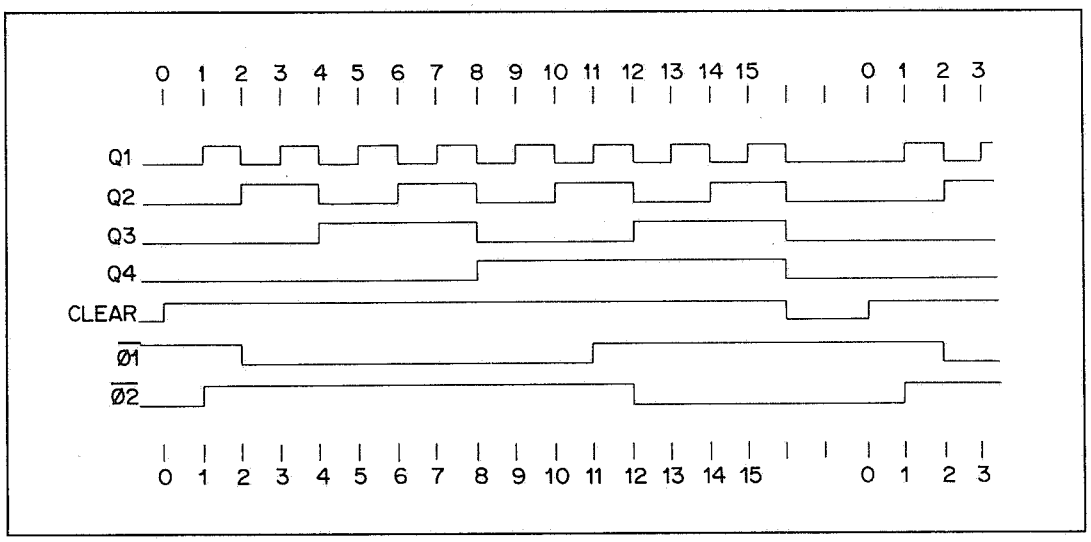


Fig. 3. Basic timing signals

MPU

The NMI input is only connected to the MCP. The DMA-request (and a MCP-signal) is fed to the HALT input, and the TSC (three-state control) is hardwired low. The inverted Ø1 clock is fed to the DBE (data bus enable) input, which takes the data bus to high impedance state during the address-set up period. The address and data buses as well as the R/W-line are taken to high impedance state when the processor is halted. VMA (valid memory address) is fed to the address decoder. BA (bus available) and a MCP-signal is fed to the DGRNT (DMA grant) of the DMAC via an and-gate.

Address Decoding

The address decoder consists of two PAL's (programmable array of logic gates).

Input signals are the address bus. (8 address bits are fed to each PAL), the VMA- and R/W-signal from the MPU, TxSTB from DMAC and an MCP-signal (PV).

Output signals are chip select signals to the following circuits: PIA, PTM, DMAC, ADLC, RWM, IPL on the GPB-board, and the address decoder of the ACA-A board.

Direction of the data bus buffer is also controlled by an output signal.

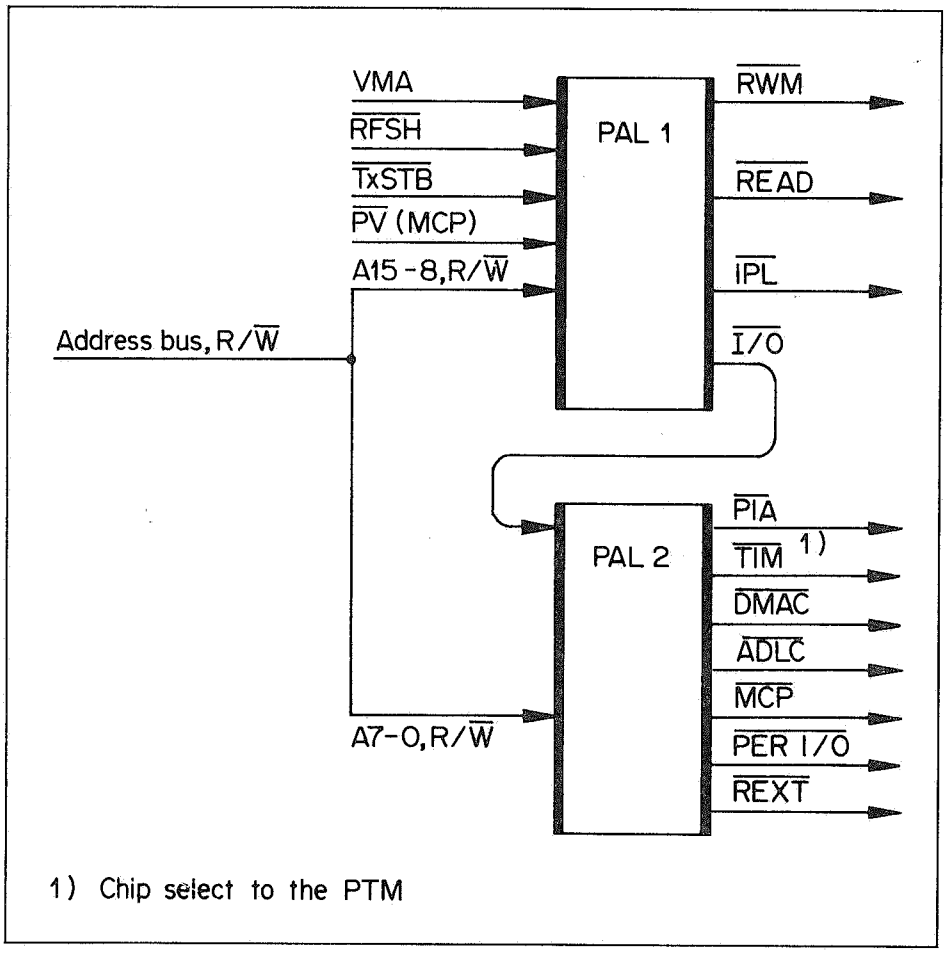


Fig. 4. Address decoder

Note that some input combinations in the decoder table may not occur according to the functions of other circuits (for example TxSTB and VMA will not be high at the same time).

High order decoder, PAL 1

Activating input combinations						Activated outputs	Comments
A15 – 8 (Hex)	VMA	PV	TxSTB	R/W	RFSH		
0 – F6	H X X	X L X	X X L	H H H	H H H	RWM,READ	The accessible RWM-area is 63232 bytes (address 0 – F6FF) while the physical area is 65536 bytes (0 – FFFF)
	H X X	X L X	X X L	X X X	X X X	RWM	
F7	H X X	X L X	X X L	X X X	X X X	I/O	Enables PAL 2 (I/O-decoder) The ADLC, however, may be selected from the DMAC without this signal
F8 – FF	H X	X L	X X	H H	X X	IPL	

I/O decoder, PAL 2

Activating input combinations				Activated outputs	Comments
A7 – 0 (Hex)	I/O	TxSTB	R/W		
X	L	X	X	PER I/O	Enables the ACA-A decoder
0 – 1F	L	X	X	DMAC	
20 – 27	L X	X L	X X	ADLC	Enables ADLC during MPU mode and DMA mode
C4 – C7	L	X	X	PIA	
C8 – CF	L	X	X	TIM	
F8 – FF	L	X	X	MCP	
28 – C3 D0 – F8	L L	X X	H H	REXT	Controls direction of ACA-A databus buffer

Microcomputer PIA, MIC PIA

The MIC PIA serves as an interface between the MPU and several I/O-signals. See chapter Microcomputer; Peripheral Interface Adapter, PIA for definitions and a general survey. IRQA, IRQB, CA1 and PB4, however, are not used.

The PA-pins are programmed to be inputs and receive the SS3-address from the dip-switch.

An interrupt flag (CRB7) is set if the Reset button is depressed (as CB1 is then taken low).

CB2 is connected to one of the LED indicators. It is brought to alternate between high and low while the PCU is loaded, and when ready it goes low.

The PB-pins are used in the following way:

PB7	Input	I4, interrupt from ADLC
PB6	Output	1 = Activate general reset. The output must be set low after power on. The MPU and the MIC PIA are not reset by a general reset.
PB5	Input	Used by the MCP.
PB3	Input	I3, interrupt from the modem interface of the ACA-A board.
PB2	Input	I2, interrupt from the printer interface of the ACA-A board.
PB1	Input	I1, interrupt from the PTM, programmable timer module.
PB0	Input	IO, interrupt from CA2 set by the MPU itself.

Programmable Timer Module, PTM

A PTM, programmable timer module is used in the peripheral control unit for time interrupt purposes.

The PTM contains three timers. Two of them is used in the PCU. Timer 1 is clocked by Ø2 and generates an interrupt and a pulse on its output pin every 20 ms.

Timer 2 is clocked by the output pulse from timer 1 and generates interrupts during load only. The interval between interrupts are a second (before the first poll) or half a second (after the first poll).

The interrupt from timer 1 makes the PCU software compatible with a display unit (this interrupt corresponds to the vertical synchronization signal of the DU).

The interrupt from timer 2 handles the flashing of LED-indicator "Ready" via PIA.

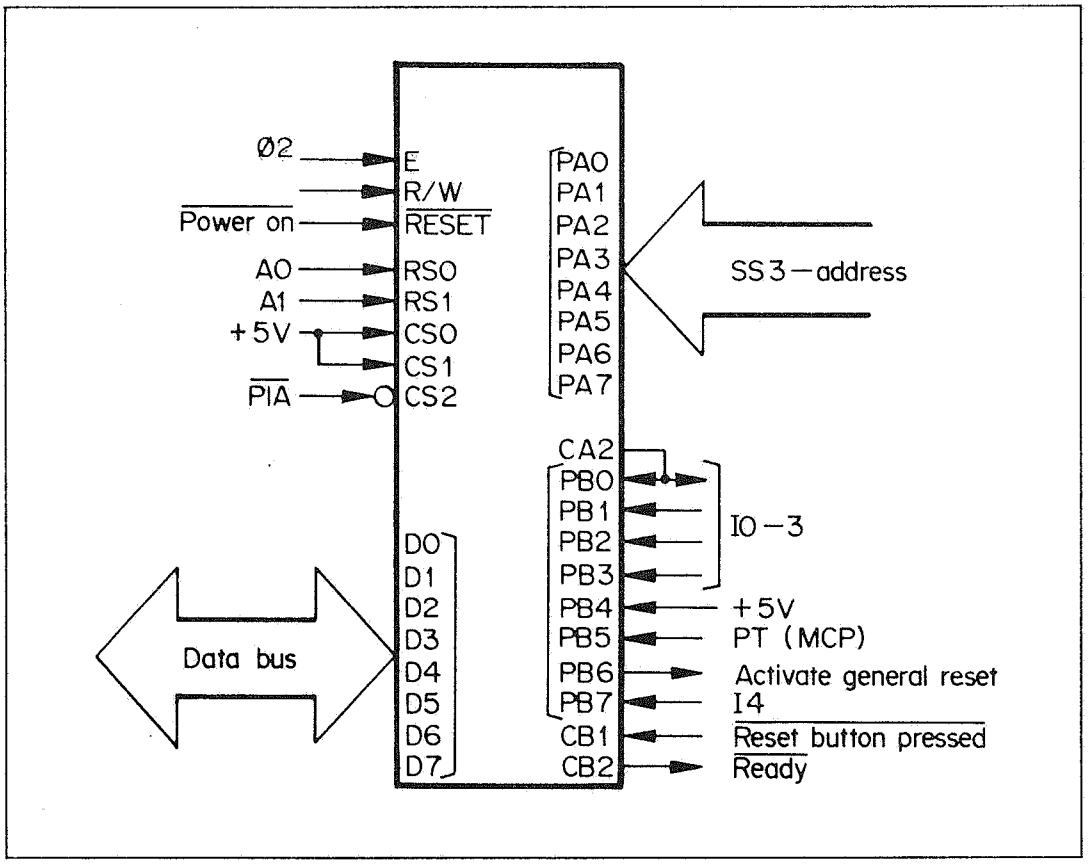


Fig. 5. PIA inputs and outputs

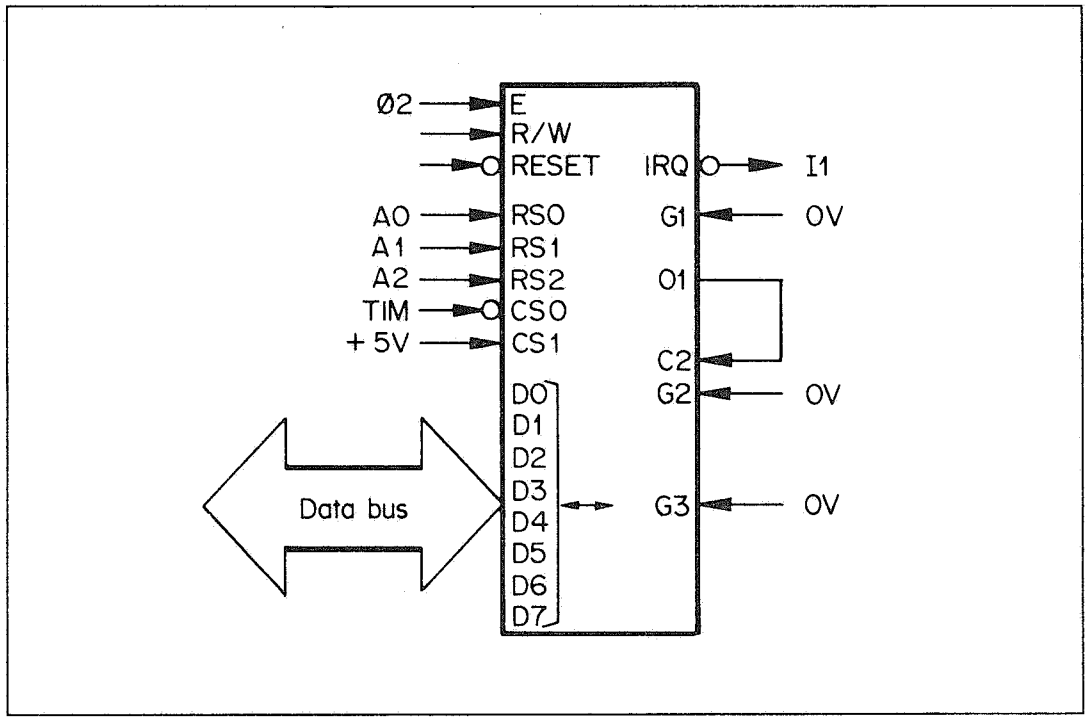


Fig. 6. PTM inputs and outputs

Read/Write Memory

The read/write memory, being dynamic, has to be refreshed. Addressing and refreshing are handled by a DMC, dynamic memory controller. When RFSH is active the address originates from the refresh counter of the DMC. When RFSH is inactive the address originates from the MPU or the DMAC.

The memory consists of 65536 × 8 bits and is organized in 256 rows and 256 8-bit columns. The combined row and column address define one memory location. They are clocked into the memory by RAS- and CAS-signals respectively.

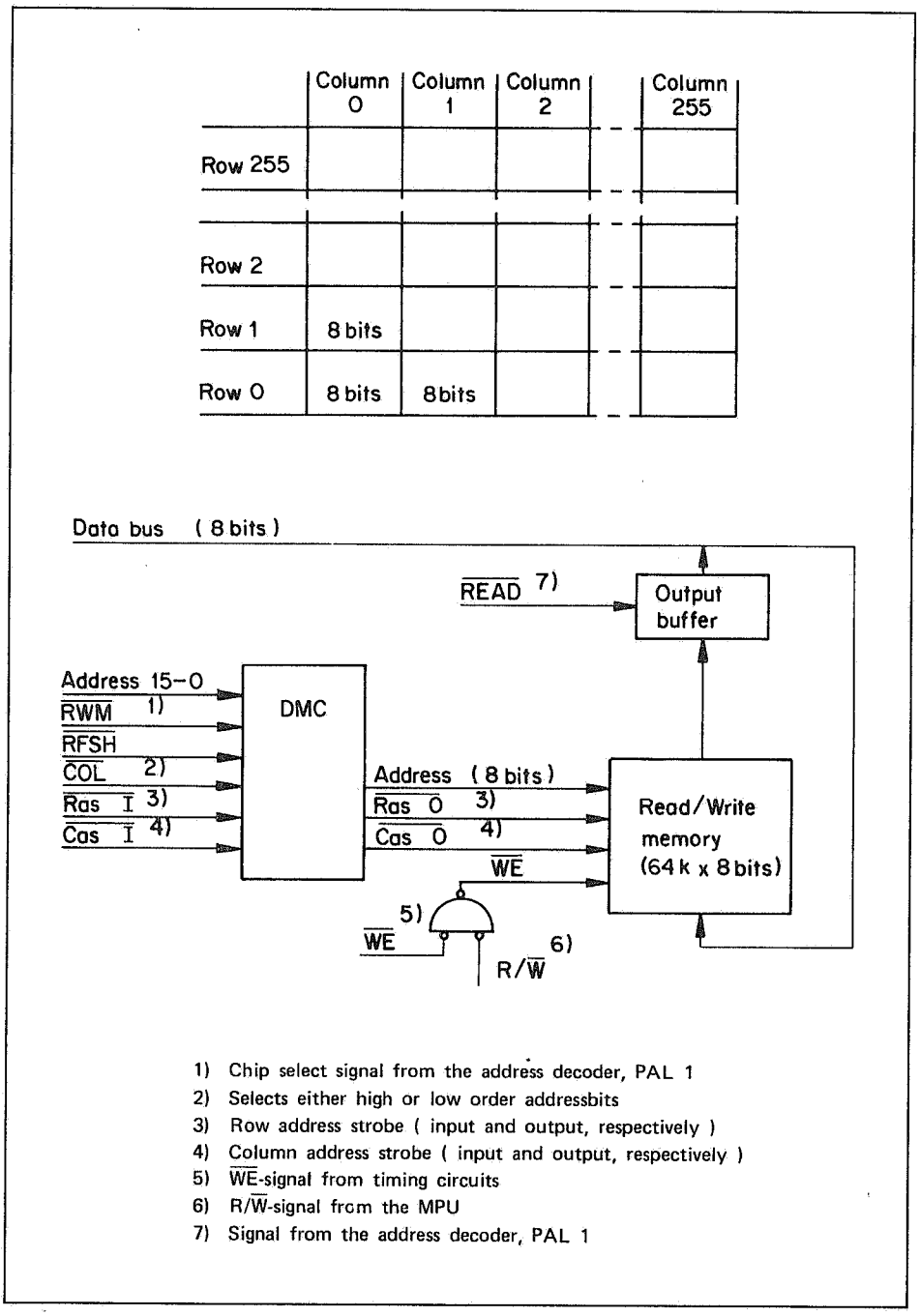


Fig. 7. Principles of read/write memory addressing

While RFSH is active the memory is refreshed, one row per cycle. The row address is the inverted contents of a counter in the DMC. This counter is advanced towards the end of the refresh period by a transition to the inactive state of RAS I. When RFSH is inactive, the memory may be accessed from either the MPU or the DMAC.

The COL-signal determines whether the high- or low-order address-bits are connected to the DMC output address pins. RFSH active, however, disconnects the input address-bits from the outputs. The RAS and CAS outputs follow the inputs RAS I and CAS I.

In the beginning of the non-refresh period (RFSH inactive) COL is inactive, which connects the low-order address-bits. The row address is then strobed by RAS I, RAS. When COL goes active, the high-order address-bits are connected. The column address is then strobed by CAS I, CAS (see Fig. 8).

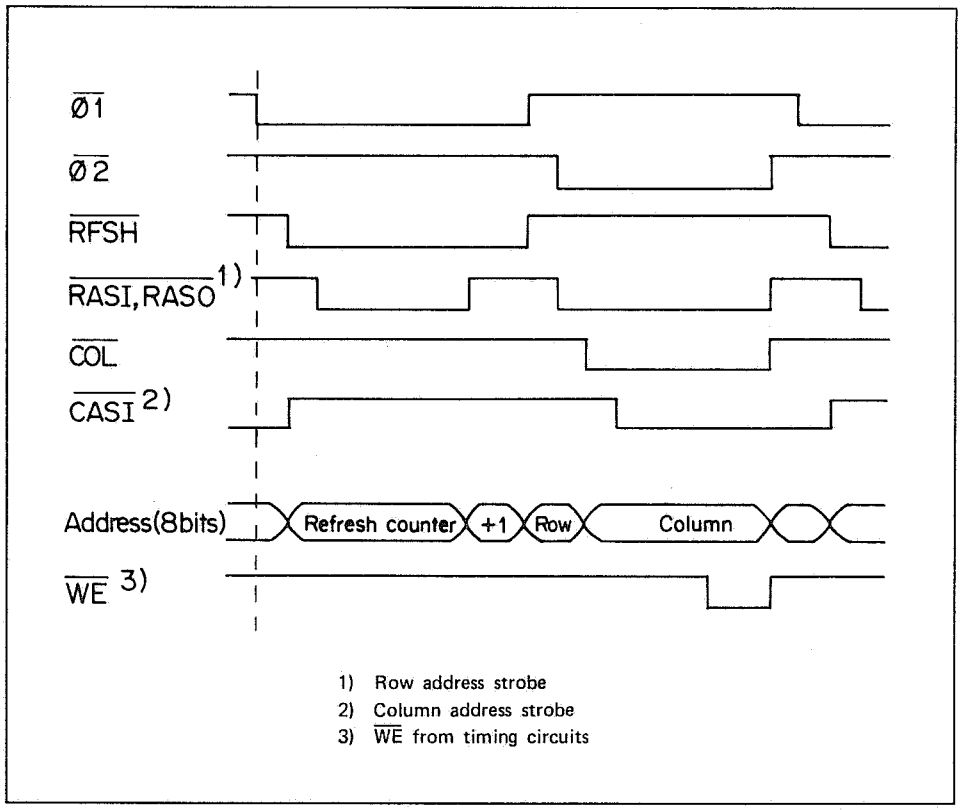


Fig. 8. Address timing

Two-wire Interface

The two-wire interface can work in MPU mode or DMA (direct memory access) mode. The MPU mode is used when the direct memory access controller, DMAC, and the advanced data link controller, ADLC, circuits are initialized or supervised by the MPU. DMA mode is used at DMA transfers.

The main circuits and signals of the two-wire interface are shown in Fig. 9. Please refer to the figure, when reading the following text.

Direct Memory Access Controller, DMAC

The DMAC handles DMA transfers between the basic read/write memory and the ADLC (see below). For definitions and a general survey of the DMAC, see the Microcomputer chapter.

The DMAC is in MPU mode addressed by the DMAC signal from the address decoder and five address bits. The R/W line is also fed to the DMAC. The bus buffers will allow the five address bits and R/W to pass in direction to the DMAC in MPU mode.

The DMAC will at a service request from the ADLC (RDSR or TDSR active) make a DMA request and when this is accepted (DMA grant active) go to DMA mode (with TXSTB active at DMA cycle 2). The DMAC then provides a memory address from an internal address counter, a R/W signal (to the ordinary R/W line) and control signals to the ADLC (TXAKA, producing a read/write signal to the ADLC and DEND at the last byte of a transferred block).

The DMAC is connected for HALT steal mode and two channel mode. Channel 0 is used for transmitting and channel 1 for receiving.

Advanced Data Link Controller, ADLC

The ADLC converts parallel data from the internal data bus to serial data (TXD) for the high speed modem or serial data from the high speed modem (RXD) to parallel data for the data bus. For definitions and a general survey of the ADLC, see the Communication chapter.

The ADLC is programmed to generate a Request to send (RTS) signal to the high speed modem. It may also interrupt the MPU. The two-wire interrupt thus originate from the ADLC – not the DMAC. (The data carrier detect input, DCD, is hard-wired active. The DTR/LOC and FLAGDET outputs are not used).

It is possible to make transfers of two-wire data via the MPU but up to now only DMA transfers are used because of the high speed.

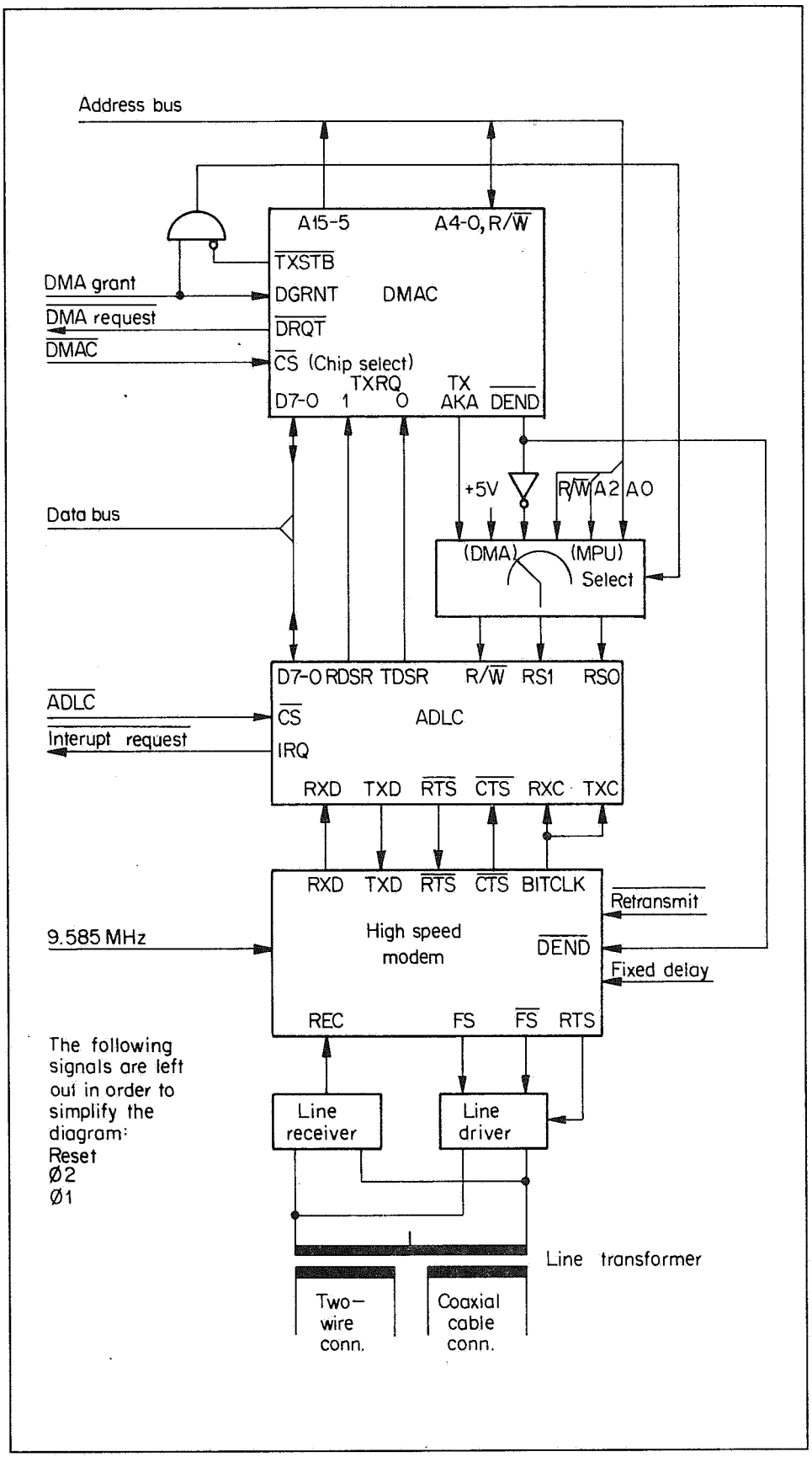


Fig. 9. Principles of two-wire interface

The ADLC is in MPU mode addressed by the ADLC chip select signal from the address decoder and two address bits, A1 – 0. The R/W line is also fed to the ADLC. The ADLC chip select signal from the address decoder is also activated at DMA cycle 2 but the R/W and RS0 inputs are in DMA mode fed from the DMAC and the RS1 is then fed from a hard-wired high. The addressing is summarized below:

Address conditions							Register addressed and read/write function
MPU mode			DMA mode			Address control bit[1]	
R/W (R/W)	A1 (RS1)	A0 (RS0)	TXAKA (R/W)	+5 V (RS1)	DEND (RS0)		
0	0	0		[2]		X	Write control register 1
0	0	1		[2]		0	Write control register 2
0	0	1		[2]		1	Write control register 3
0	1	0	0	1	0	X	Write transmitter FIFO[3]
0	1	1	0	1	1	0	Write transmitter FIFO[4]
0	1	1	0	1	1	1	Write control register 4
1	0	0		[2]		X	Read status register 1
1	0	1		[2]		X	Read status register 2
1	1	X	1	1	X	X	Read receiver FIFO

[1] Control register 1 bit 0. [2] 0 is not possible. [3] Frame continue. [4] Frame terminate.

High Speed Modem

The high speed digital modem consists of one chip, which contains a modulator, a demodulator, a frequency divide and phase correction logic, and a delay circuit for Clear to send. See Fig. 10.

Data to be transmitted (TXD, from the ADLC) is presented in serial format to the modulator. The modulator converts the signal to a frequency shifted signal (FS), which then is fed to the two-wire if Request to send (RTS) is active.

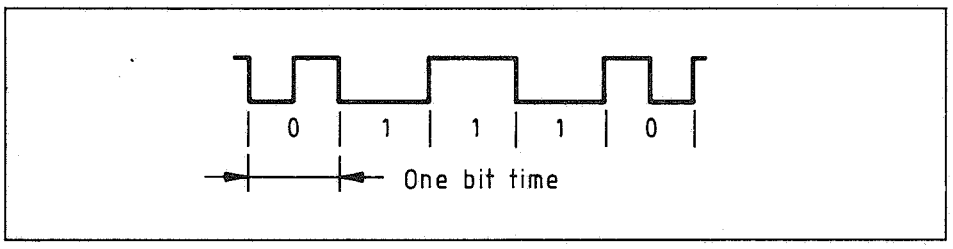
The frequency shifted data transferred from the two-wire (REC) is fed to the demodulator and to the frequency divide and phase correction logic. An internal strobe from the latter logic shifts the frequency shifted received data into the demodulator and generates a bit clock (BITCLK), fed to the ADLC. The demodulator converts the received data to demodulated valid data (RXD fed to the ADLC). (Received data is not, as Retransmit is hard-wired high, fed to the modulator and retransmitted).

The modulation method, clock generation and phase correction are described in the Communication chapter.

The Request to send output (RTS) follows (but inverted) the Request to send input (RTS). (It is also a function of received data in the not used retransmit mode).

Clear to send (CTS) follows the Request to send input (RTS) with a delay of 32 (controlled by strapping of the fixed delay input) bit clock periods. This delay will enable the modem in the receiving unit (e.g. in a CPR) to synchronize before the transmission is started. A frequency corresponding to the state of the transmitted data input (TXD) will be sent on the FS lines during the delay. An active DEND signal (at the end of a transmission sequence) from the DMAC will deactivate Clear to send.

Frequency shifted received data is validated, i.e. each bit must consist of an LH, HL, LL or HH and there must be a shift of polarity between the bits in order to be accepted:



Received data (RXD) is always high when received data is invalid or before the modem has synchronized.

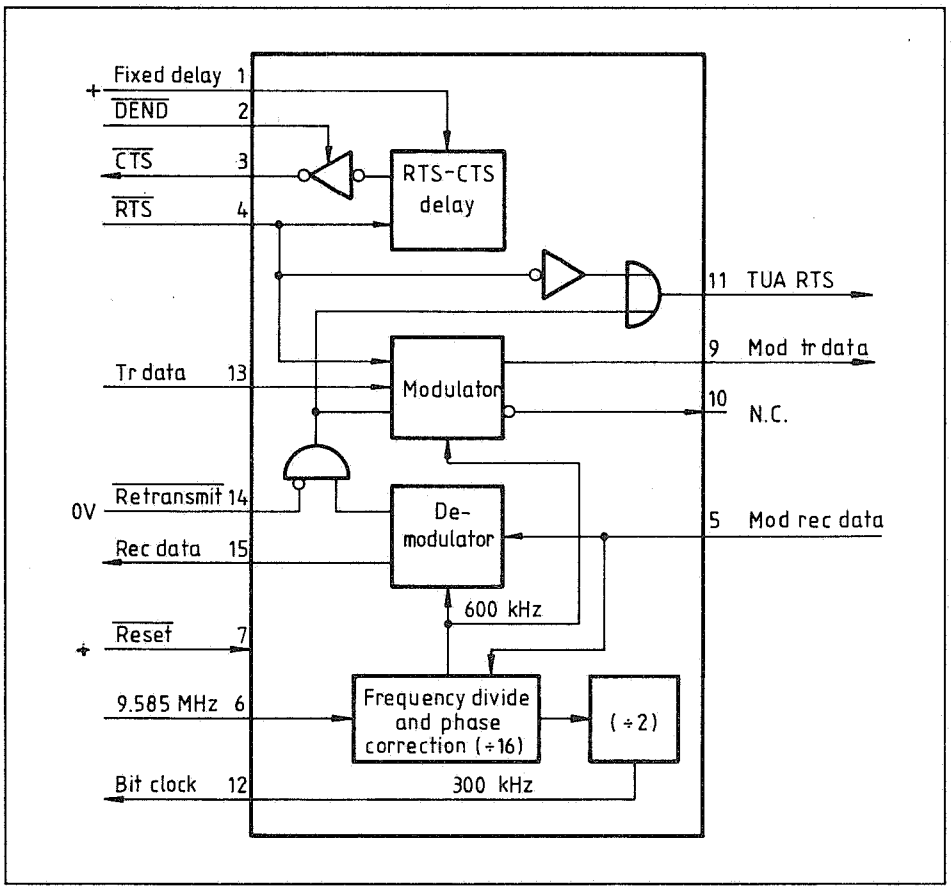


Fig. 10. High speed modem circuit

Line Interface

The high speed modem is connected to a line transformer via a line driver and a line receiver. The line transformer has separate windings for a two-wire and a coaxial cable connection.

Transmitting

To initiate a block transmission, using DMA, the program must prepare itself to answer an interrupt from the ADLC (interrupt 4) when the whole block (frame) has been transmitted. The DMAC channel 0 address register, byte count register and control register should also be programmed with:

- Start address of DMA, i.e. the memory address of the first byte (except flag) to be transmitted on the two-wire connection.
- Number of memory bytes to be transmitted.
- Order to count up or down when addressing the memory.
- Order to produce the read state on the R/W line when addressing the memory.
- HALT steal mode.
- Set Enable TXRQ 0 in the DMAC priority control register in order to make the DMAC react to TDSR (Transmit data service request) from the ADLC.

The ADLC must be programmed for:

- Correct word length (8 bits)
- NRZ data format
- Non-loop mode
- Transmitter DMA mode etc.

The transmitting section of the ADLC is released and the request to send (RTS) control bit is set by the program. After a delay Clear to send (CTS) is signalled back to the ADLC, and thus the Transmitter data service request (TDSR) output from the ADLC is no longer inhibited.

The following will happen as an answer to a TDSR signal:

- A DMA request signal (DRQT) will be sent to the HALT input of the MPU.
- The MPU will finish the current instruction and then be halted and answer with a DMA grant signal (DGRNT) from the bus available (BA) output.
- The DMAC will then produce a low level on the transfer acknowledge A (TXAKA) output as channel 0 is used. This will ensure a write strobe for the ADLC when;

  – The DMA mode signal is activated by the Transfer strobe (TXSTB).

In the same time the ADLC is enabled, its transmit data register is addressed and a memory cell is addressed by the DMAC as the drivers for the address lines from the DMAC are enabled.

After a direct memory access the byte is shifted to the high speed modem. Thus it takes some time before the transmit data register is again available. The microcomputer is consequently undisturbed by any DMA request during this time.

When the DMA transfers (consisting of the number of bytes defined in the DMAC byte count register) are being completed, i.e. when the last byte is being fetched from the memory, the DEND output of the DMAC is activated. This will result in:

- A termination of the frame (write into the "transmit terminate FIFO" register; the ADLC adds CRCC and flag).
- A deactivation of the Clear to send (CTS) signal. The ADLC will react by not signalling any more TDSR.

The non-CTS signal causes an interrupt. As an answer, the program resets the RTS signal, clears status bits in the ADLC and prepares the DMAC and the ADLC for reception.

Receiving

Preparatory programming of the DMAC and the ADLC for receiving operations resembles the one described in the Transmitting paragraph. Furthermore, the program sees to it that an interrupt will be answered by a program check of receiver status etc.

When bytes following a leading flag are received, the ADLC sends receiver data service request (RDSR) to the DMAC. These bytes will then be stored in the memory according to DMAC channel 1 programming. The DMA transfer is terminated as a response to an interrupt caused by error (framing error, abort received or receiver overrun) or by frame valid status. The program then resets the ADLC and takes care of the received data on interrupt 0 level. Note that Request to send must of course be reset during reception.

Appendix 1

I/O Addresses, $F7FF_{(16)} - F700_{(16)}$

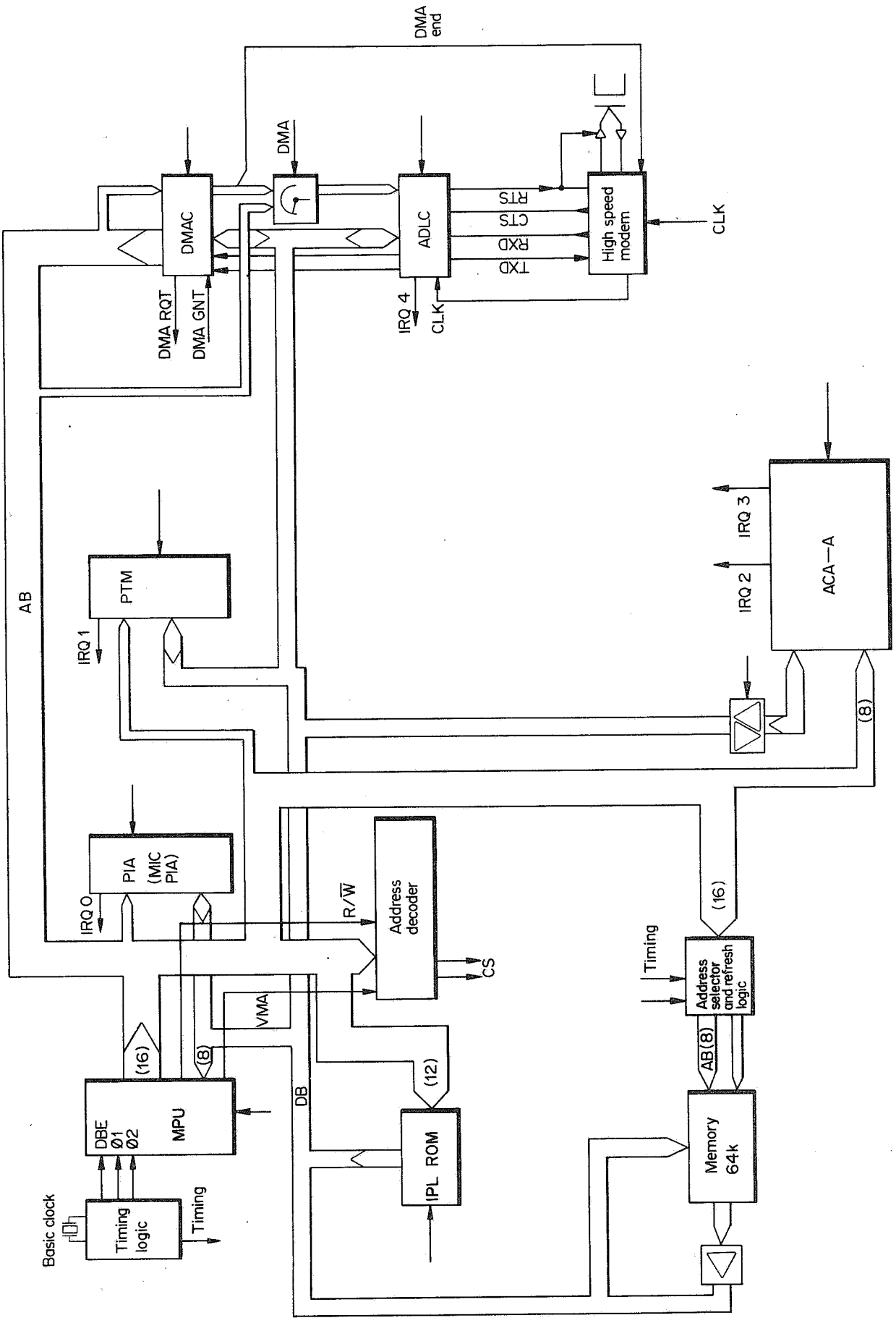
(The first two figures, $F7_{(16)}$, are omitted in the address column below. R/W = W means write, R/W = R means read and R/W = R/W means read or write as seen from the MPU.)

Address (Hex)	R/W	Other condition	Addressed unit and register
			MCP
FF	W		STORE (bit 7), CCR, ACCB, ACCA, XHI, XLO, PCHI and PCLO switches
FE	W		Data 7–0 switches
FD	W		Address 15–8 switches
FC	W		Address 7–0 switches
FB	R		A, B, C, D, NMI, IRQ, VMA and R/W LEDs
FA	R		Data 7–0 LEDs
F9	R		Address 15–8 LEDs
F8	R		Address 7–0 LEDs
F7–D0			**Not used**
			MIC PTM
CF	W		Timer 3 LSB latches
CF	R		LSB buffer (timer 3)
CE	W		MSB buffer (timer 3)
CE	R		Timer 3 counter MSB
CD	W		Timer 2 LSB latches
CD	R		LSB buffer (timer 2)
CC	W		MSB buffer (timer 2)
CC	R		Timer 2 counter MSB
CB	W		Timer 1 LSB latches
CB	R		LSB buffer (timer 1)
CA	W		MSB buffer (timer 1)
CA	R		Timer 1 counter MSB
C9	W		Control register 2 (CR2)
C9	R		Status register
C8	W	CR2:0 = 0	Control register 3
C8	W	CR2:0 = 1	Control register 1
			MIC PIA
C7	R/W		Control register B (CRB)
C6	R/W	CRB2 = 0	Data direction register B
C6	R/W	CRB2 = 1	Peripheral register B
C5	R/W		Control register A (CRA)
C4	R/W	CRA2 = 0	Data direction register A
C4	R/W	CRA2 = 1	Peripheral register A
C3–A8			**Not used**
			ACA-A ACIA (Modem interface)
A7	→		= A1
A6	→		= A0
A5	→		= A1
A4	→		= A0
A3	→		= A1
A2	→		= A0
A1	W		Transmitted data register
A1	R		Received data register
A0	W		Control register
A0	R		Status register

Address (Hex)	R/W	Other condition	Addressed unit and register
			ACA-A PTM (Modem interface)
9F	W		Timer 3 LSB latches
9F	R		LSB buffer (timer 3)
9E	W		MSB buffer (timer 3)
9E	R		Timer 3 counter MSB
9D	W		Timer 2 LSB latches
9D	R		LSB buffer (timer 2)
9C	W		MSB buffer (timer 2)
9C	R		Timer 2 counter MSB
9B	W		Timer 1 LSB latches
9B	R		LSB buffer (timer 1)
9A	W		MSB buffer (timer 1)
9A	R		Timer 1 counter MSB
99	W		Control register 2 (CR2)
99	R		Status register
98	W	CR2:0 = 0	Control register 3
98	W	CR2:0 = 1	Control register 1
			ACA-A PIA (Modem interface)
97	R/W		= 53
96	R/W	→	= 52
95	R/W		= 51
94	R/W	→	= 50
93	R/W		Control register B (CRB)
92	R/W	CRB2 = 0	Data direction register B
92	R/W	CRB2 = 1	Peripheral register B
91	R/W		Control register A
90	R/W	CRB2 = 0	Data direction register A
90	R/W	CRB2 = 1	Peripheral register A
8F – 68			**Not used**
			ACA-A ACIA (Printer interface)
67	→		= 61
66	→		= 60
65	→		= 61
64	→		= 60
63	→		= 61
62	→		= 60
61	W		Transmitted data register
61	R		Received data register
60	W		Control register
60	R		Status register
			ACA-A PTM (Printer interface)
5F	W		Timer 3 LSB latches
5F	R		LSB buffer (timer 3)
5E	W		MSB buffer (timer 3)
5E	R		Timer 3 counter MSB
5D	W		Timer 2 LSB latches
5D	R		LSB buffer (timer 2)
5C	W		MSB buffer (timer 2)
5C	R		Timer 2 counter MSB
5B	W		Timer 1 LSB latches
5B	R		LSB buffer (timer 1)
5A	W		MSB buffer (timer 1)
5A	R		Timer 1 counter MSB
59	W		Control register 2 (CR2)
59	R		Status register
58	W	CR2:0 = 0	Control register 3
58	W	CR2:0 = 1	Control register 1

Address (Hex)	R/W	Other condition	Addressed unit and register
			ACA-A PIA (Printer interface)
57	R/W		= 53
56	R/W	→	= 52
55	R/W		= 51
54	R/W	→	= 50
53	R/W		Control register B (CRB)
52	R/W	CRB2 = 0	Data direction register B
52	R/W	CRB2 = 1	Peripheral register B
51	R/W		Control register A
50	R/W	CRB2 = 0	Data direction register A
50	R/W	CRB2 = 1	Peripheral register A
4F–28			**Not used**
			ADLC
			(Can also be addressed by the DMAC at DMA)
27	→	→	= 23
26	→		= 22
25	→	→	= 21
24	→		= 20
23	W	CR1:0 = 0	Transmitter FIFO (terminate)
23	W	CR1:0 = 1	Control register 4
23	R		Receiver FIFO
22	W		Transmitter FIFO (continue)
22	R		Receiver FIFO
21	W	CR1:0 = 0	Control register 2
21	W	CR1:0 = 1	Control register 3
21	R		Status register 2
20	W		Control register 1 (CR1)
20	R		Status register 1
1F–17			**Not used**
			DMAC
16	R/W		Data chain
15	R/W		Interrupt control
14	R/W		Priority control
13	R/W		Channel 3 control
12	R/W		Channel 2 control
11	R/W		Channel 1 control
10	R/W		Channel 0 control
0F	R/W		Byte count, channel 3 (LSB)
0E	R/W		Byte count, channel 3 (MSB)
0D	R/W		Address, channel 3 (LSB)
0C	R/W		Address, channel 3 (MSB)
0B	R/W		Byte count, channel 2 (LSB)
0A	R/W		Byte count, channel 2 (MSB)
09	R/W		Address, channel 2 (LSB)
08	R/W		Address, channel 2 (MSB)
07	R/W		Byte count, channel 1 (LSB)
06	R/W		Byte count, channel 1 (MSB)
05	R/W		Address, channel 1 (LSB)
04	R/W		Address, channel 1 (MSB)
03	R/W		Byte count, channel 0 (LSB)
02	R/W		Byte count, channel 0 (MSB)
01	R/W		Address, channel 0 (LSB)
00	R/W		Address, channel 0 (MSB)

Appendix 2


DMA end
DMA
DMAC
ADLC
High speed modem
RTS
CTS
RXD
TXD
CLK
DMA RQT
DMA GNT
IRQ 4
CLK
AB
PTM
IRQ 1
IRQ 3
IRQ 2
ACA—A
(8)
PIA (MIC PIA)
IRQ 0
R/W
Address decoder
CS
(16)
Timing
Address selector and refresh logic
VMA
(16)
(8)
DB
(12)
AB(8)
DBE
Ø1
Ø2
MPU
IPL ROM
Memory 64k
Basic clock
Timing logic
Timing

DATASAAB

Synchronous Communication Adapter

Contents

Figures

EE364-810
1978-11-24

General

The synchronous communication adapter (SCA) board is used in a single display unit or in a Communication Processor, Remote 4101 when synchronous serial communication with e.g. a host computer via a modem (remote connection) is wanted. The SCA is connected to the external address and data buses of the microcomputer and its registers are located in the I/O address area of the memory map. It also provides one interrupt (IRQ) to the microcomputer interrupt logic, (IRQ 4 for DU 4110, IRQ 3 for CPR 4101) used to request byte transfers or to signal errors or changes in the state of the communication equipment.

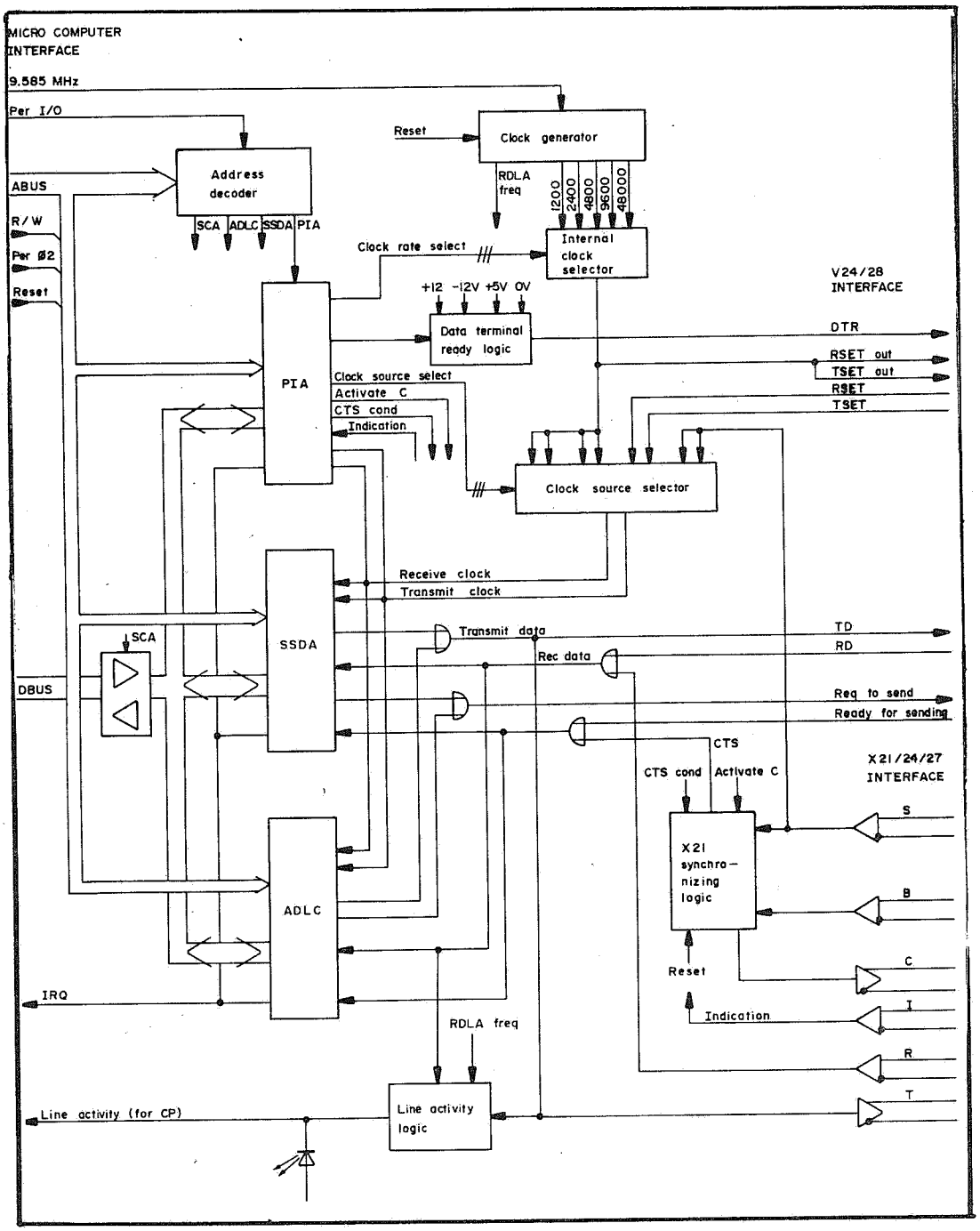


Fig. 1. Block diagram of synchronous communication adapter

Communication may be made using the BSC or the SDLC protocol. BSC (Binary synchronous communication) is made via the SSDA (Synchronous serial data adapter) IC. For SDLC (Synchronous data link control) the circuit ADLC (Advanced data link controller) is used.

As can be seen in the SCA block diagram (Fig. 1), SSDA and ADLC are interfacing the modem with common lines (serial receive and transmit data lines etc).

The ADLC is described in the Microcomputer chapter, section Two-wire Communication. The most important parts of the SSDA will be described below. For a detailed description of the PIA, see "Microcomputer", section "PIA".

Note that the SCA may also be used for connection to the Nordic public data net (NPDN). In that case both SSDA and ADLC should be installed (X21/24/27 interface).

Addressing

The location of an SCA register in the microcomputer memory map is decided by

- Address decoder on the basic logic board, providing the Per. I/O signal ≈ $F7XX_{16}$
- Address decoder PROM on SCA. This PROM uses the Per. I/O signal and address bits 7 through 3 and provides circuit enables according to the table below. Note that the SCA signal is activated for all SCA addresses (F768 - F77F).
- The address lines tied to the actual integrated circuit. Address bits 1 and 0 or only bit 0 is used.

Table 1. SCA register addressing

Enabled circuit	Address (hex) 1)	2)	Accessed register	
PIA	F768	(F76C)	PIAA: PIA A-side per reg. (Accessed if PIACRA: 2 = 1)	PIADDRA: Data direction reg. (PIACRA: 2 = 0)
	F769	(F76D)	PIACRA: PIA A-side control register	
	F76A	(F76E)	PIAB (if PIACRB: 2 = 1)	PIADDRB (if PIACRB: 2 = 0)
	F76B	(F76F)	PIACRB: PIA B-side control register	
			Write (R/W = 0)	Read (R/W = 1)
SSDA	F770	(2,4,6)	Control reg. 1	Status reg.
	F771	(3,5,7)	Control reg. 2 (CR1:7,6 = 0 0) Control reg. 3 (0 1) Sync code reg. (1 0) Transmit data reg. (1 1)	Receive data register
ADLC	F778	(F77C)	Control reg. 1	Status reg. 1
	F779	(F77D)	Control reg. 2 (CR1:0 = 0) Control reg. 3 (1)	Status reg. 2
	F77A	(F77E)	Transmit FIFO, cont	Receive FIFO
	F77B	(F77F)	Transmit FIFO, term (0) Control reg. 4 (1)	Receive FIFO

Notes 1) The Per. I/O signal is activated for addresses≈ $F7XX_{16}$

2) The double (or multiple) register addresses are caused by address bit 2 (and bit 1) not being decoded.

Synchronous Serial Data Adapter, SSDA

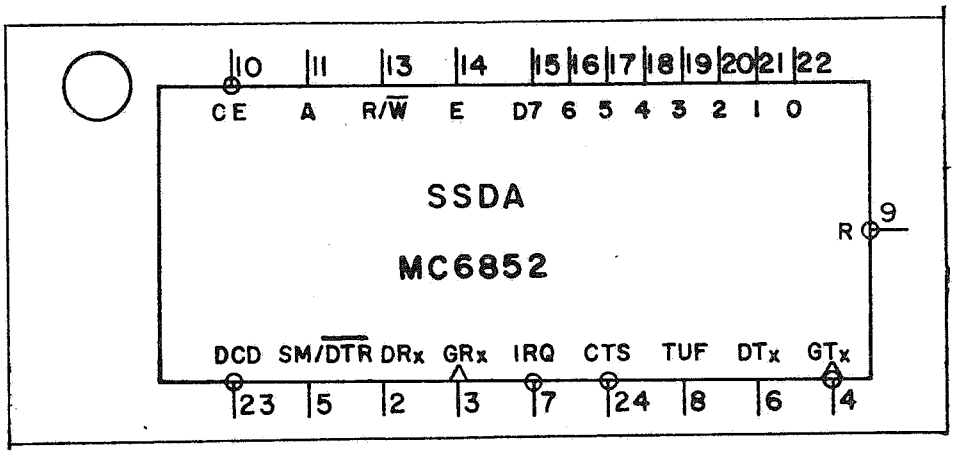


Fig. 2. SSDA drawing symbol

The SSDA contains two read-only registers (Status register and Receive data FIFO) and five write-only registers (Three control registers, a sync. code register and a transmit data FIFO). The addressing of these is made with the Chip enable (CE) line, the E ($\Phi 2$) line, also used to transfer data through the internal 3-byte receive and transmit data FIFO:s, the R/$\overline{W}$ line and the Address line (A). For the last four registers, the state of the two address control bits of CR 1, together with the external addressing lines mentioned above, define which register that is accessed. Apart from these lines, the SSDA interface to the MPU consists of the eight bit data bus (D7 through 0), the Reset line, which will reset the SSDA by way of the two control bits Receiver reset and Transmitter reset, and the Interrupt request line (IRQ).

The peripheral or modem interface consists of the following lines.

Receiver section:

D_{Rx} : Bit-serial receive data input

G_{Rx} : Receiver bit clock input

DCD: Data carrier detected input. A low to high transition on this input (e.g. indicating loss of data carrier) will be stored in the status register and may cause an interrupt (IRQ). It will result in an inhibition of the receiver section apart from the receive data FIFO and the RDA (Receive data available) status bit

Transmitter section:

D_{Tx} : Bit-serial transmit data output

G_{Tx} : Transmitter bit clock

CTS : Clear to send input. A positive edge on this input (indicating $\overline{\text{Clear-to-send}}$) will be stored in the status register and may cause an IRQ. As a result, the transmitter section of the SSDA is inhibited. TDRA (Transmit data reg. available) status bit is however not affected in the external sync. mode.

Table 2. SSDA programming model

Register	Control Inputs A	R/W	Address Control AC2	AC1	Register Content Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Status (S)	0	1	X	X	Interrupt Request (IRQ)	Receiver Parity Error (PE)	Receiver Overrun (Rx Ovrn)	Transmitter Underflow (TUF)	Clear-to-Send (CTS)	Data Carrier Detect (DCD)	Transmitter Data Register Available (TDRA)	Receiver Data Available (RDA)
Control 1 (C1)	0	0	X	X	Address Control 2 (AC2)	Address Control 1 (AC1)	Receiver Interrupt Enable (RIE)	Transmitter Interrupt Enable (TIE)	Clear Sync	Strip Sync Characters (Strip Sync)	Transmitter Reset (Tx Rs)	Receiver Reset (Rx Rs)
Receive Data FIFO	1	1	X	X	D7	D6	D5	D4	D3	D2	D1	D0
Control 2 (C2)	1	0	0	0	Error Interrupt Enable (EIE)	Transmit Sync Code on Underflow (Tx Sync)	Word Length Select 3 (WS3)	Word Length Select 2 (WS2)	Word Length Select 1 (WS1)	1-Byte/2-Byte Transfer (1-Byte/2-Byte)	Peripheral Control 2 (PC2)	Peripheral Control 1 (PC1)
Control 3 (C3)	1	0	0		Not Used	Not Used	Not Used	Not Used	Clear Transmitter Underflow Status (CTUF)	Clear CTS Status (Clear CTS)	One-Sync-Character/ Two-Sync Character Mode Control (1 Sync/ 2 Sync)	External/ Internal Sync Mode Control (E/I Sync)
Sync Code	1	0	1	0	D7	D6	D5	D4	D3	D2	D1	D0
Transmit Data FIFO	1	0	1	1	D7	D6	D5	D4	D3	D2	D1	D0

X = Don't care

STATUS REGISTER

IRQ Bit 7 The IRQ flag is cleared when the source of the IRQ is cleared. The source is determined by the enables in the Control Registers: TIE, RIE, EIE.

Bits 6-0 indicate the SSDA status at a point in time, and can be *reset* as follows:

PE Bit 6 Read Rx Data FIFO, or a "1" into Rx Rs (C1 Bit 0).

Rx Ovrn Bit 5 Read Status and then Rx Data FIFO, or a "1" into Rx Rs (C1 Bit 0).

TUF Bit 4 A "1" into CTUF (C3 Bit 3) or into Tx Rs (C1 Bit 1).

$\overline{CTS}$ Bit 3 A "1" into Clear $\overline{CTS}$ (C3 Bit 2) or a "1" into Tx Rs (C1 Bit1).

$\overline{DCD}$ Bit 2 Read Status and then Rx Data FIFO or a "1" into Rx Rs (C1 Bit 0)

TDRA Bit 1 Write into Tx Data FIFO.

RDA Bit 0 Read Rx Data FIFO.

CONTROL REGISTER 1

AC2, AC1 Bits 7, 6 Used to access other registers, as shown above.

RIE Bit 5 When "1", enables interrupt on RDA (S Bit 0).

TIE Bit 4 When "1", enables interrupt on TDRA (S Bit 1).

Clear Sync Bit 3 When "1", clears receiver character synchronization.

Strip Sync Bit 2 When "1", strips all sync codes from the received data stream.

Tx Rs Bit 1 When "1", resets and inhibits the transmitter section.

Rx Rs Bit 0 When "1", resets and inhibits the receiver section.

CONTROL REGISTER 3

CTUF Bit 3 When "1", clears TUF (S Bit 4), and IRQ if enabled.

Clear $\overline{CTS}$ Bit 2 When "1", clears $\overline{CTS}$ (S Bit 3), and IRQ if enabled.

1 Sync/2 Sync Bit 1 When "1", selects the one-sync-character mode; when "0", selects the two-sync-character mode.

E/I Sync Bit 0 When "1", selects the external sync mode; when "0", selects the internal sync mode.

CONTROL REGISTER 2

EIE Bit 7 When "1", enables the PE, Rx Ovrn, TUF, $\overline{CTS}$, and $\overline{DCD}$ interrupt flags (S Bits 6 through 2).

Tx Sync Bit 6 When "1", allows sync code contents to be transferred on underflow, and enables the TUF Status bit and output. When "0", an all mark character is transmitted on underflow.

WS3, 2, 1 Bits 5-3 Word Length Select

Bit 5 WS3	Bit 4 WS2	Bit 3 WS1	Word Length
0	0	0	6 Bits + Even Parity
0	0	1	6 Bits + Odd Parity
0	1	0	7 Bits
0	1	1	8 Bits
1	0	0	7 Bits + Even Parity
1	0	1	7 Bits + Odd Parity
1	1	0	8 Bits + Even Parity
1	1	1	8 Bits + Odd Parity

1-Byte/2-Byte Bit 2 When "1", enables the TDRA and RDA bits to indicate when a 1-byte transfer can occur; when "0", the TDRA and RDA bits indicate when a 2-byte transfer can occur.

PC2, PC1 Bits 1-0 SM/$\overline{DTR}$ Output Control

Bit 1 PC2	Bit 0 PC1	SM/$\overline{DTR}$ Output at Pin 5
0	0	1
0	1	Pulse ⎍ 1-Bit Wide, on SM
1	0	0
1	1	SM Inhibited, 0

NOTE: When the SSDA is used in applications requiring the MSB of data to be received and transmitted first, the data bus inputs to the SSDA may be reversed (D0 to D7, etc.). Caution must be used when this is done since the bit positions in this table will be reversed, and the parity should not be selected.

The SM/$\overline{\text{DTR}}$ (Sync. match/$\overline{\text{Data terminal ready}}$) output may be used for several purposes. When CR2:0 is 0 it presents the inverse of what is written into CR2:1. This may be used to signal DTR or RTS (Request to send). When CR2:0 = 1 and CR2:1 = 0 it will present a positive pulse when a sync. code equal to the one stored in the sync. code register is detected (see Table 2).

V24/28 Interface

Interface Signals

The V24/28 interface contains:

- Receive data input (RD). This line is tied both to SSDA and ADLC. Which of these circuits that will take care of the incoming data depends on programming.
- Transmit data output (TD), presenting serial data from SSDA or ADLC.
- Receive bit clock (RSET, Receiver signal element timing) and transmit bit clock (TSET) from external equipment.
- RSET out and TSET out: Bit clocks (from the same source) generated on SCA.
- Request to send (RTS). This signal is generated by ADLC (RTS output) or SSDA (SM/$\overline{\text{DTR}}$ output) to prepare the data circuit-terminating equipment, DCE (typically modem), for transmission to the line from SCA.
- Ready for sending (RFS) is the signal by which the DCE (modem) answers that it is ready to (modulate and) send what it gets on the transmit data line from SCA. This signal is tied to the CTS (Clear to send) inputs of SSDA and ADLC.
- Data terminal ready (DTR) may e.g. be used as an enable signal for the modem, telling the modem that the data terminal equipment is operable. It is generated when all supply voltages for SCA are present and the program signals "ready" by writing a zero into PIA A-side peripheral reg. bit 7 (at address $F768_{16}$).

Clock Generation

For the V24/28 interface, the bit clock may be selected (by two PIA bits) to be external or internal. Internal clocks are provided by the clock generator on SCA. Via three PIA bits, the internal bit clock rate is selected; 1200, 2400, 4800, 9600 or 48.000 bits per second. For details, see "PIA Programming" paragraph below.

X21/24/27 Interface

The X21/24/27 interface comprises balanced, two-wire connections. A positive differential voltage ($V_A - V_B > 0.3$ V) signifies the ON state of a control signal or, for data circuits, a logic 0; a negative differential voltage ($V_A - V_B < -0.3$ V) signifies the OFF state or a logic 1 respectively.

Interface Signals

The bit clock always originates from the DCE (Data circuit-terminating equipment). It is called

- S Signal element timing. The OFF to ON transition of this signal coincides with the transitions (bit limits) of data. Transmit data is thus shifted out from SSDA/ADLC at the OFF to ON transitions. Receive data is shifted into SSDA/ADLC at the ON to OFF transitions of S, in the middle of data bits.

- B Byte timing (from DCE). This signal is used to synchronize the setting of the C (Control) signal to a certain time in the byte and to start transmission from SSDA or ADLC byte-synchronously with the DCE, by setting the CTS (Clear to send) signal.

- C Control is e.g. used to signal to the DCE

		C transition
that	1) I want to establish a connection (make a call)	OFF → ON
	2) I want to clear the connection	ON → OFF
or that	1) I accept a call	OFF → ON
	2) I confirm a clear connection indication from DCE	ON → OFF

  Control is generated by setting the Activate C signal (PIA B-bit 7 = 0). At next OFF to ON transition of B, followed by an OFF to ON transition of S, C will go to the ON state.
  Control is deactivated at the OFF to ON transition of S following a setting of the PIA B-side bit 7.

- I Indication is used for signalling from the DCE, e.g.

		I transition
that	1) The connection I wanted is established	OFF → ON
	2) DCE confirms my "clear connection" request	ON → OFF
or that	1) Connection wanted by "other end" is established	OFF → ON
	2) "Other end" wants to clear the connection	ON → OFF

  As the connection/disconnection sequences take some time, the I signal tells the microcomputer software the changes in state by interrupting the MPU. (I is tied to CB1 and CB2 inputs of PIA.)

- R are the bit-serial receive data lines (R(A) and R(B))

- T are the bit-serial transmit data lines

Note that both for R and T, a positive differential voltage (e.g. between R(A) and R(B)) corresponds to a low level on SSDA or ADLC, where positive logic is at hand.

Note that the CTS signal for SSDA and ADLC is not generated by the DCE, but is activated at the ON to OFF transition of B following an activation of the CTS condition signal (PIA B-bit 6 = 0). For PIA programming details, see below.

Line Activity Indication

The SCA contains logic to drive a light-emitting diode (LED) on top of the logic board. This LED indicates activity on the receive and transmit data lines in the following way.

- Extinguished when there is no activity on either line.
- Glowing steadily when data are transmitted from SCA with no longer pauses between transmissions than 17 s.
- Blinking when the pauses between transmissions are longer than 17 s and data are received. The blink frequency of about 7.5 Hz is provided by the clock generator.

Detailed Description

The line activity logic comprises two retriggerable monostable flip-flops, active during 275 ms after the last triggering (1 to 0 transition on the respective data line). When the transmit data disappear, a timeout counter will be started, providing a reset for the TDLA (Transmit data line activity) signal after 17 s, unless new data have been transmitted within that time. In the latter case the timeout counter is reset etc.

Note that the signal driving the line activity LED is distributed to the CPB (Communication processor board) when SCA is installed in CPR 4101.

Furthermore the TDLA signal is tied to a PIA input, thus accessible for the software.

PIA Programming

Suppose that the PIA has been reset.

1) Write 1 - - 1 1 1 1 1 into data direction register A (DDRA) at address F768. This means that the lines used for Data terminal ready, clock source selection (2 lines) and clock rate selection (3 lines) will function as outputs.

2) Write 1 1 - - 0 0 0 0 into DDRB at address F76A. Consequently Activate C and CTS condition lines (used for X21) will function as outputs and the four least significant lines will work as test inputs (for Receive clock, Transmit clock, TDLA and Internal clock respectively).

3) Write - - - - - 1 - - into control register A at address F769 to make the A-side peripheral register accessible for the MPU.

4) Write e.g. - - 0 0 1 1 1 1 into control register B at address F76B.

Result:
- 1 — IRQ because of X21 signal I going ON.
- 1 — B-side peripheral register accessible.
- 1 — IRQ via CB2 if X21 signal I goes OFF.
- 0 — PIA CB2 functions as interrupt input.

Summary of Peripheral Register Bit Functions

	Output/Input	If = 0	If = 1
PA 7	Output	Data terminal ready may be signalled	DTR may not be signalled
PA 6	Not used		
PA 5	Not used		
PA 4	Output	X21 clock selected (if PA 3 = 1)	External RSET and TSET selected (if PA 3 = 1)
PA 3	Output	Internal clock selected	External clock selected (See PA 4)
PA 2	Output	000, 001 and 010: No internal clock rate selected. Int. clock = high	
PA 1	Output	011: Int. clock rate 48,000 Hz	
PA 0	Output	100: Int. clock rate 9,600 Hz 101: Int. clock rate 4,800 Hz 110: Int. clock rate 2,400 Hz 111: Int. clock rate 1,200 Hz	

	Output/Input	If = 0	If = 1
PB 7	Output	Activate C signalled	Activate C signal off
PB 6	Output	CTS condition	Non-CTS condition
PB 5	Not used		
PB 4	Not used		
PB 3	Input	Rec. clock for SSDA/ ADLC low	Receive clock high
PB 2	Input	Transm. clock for SSDA/ ADLC low	Transmit clock high
PB 1	Input	No Transmit data line activity	Transmit data line activity
PB 0	Input	Internal clock (RSET out and TSET out) low	Internal clock high

Synchronous Communication Controller

Contents

Figures

GENERAL

The Synchronous Communication Controller (SCC) is an autonomous module in Alfaskop System 41. SCC is controlled by a parent (main) microprocessor in either the communication processor (control unit) or the display unit. SCC carries out all line protocol work needed for a serial, synchronous data connection. This means that SCC relieves the main microprocessor (when connected to systems where there are two host computers for example) of complicated message formatting. This is particularly useful at high transmission rates (greater than 9600 baud).

SCC can be included in a display unit which operates in a single-display configuration. One communication processor can incorporate one or two SCC units.

SCC communicates with the line via a V24/28 or X21/24/27 interface. Communication with the main microprocessor in the communication processor (CP) or the display unit (DU) proceeds via the CP/DU bus and also via a shared memory.

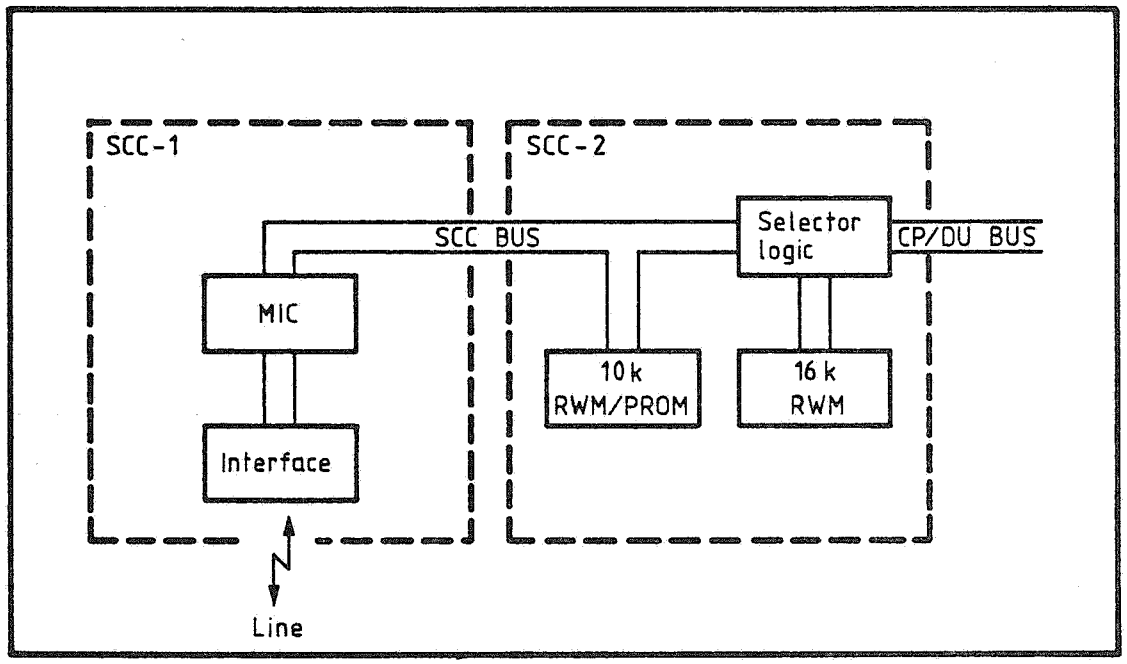


Fig. 1. SCC

SYSTEM OPERATION

Design and Construction

SCC is arranged on two circuit boards designated SCC-1 and SCC-2. See Fig. 1.

The SCC-1 board contains a microprocessor, the IPL-PROM and circuits which control communication with the line or modem.

The SCC-2 board contains a memory area consisting of two parts. The first part can be equipped with either ROM or RWM circuits (10 Kbytes). The second part (16 Kbytes) contains RWM circuits. Part of this RW memory (4 or 8 Kbytes) serves as a buffer memory. That is to say, it can be addressed by both the SCC-processor and the main processor in the communication processor (CP) or display unit (DU). The memory is addressed via the selector logic.

General Operation

- Loading the program
  After the communication processor/display unit (CP/DU) which contains SCC is started up, SCC must be loaded with its system program unless the memory is equipped with ROM circuits in which the system program has been stored. The system program is loaded from the system diskette in the flexible disk unit (FD). It is loaded into the buffer area in the shared memory via the CP/DU-processor. This operation is concluded by having the SCC-processor store the system program in the RW memory.

- Sending a message to the host computer
  When CP/DU is to send information to the host computer, the message is first stored in the buffer memory by CP/DU. An interrupt signal is then sent to the SCC-processor and CP/DU returns to its regular task.

  SCC then processes the message and sends it to the host computer without any further intervention by CP/DU.

- Receiving a message from the host computer
  When a message arrives from the host computer, SCC handles the entire reception procedure without assistance. The message is stored in the buffer memory and SCC then sends an interrupt signal to the main processor in CP/DU. This signal notifies CP/DU-processor that a message is waiting to be fetched from the buffer memory.

Interface with CP/DU

The interface between SCC and CP/DU consists of the CP/DU bus and the part of the SCC memory which serves as a shared memory. Two types of signals are sent via the CP/DU bus, namely interrupts and flag control signals.

- An interrupt can be sent out by SCC and also by CP/DU, each of which are provided with a conductor intended for interrupt signals. When an interrupt is issued, a flip-flop is set in the receiving unit. This flip-flop forwards the interrupt to the internal interrupt logic. It is then the responsibility of the receiving processor to reset the interrupt flip-flop.

- An interrupt indicates to SCC that a command is waiting to be fetched from the shared memory area. The interrupt indicates to CP/DU that a status message is waiting to be fetched from the shared memory area.

- The flag control signal is needed to ensure proper operation when two processors are working in the shared memory area. This function permits one processor to prevent the other processor from accessing the memory area while the first is carrying out an operation there.

Interface with Line

The interface with the line consists of communication circuits used for serial, synchronous transmission. The circuits have been adapted to SCC-microprocessor and include matching amplifiers for the line as well as clock logic and monitoring logic.

Using this logic, SCC can handle all of the synchronization and control signals needed for the line, modem or data circuit terminating equipment (DCE) used for line protocols of the following types:

- Binary Synchronous Communication (BSC)
- Synchronous Data Link Control (SDLC)
- High-Level Data Link Control (HDLC)
- The combination of protocols needed for the Nordic Public Data Network (NPDN)

Moreover, the hardware can carry out the following

- CRC calculation for SDLC and HDLC protocols.

- The sending of extra characters needed in certain protocols.

- The generation of an internal clock pulse for the line.

- Time monitoring of certain signals in connection with sending or monitoring the number of bits sent (maximum of 4 Kbytes).

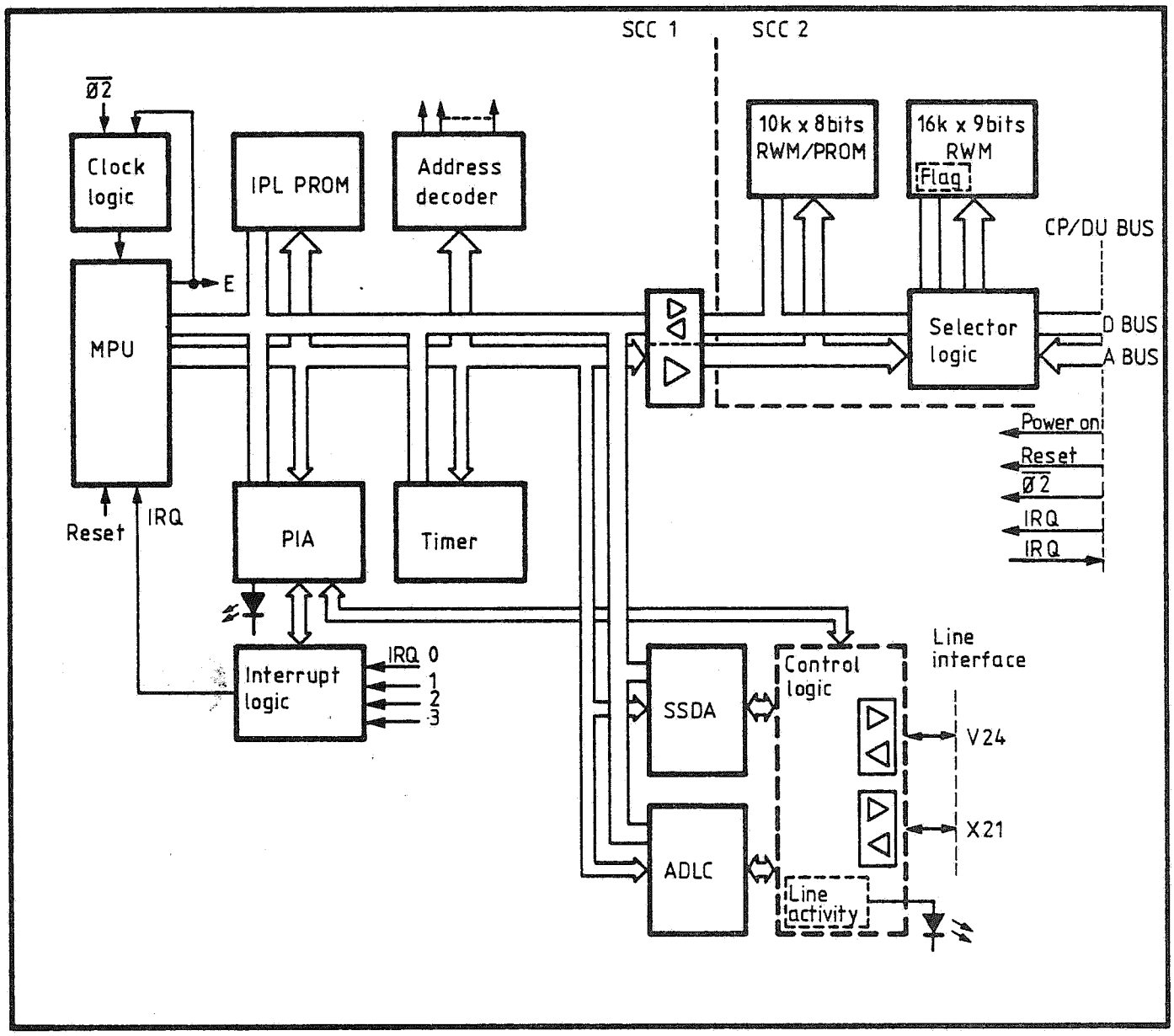


Fig. 2. SCC, block diagram

INTERNAL OPERATION

General

The block diagram shown in Fig. 2 presents an overview of SCC. The description of SCC is divided up on the basis of the following blocks

- SCC-MIC which contains the clock logic, the microprocessor, the address decoder, the timer, PIA and the interrupt logic
- The SCC memory which contains the memory and the selector logic
- The SCC line interface which contains SSDA, ADLC, part of PIA and the control logic used for the interface.

Each block description discusses the circuits which operate within the block and also special block functions. For further information about the circuits that are used, see the manufacturers' data sheets.

SCC-MIC

SCC-MIC, see Fig. 3, contains the microprocessor (MPU). When start-up occurs, this microprocessor is controlled by the initialization program, after which it is controlled by the system program. MPU is clocked by the clock logic which makes a phase correction for the main processor system clock (Ø2). MPU senses interrupt signals obtained from different SCC circuits and from CP/DU. MPU can generate an interrupt signal which is sent to CP/DU via the PIA circuit. Signals are also sent via the PIA circuit to and from the line interface and interrupt logic. MPU controls the programming of the timer and the line-interface circuits (SSDA and ADLC).

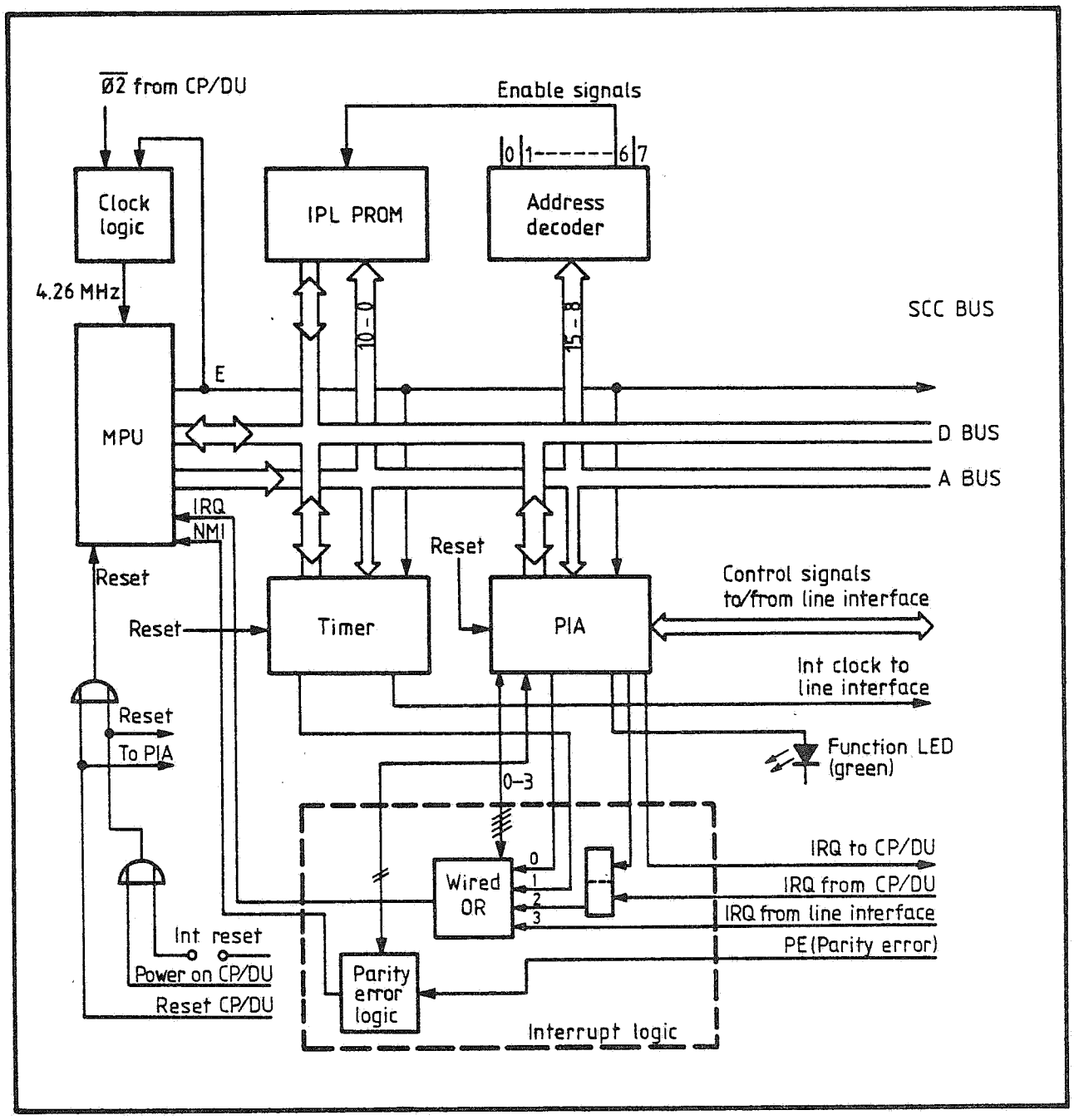


Fig. 3. SCC-MIC, block diagram

Clock Logic

In a system where two microprocessors operate in a shared memory, the phase relationships between the clock pulses in the two systems must be correct. Phase comparison in SCC is handled by the clock logic.

The clock logic contains a phase detector and a voltage controlled oscillator (VCO) which sends a 4.26 MHz signal to MPU. This signal is divided by four in MPU and sent out as a clock signal called Enable (E = 1.065 MHz) to the bus-connected circuits in SCC. Signal E (the resultant phase) is compared with $\overline{\emptyset 2}$ (the reference phase obtained from CP/DU) in the phase detector which controls VCO so that the phase difference between E and Ø2 can be eliminated. See Fig. 4.

MPU 6802

MPU is an 8-bit parallel tri-state device. It has 16 address bits and is thus capable of addressing 65536 memory locations.

MPU 6802 is completely software-compatible with the 6800 MPU described under the Microcomputer thumb index. MPU 6802 contains all the registers and accumulators present in the 6800 plus a built-in 128 byte RWM. Moreover, MPU 6802 contains a clock oscillator and driver for the system clock signal (E).

- Registers

MPU contains the following registers

- Accumulator A and accumulator B (8 bits) used for storing operands and results from operations

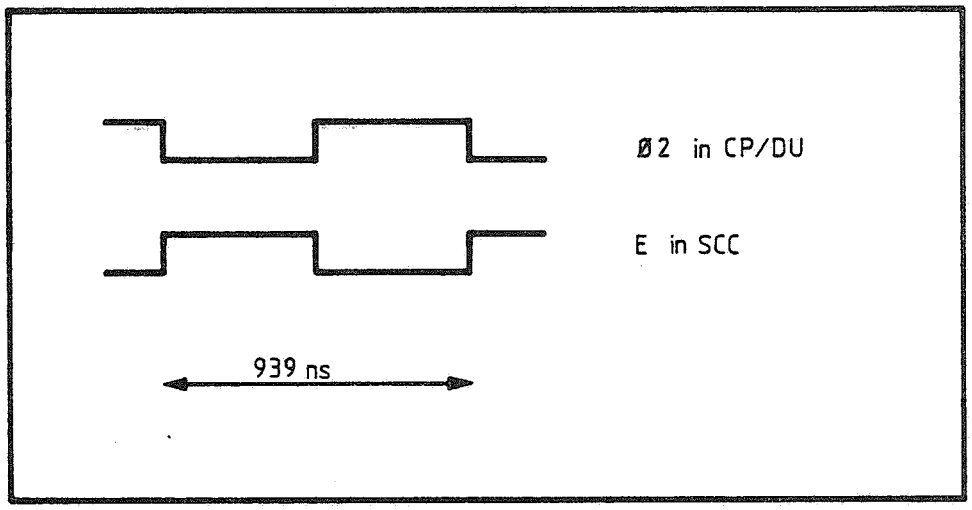


Fig. 4. Clock signals E and Ø2

- Index register (16 bits) which reduces program memory requirements since it can be loaded with a memory address different from those contained in the program counter and stack pointer

- Stack pointer (16 bits) should be initialized to point to the highest address in the RWM area to which the stack function has been assigned.

  When a Push (PSH) instruction is issued, the content of one of the accumulators (A or B) is stored in the memory location pointed to by the stack pointer, and the stack pointer is automatically decremented.

  When data is to be returned to an accumulator, the Pull (PUL) instruction is issued. This results in automatic incrementation of the stack pointer after which the content of the memory cell pointed to by the stack pointer is loaded into the specified accumulator.

- Program counter (16 bits). When a reset operation occurs or when an interrupt is obtained, this counter is loaded with the program address stored at a specific memory location in the highest part of the memory area. The content of the program counter is then stepped automatically through the program and changed for jump instructions. There is no other way to change or read the program counter contents.

- Condition code register (CCR, 8 bits) contains 6 flag bits used to indicate the results of a previous operation. The CCR bits function as follows

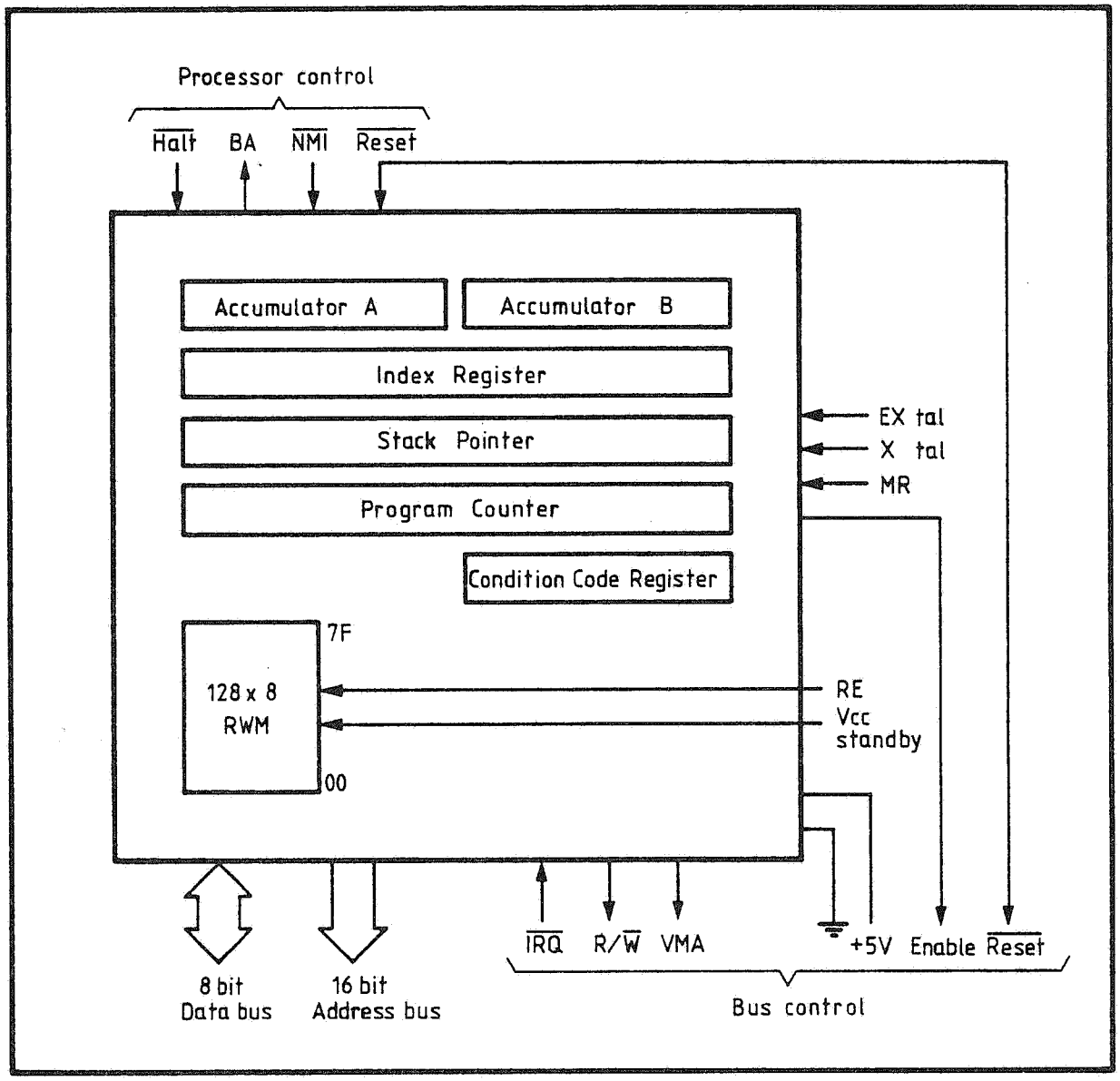


Fig. 5. MPU 6802, registers and signals

Bit	Name	Function
0	C (Carry)	Set if carry from or borrow to bit 7 was needed for an operation
1	V (Overflow)	Set if twos complement overflow occurs (result of operation greater than 127 or less than -128)
2	Z (Zero)	Set if the result of the operation is 0
3	N (Negative)	Set if the result of the operation is negative.
4	I (Interrupt)	Set after an Interrupt request (IRQ) thus hindering (masking) further interrupts.
5	H (Half carry)	Set if half-carry occurs from bit 3 to bit 4 in an operation.
6 and 7		Not used (always set)

- MPU RWM
  MPU also contains a 128-byte RWM located at hex address 0000-007F.

● MPU-signals

Apart from the 16 address bus lines and the eight bi-directional data bus lines there are several other timing and control signals in the system.

- EXtal and Xtal. The MPU has an internal oscillator that can be crystal-controlled (parallel fundamental crystal). EXtal can also be driven externally by a TTL input signal. Xtal is to be left open in this mode.

- E (Enable) is obtained by dividing the EXtal signal by four. E is a single phase TTL-compatible signal that serves as the clock for MPU and the rest of the system.

- MR (Memory Ready). When the MR input is low, E can be extended to integral multiples of half periods, thus permitting an interface to slow memories to be established.

Normal program execution can be stopped by four external signals: Interrupt Request (IRQ), Non-Maskable Interrupt (NMI), Reset and Halt.

- When the IRQ input goes low, the microprocessor jumps to an interrupt routine having its address at memory locations FFF8 and FFF9. This jump takes place after the MPU register contents are stored in the stack. However, this occurs only if the interrupt mask bit (I) in the MPU condition code register (CCR) is 0-set.

- When NMI goes low, MPU jumps to an interrupt routine having its address at memory locations FFFC and FFFD. This occurs after the MPU register contents are stored in the stack.

- When Reset goes low, the microprocessor stops. When Reset goes high again, the microprocessor jumps to a restart sequence having its address at memory locations FFFE and FFFF. The interrupt mask bit in CCR will be set in connection with a Reset, and it must be cleared by the program before an IRQ can be carried out.

- When Halt goes low, all program execution stops. Any Reset or Interrupt signals arriving while the Halt condition prevails will be preserved and serviced after Halt has gone high again.

There are also three bus mode indication signals

- When the BA (Bus Available) signal goes high, it indicates that the microprocessor has been stopped by a Halt signal or by the Wait instruction (WAI). When this happens, both the address bus and the data bus are in the high-impedance (third) state.

- When the VMA (Valid Memory Address) signal goes high, it indicates to peripheral devices that there is a valid address on the address bus.

- When the R/W (Read/Write) signal goes high, it indicates to the peripheral and memory devices that MPU is in the Read state. When this signal goes low it indicates that MPU is in the Write state.

Moreover, the following two signals are sent to the built-in RWM memory

- The V_{cc} Standby signal supplies DC voltage to the first 32 bytes of RWM.

- When the RE (RWM Enable) signal is high, the RWM is enabled so that it can respond to MPU control signals.

Address Decoder

This unit consists of a PROM (see Fig. 6) and gating logic. The address decoder generates enable pulses which are sent to units connected to the SCC bus. Decoding takes place from the address bus (bits 8 to 15).

Address decoder output	Object	Address
0	ADLC/DMA (Unused)	F0XX
1	PIA and Timer	F1XX
2	SSDA	F2XX
3	ADLC	F3XX
4	MCP (testing unit)	F4XX
5	Flag	F7XX
6	IPL-PROM	F8XX-FFXX
7	Data buffer to/from memory	18XX-7FXX

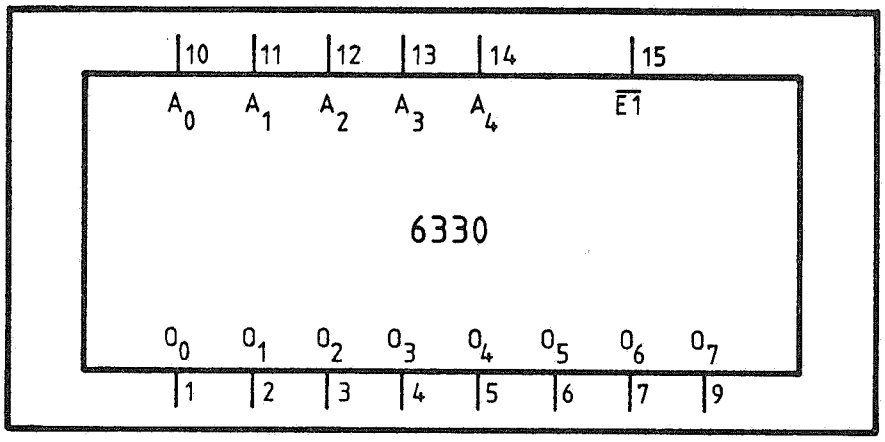


Fig. 6. PROM 6330, symbol

IPL-PROM

The IPL-PROM (F800-FFFF) contains the start-up program and the start addresses that are fetched for the different interrupt routines.

Timer

The timer unit consists of a programmable timer module (PTM) circuit. See Fig. 7.

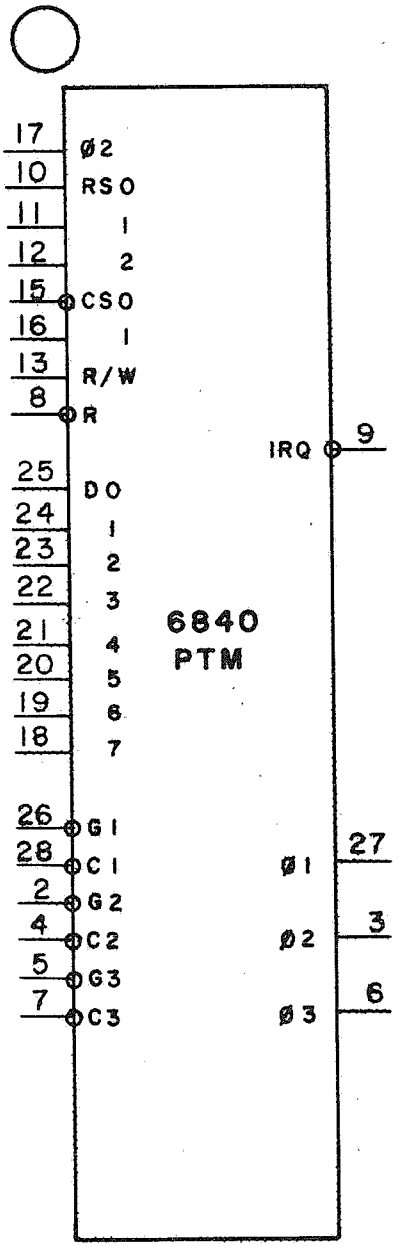


Fig. 7. PTM, symbol

The PTM circuit contains

- Three separate timers, each of which contains a 16-bit counter and a control register

- One status register

Each timer unit has a clock input C (for external clock), a gate input G and a timer-out output Ø. The counters and control registers are loaded via the data bus (D_0-D_7) and the address bus (RS0-2) See section headed Addressing which follows. Via the control registers, the counters can be programmed to operate in different modes, to select clock inputs, to mask interrupts, to mask timer-out signals, etc. The status register indicates which of the counters issued an interrupt.

For a detailed description of PTM, see the section headed PTM under the Microcomputer thumb index.

Operations carried out in SCC

Timers 1 and 2 are used by SCC to order interrupts in advance (time monitoring). Timer 1 is supplied with signals from the system clock (E). Timer 2 is supplied with signals from the output of timer 1 (Ø1). This means that the two timers are linked (32-bit counter). The program can either order an interrupt from Timer 1 (intervals of less than 61.5 ms) or from Timer 2 (longer intervals). The interrupt output on the PTM circuit is connected to level 1 in the interrupt logic.

Timer 3 is used to generate an internal clock signal sent to the line interface. Timer 3 is supplied with signals from the system clock (E) and is programmed to send out a certain frequency (1200, 2400, 4800, 9600 or 48000 Hz for example) from the Ø3 output which is connected to the interface circuits.

Addressing

The following table presents the addresses of the control registers and counters in PTM. Note that both PIA and the timer unit are connected to the same enable signal sent out from the address decoder (output having address F1XX, see section on Adress Decoder). The timer is assigned address F18X via the address bus (bit 7) which is connected to CE1. PIA is assigned address F10X via the address bus (bit 7) which is connected to $\overline{CE2}$ on PIA.

Address	R/W	CR2:0	Addressed register
F180	0	1	Write CR1
F180	0	0	Write CR3
F181	0	X	Write CR2
F181	1	X	Read status register
F182	1/0	X	Read/write counter 1 MSB (bits 15-8)
F183	1/0	X	Read/write counter 1 LSB (bits 7-0)
F184	1/0	X	Read/write counter 2 MSB
F185	1/0	X	Read/write counter 2 LSB
F186	1/0	X	Read/write counter 3 MSB
F187	1/0	X	Read/write counter 3 LSB

Peripheral Interface Adapter (PIA)

The PIA circuit (see Fig. 8) contains two separate 8-bit I/O ports designated A and B. Each of these consists of

- A data register used for incoming or outgoing data
- A data direction register used to control the data register. (When bit X in the data direction register is 1, the corresponding bit in the data register functions as an output. When bit X in the data register is 0, the corresponding bit in the data register functions as an input.)

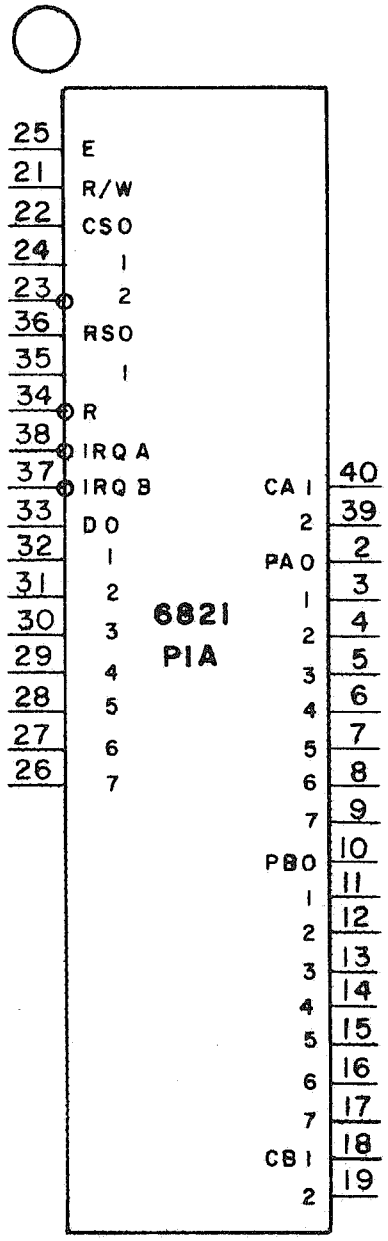


Fig. 8. PIA, symbol

- One control register used to control (among other things) interrupts obtained from the CA and CB inputs.

For a detailed description of PIA, see the section headed PIA under the Microcomputer thumb index.

The PIA circuit is used by SCC for

- Checking the interrupt level
- Masking interrupts
- Interrupt generation/acknowledgement
- Sending control signals to the interface circuits
- Indicating function by means of LED
- Parity error checking
- Controlling certain internal SCC signals.

Addressing

Address	R/W	CRA:2	CRB:2	Addressed register
F100	1/0	1	X	Read/write data register A
F100	0	0	X	Write data direction register A
F101	1/0	X	X	Read/write control register A
F102	1/0	X	1	Read/write data register B
F102	0	X	0	Write data direction register B
F103	1/0	X	X	Read/write control register B

PIA data register A is used as follows

When bit 0 output is 1	Interrupt level 0 is enabled
When bit 1 output is 1	Interrupt level 1 is enabled
When bit 2 output is 1	Interrupt level 2 is enabled
When bit 3 output is 1	Interrupt level 3 is enabled

These four outputs are connected to the interrupt logic in order to mask out certain interrupt levels. They can also be used as inputs for interrupts in order to determine the level from which the interrupt was issued. See also the section headed Interrupt Logic.

When bit 4 output is 1	Interrupt is acknowledged from CP/DU
When bit 5 output is 0	Interrupt is sent to CP/DU
When bit 6 output is 1	Function diode is lighted
When bit 7 input is 0	MCP (test unit) is in test mode
When CA1 input changes from 1 to 0	Power-on/reset has occurred
When CA2 output is 0	Interrupt is sent from operating system (OS) to level 0

PIA data register B is used as follows

When bit 0 input is 1	Parity error
When bit 1 output is 1	Parity error logic blocked
Bit 2 is not used	
When bit 3 output is 1	Internal clock pulse sent from timer to interface
When bit 3 output is 0	External clock pulse obtained from V24 or X21
When bit 4 output is 0	Active DTR (V24) or Control (X21)
When bit 5 output is 0	Active CTS (X21)
When bit 6 output is 1	V24 interface is selected
When bit 6 output is 0	X21 interface is selected
When bit 7 input is x)	DSR (V24) or Indication (X21)
When CB1 input is x)	Calling ind (V24) or Indication (X21)
When CB2 input is x)	Indication

x) Depends on programming. See also section on SCC Line Interface.

Interrupts

The four following types of interrupts are available in SCC MPU

- A power-on/reset interrupt clears all MPU internal registers to 0 and causes the start address to be fetched from FFFE-FFFF in the memory.
  A power-on/reset interrupt is generated when power to the CP/DU is turned on or the CP/DU is reset (by depressing the RESET pushbutton on CP/DU). The signal obtained when power is turned on is sent to the reset input on MPU and to the bus-connected peripheral circuits in SCC. The signal obtained from the RESET pushbutton is sent to the reset input on MPU and to the CA1 input on PIA.

- A non-maskable interrupt (NMI) causes the contents of the internal registers in MPU (except for the stack pointer) to be saved in the stack and causes the start address to be fetched from FFFC-FFFD in the memory. An NMI interrupt is generated in SCC by the parity error logic. Output from this logic can be inhibited by setting PIA B bit 1 so that an NMI interrupt will not be obtained in the event of a parity error.

  An NMI interrupt can also be generated in the microprocessor control panel (MCP) when it is connected (for testing and fault tracing in SCC).

- A software interrupt (SWI) causes the same type of save operation to be carried out as an NMI interrupt. Here, however, the start address is fetched from FFFA-FFFB in the memory. An SWI interrupt can only be generated by the program by means of the SWI instruction (3F).

- An interrupt request (IRQ) is an interrupt which causes the contents of the internal registers in MPU to be saved in the stack (except for the stack pointer). Here, the start address is fetched from FFF8-FFF9 in the memory. IRQ can be masked by the program by means of instruction SEI (0F). IRQ is generated in SCC via the interrupt logic to which the four different types of interrupts are connected.

Interrupt Logic

IRQ always provides the same start address (FFF8-FFF9) for MPU, regardless of the source of the interrupt. The interrupt program must therefore ascertain which unit caused the interrupt in order to take the correct measures.

This determination of the source of an interrupt is carried out by having MPU read PIA A bits 0-3. Each of the leads that indicate the states of these bits is connected directly to its own interrupt lead (when bit X is 1, an interrupt signal is obtained via lead X). See Fig. 3.

PIA A bits 0-3 can also be used by MPU to inhibit (mask) the signal on any desired interrupt lead by entering a 0 in the desired PIA A bit. Note that the PIA A output is of the open collector type and is connected to the interrupt lead by means of a WIRED OR connection. Using this masking function, MPU can assign internal priorities to the different interrupt leads (0-3) by masking out undesired interrupt levels.

The four different interrupt levels are described below

- Level 0 (PIA A bit 0) comprises program-controlled interrupts obtained via PIA CA2. Acknowledgment of an interrupt on this level is controlled by the program, also via PIA CA2.

- Level 1 (PIA A bit 1) consists of timer IRQ interrupts. Here, an interrupt is ordered in advance by MPU by programming the timer. Acknowledgement can be carried out (for example) by means of a Read Timer Counter command sent to the timer after the timer status register has been read.

- Level 2 (PIA A bit 2) consists of interrupts received from CP/DU (or the two-wire interface).

  The interrupt signal received from CP/DU sets a flip-flop in the interrupt logic. An interrupt on this level is acknowledged by means of a reset pulse sent to the flip-flop via PIA A bit 4 (⎍⎍).

  Due to the fact that two SCCs can be included in a CP, a jumper has been provided which permits the desired interrupt lead from CP to be selected (IRQ to SCC X or IRQ to SCC Y).

- Level 3 (PIA A bit 3) consists of interrupts received from an SSDA circuit, ADLC circuit or PIA circuit. After an interrupt, the program checks to see which of these circuits issued the interrupt. Acknowledgment of the interrupt is carried out by programming the circuit in question.

The interrupt logic also sends an interrupt signal to the main processor in CP/DU. This is accomplished by setting PIA A bit 5 to 0. A jumper can be used to send the interrupt signal to either of two leads, one for SCC X and the other for SCC Y in CP.

The interrupt logic also includes a flip-flop used for parity error checking. If a parity error occurs in SCC or on the CP/DU data bus (see section headed SCC Memory), an interrupt signal is sent to the NMI input on SCC-MPU. This signal is also sent to PIA B bit 0 to indicate that a parity error has occurred. The parity error logic can be blocked by setting PIA B bit 1 to 0.

Power-On/Reset Function

- Power on (Bus pin 81)
  When CP/DU commences to operate after power is turned on, a Power On signal which endures for about five seconds is sent to SCC. This Power On signal causes a Reset signal to be sent to SCC-MPU and to circuits connected to the SCC bus. After the Power On signal has been received, the main processor in CP/DU and SCC-MPU start their IPL programs.

  When CP/DU starts, a constant interrupt signal (IRQ 2) is sent to SCC via bus pin 7 or 87. SCC senses this interrupt signal until it vanishes, i.e. until CP/DU has initialized the PIA that was issuing the interrupt signal. The SCC IPL program then continues as far as the RWM test. This test is not started before SCC has received an interrupt and command from CP/DU.

  - If SCC does not establish contact with CP/DU within 10 seconds after the Power On signal has been obtained, the green function LED on the printed circuit board flashes at a frequency of 1 Hz.

  - If an error is detected in the RWM/ROM area, CP/DU responds to the command by issuing negative status and the green function LED flashes at a frequency of 8-10 Hz.

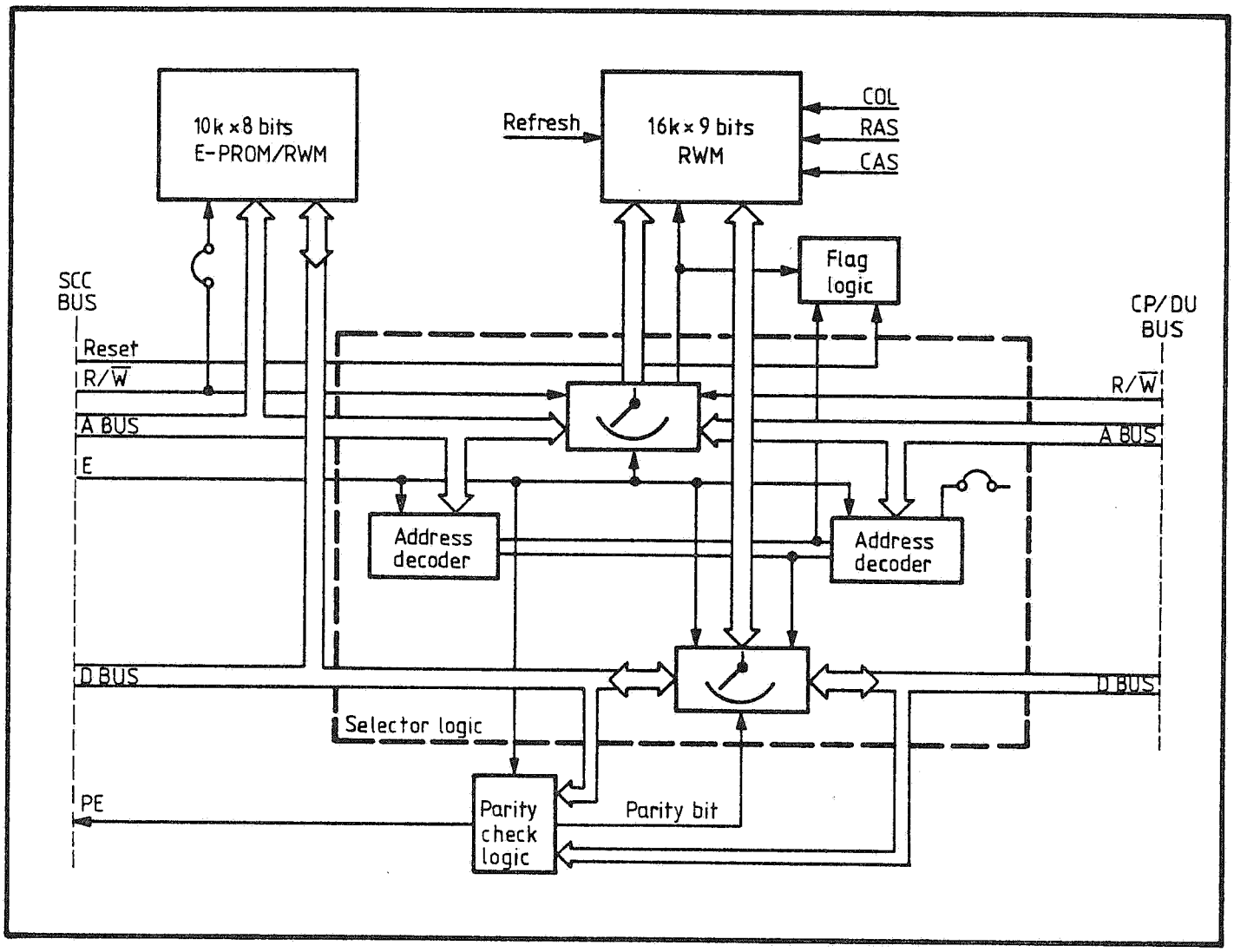


Fig. 9. SCC memory, block diagram

- If the error is of such a type that there can be no response to the command, there is a time monitoring function in CP/DU which causes error status to be issued from SCC.

- If the IPL tests have been passed successfully, the command issued by CP/DU is answered with positive status and the green function LED glows steadily.

- Reset (Bus pin 82)
  The Reset signal is sent to the reset input on SCC-MPU and to CA1 on PIA, where bit 7 in control register A is set to 1. The IPL program then waits for an interrupt and command from CP/DU which can

  - Request the dumping of the contents of the memory and peripheral circuits in SCC after which the Power On signal will be issued.

  - Issue the Power On signal immediately.

SCC Memory

The SCC memory, see Fig. 9, contains memory circuits used for storing the system program and for buffer storage of messages to/from the host computer. The buffer memory can be addressed by both SCC and CP/DU via the selector logic which is controlled by the system clock (E). The SCC memory also contains logic used for parity bit generation and checking. The flag logic controls processor operations in the buffer memory.

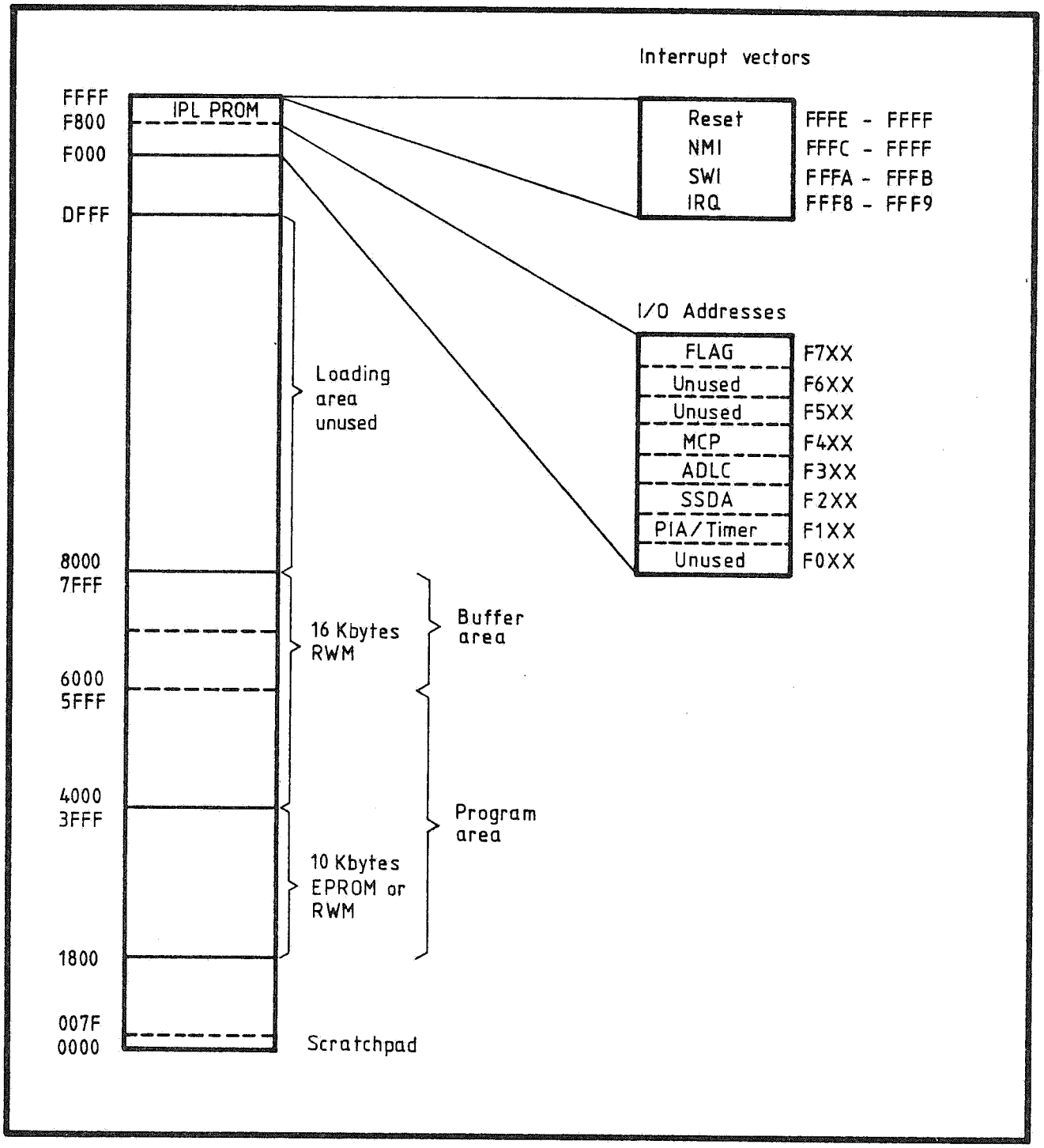


Fig. 10. Memory map

Memory Area

The memory area in SCC, see Fig. 10, is arranged as follows

- Scratchpad area (address 0000-007F).
  This 128-byte memory area is built into the MPU circuit.

- Program area (address 1800-5FFF).
  The 18-Kbyte program area is arranged in such a way that the low part (10 Kbytes) can consist of either PROM or RWM circuits. The high part of the program area consists of 8 Kbytes of RWM.

- Buffer area (address 6000-7FFF).
  This 8-Kbyte RWM memory area can be addressed by both SCC and CP/DU.

- Loading area (address 8000-DFFF). This area is unused in SCC.

- I/O address area (address F0XX-F7XX). This area is used for registers in bus-connected circuits and units (MCP).

- IPL-PROM area (address F800-FFFF).
  This part of the memory contains the start-up program and jump addresses for different types of interrupts.

10 Kbytes of EPROM/RWM

This memory is arranged so that it can be equipped with either EPROM (2716) or RWM (6116). The Write signal is eliminated by means of a jumper when this memory is equipped with EPROM.

16 Kbytes of RWM

This memory consists of dynamic RWM (4116) and an address multiplexer/refresh counter circuit. Eight Kbytes of this memory are used as a buffer memory which can be addressed by both the SCC-processor and the CP/DU-processor. The size of the buffer area and the CP/DU address to the memory are determined by means of a jumper in the CP/DU address decoder in the selecter logic as follows

FUNCTION	AREA	SCC ADDR	CP/DU ADDR
SCC X in CP	8 kB	6000-7FFF	6000-7FFF
SCC Y in CP	4 kB	6000-6FFF	E000-EFFF
SCC Y in DU	4 kB	6000-6FFF	E000-EFFF

Selector Logic

The selector logic handles the switching of the data bus and address bus and the R/W signal that is needed to permit two processors to operate in a shared memory. This switching between buses is controlled by the E signal and the address decoders.

The SCC bus and the CP/DU bus each have an address decoder which sends enable pulses to the data bus selector and flag logic. The CP/DU address decoder is provided with jumpers that can be used to select the size of the buffer memory, the address of the buffer memory and the address of the flag logic. (See sections headed 16 Kbytes of RWM and Flag Logic.)

Parity Check Logic

This unit provides a parity bit obtained from the SCC data bus or CP/DU data bus. This parity bit is entered together with data into the 16-Kbyte RWM, which contains 9-bit words (D_0-D_7 + parity). The parity check unit also reads the data bus in connection with a read sequence and sends the Parity Error (PE) signal if a parity error is detected. The PE signal is sent to the parity error logic (see section headed SCC-MIC).

Flag Logic

The flag logic indicates that a processor is operating in the shared memory area (processing the pointer queue, for example). If a processor seeks access to a buffer area, it must first test the flag. If a 0 is read, it indicates that the buffer area is being used by another MPU. The processor must then continue testing until the flag reads 1. After the processor has read this 1, the flag is reset to 0 automatically by the read pulse, thus indicating to the other MPU that the memory is busy. After the operation is completed, the processor sets the flag to 1 by means of a write sequence sent to the flag address. The flag becomes 1 regardless of the contents of the data bus. The flag contains one byte in which only bit 7 is active (the states of the other bits are irrelevant).

The address of the flag logic can be changed by means of a jumper in the CP/DU address decoder, which sends the enable pulse to the flag logic.

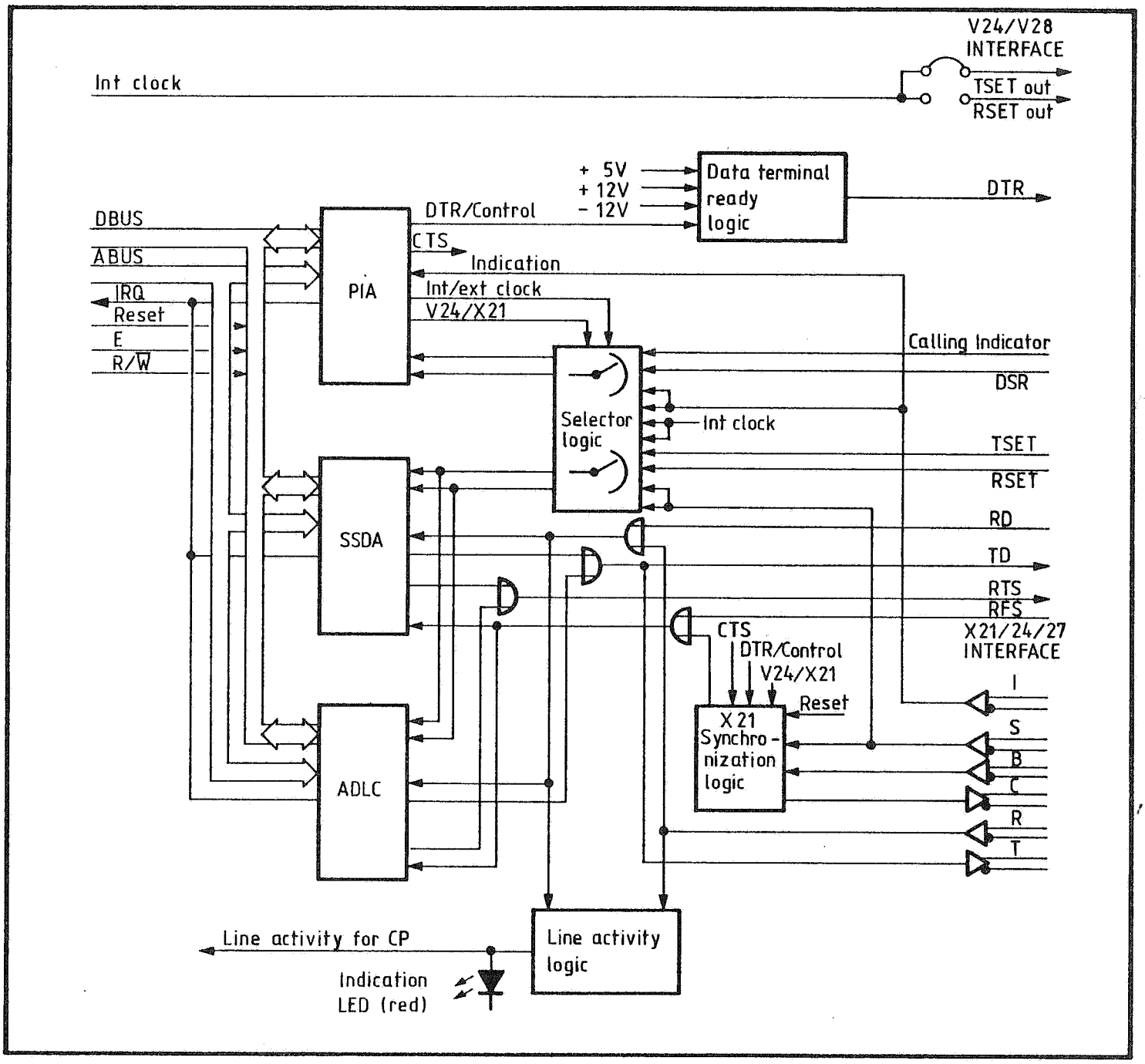


Fig. 11. SCC line interface, block diagram

FUNCTION	SCC ADDR	CP/DU ADDR	DATA BIT 7
SCC X in CP	F7XX	F748-F74B	0 = Busy 1 = Available
SCC Y in CP	F7XX	F74C-F74F	
SCC Y in DU	F7XX	F74C-F74F	

SCC Line Interface

This part of SCC controls the communication with a host computer. Communication proceeds via two interface circuits designated SSDA and ADLC. These circuits are designed to handle bi-directional synchronous data protocols.

- Binary Synchronous Communication (BSC) is established via SSDA.

- Synchronous Data Link Control (SDLC) and High-level Data Link Control (HDLC) are established via ADLC.

- Nordic Public Data Network (NPDN) communication is established via SSDA or a combination of SSDA and ADLC.

Communication proceeds via one of two interfaces, the V24/28 or the X21/24/27. Each circuit can communicate via either of these interfaces. The circuit is selected by programming the current circuit. The interface is selected by programming PIA. PIA then controls the selector logic which selects the interface.

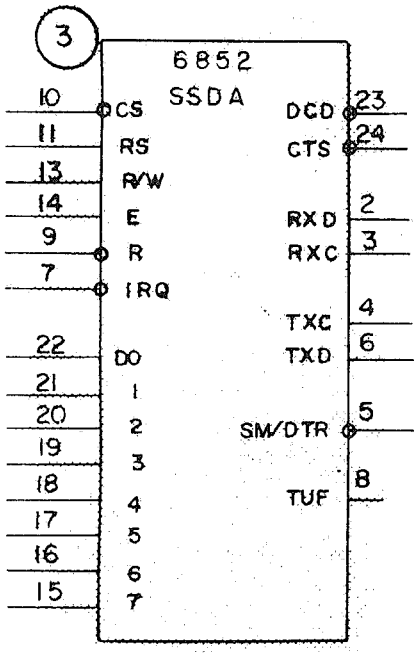


Fig. 12. SSDA, symbol

Synchronous Serial Data Adapter (SSDA)

SSDA, see Fig. 12, contains two read-only registers (the status register and the receive data FIFO register) and five write-only registers (three control registers, a sync code register and a transmit data FIFO register). These are addressed by means of the Chip Select (CS), E, R/W, and Address (RS) signals. (The E signal is also used to transfer data through the internal 3-byte receive and transmit data FIFO registers.) The states of the two address control bits in CR1, together with the external address signals mentioned above, specify the register that is to be accessed. Apart from these signals the SSDA interface to MPU carries 8-bit data bus information (D_7 through D_0), the Reset signal and the Interrupt Request (IRQ) signal. Note that the Reset signal is used to reset the SSDA by means of two control bits called receiver reset and transmitter reset.

SSDA communicates with the interface by means of the following signals

- Receiver section:
  - RXD: Bit-serial receive data input
  - RXC: Receive bit clock input

- Transmitter section:
  - TXD: Bit-serial transmit data output
  - TXC: Transmitter bit clock input
  - CTS: Clear to send input. A positive-going transition at this input will be stored in the status register and can cause an IRQ.
  - SM/$\overline{\text{DTR}}$: Sync match/Data terminal ready. (See RTS signal in section headed V24/28 Interface.)

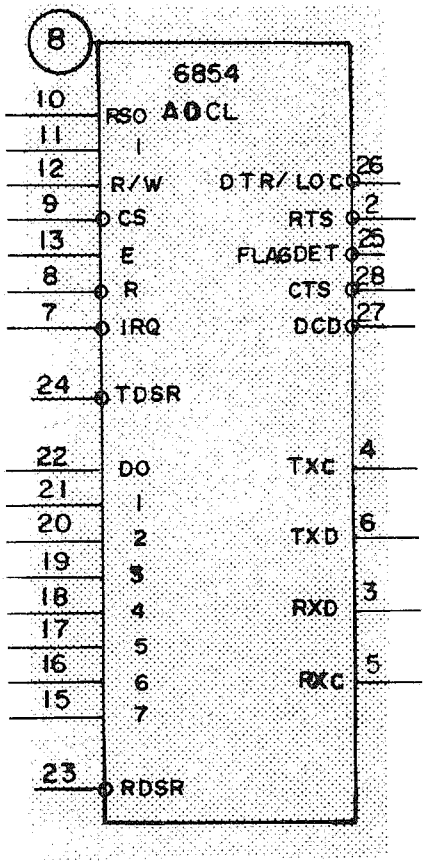


Fig. 13. ADLC, symbol

Addressing

Address	R/W	CR1:7	CR1:6	Addressed register
F200	0	X	X	Write control reg. 1 (CR1)
F200	1	X	X	Read status reg.
F201	1	X	X	Read receive data FIFO reg.
F201	0	0	0	Write control reg. 2
F201	0	0	1	Write control reg. 3
F201	0	1	0	Write sync code reg.
F201	0	1	1	Write transmit data FIFO reg.

Advanced Data Link Controller (ADLC)

ADLC, see Fig. 13, contains four read-only registers (two Status registers and two receive data registers) and six write-only registers (four control registers and two transmit data registers). These registers are addressed by means of R/W, CS, RS0, RS1 and bit 0 in control register 1. (See section headed Addressing.) The connection to MPU consists of a 8-bit data bus (D_0-D_7), the Reset signal (R) line and the Interrupt request signal (IRQ). The system clock (E) is used to transfer data through the internal 3-byte receive and transmit registers.

ADLC communicates with the interface by the following signals:

- Receiver section:
  - RXD: Bit-serial receive data input.
  - RXC: Receiver bit clock input.
- Transmitter section:
  - TXD: Bit-serial transmit data output.
  - TXC: Transmitter bit clock.
  - CTS: Clear to send input. A positive-going transition of this input will be stored in the status register and can cause an IRQ.
  - RTS: Request to send output. (See section headed V24/28 Interface.)

Addressing

Address	R/W	CR1:0	Addressed register
F300	1	X	Read status register 1
F301	1	X	Read status register 2
F302, F303	1	X	Read receive data registers
F300	0	X	Write control reg. 1 (CR1)
F301	0	0	Write control reg. 2
F301	0	1	Write control reg. 3
F303	0	1	Write control reg. 4
F302	0	X	Write transmitter data reg. (frame continue)
F303	0	0	Write transmitter data reg. (frame terminate)

Selector Logic

The selector logic selects the interface (V24 or X21) that is to be used and the clock source (internal or external clock). The selector logic is controlled by PIA B bit 6 (selects V24 or X21 interface) and PIA B bit 3 (selects internal or external clock).

The selected clock signal (Internal Clock, RSET and TSET (V24) or S (X21) are sent to SSDA and ADLC circuits inputs GR_X and GT_X.

The selected interface signals are sent to PIA as follows if the V24 interface is selected

- Calling Indicator — PIA CB1.
- DSR (Data Set Ready) — PIA B bit 7.

If the X21 interface is selected, the selected interface signals are sent to PIA as follows

- Indication — PIA B bit 7 and CB1.
  Note that Indication is also connected to PIA CB2.

V24/28 Interface

The V24/28 interface contains the following signals

- RD (Receive Data Input). This signal is sent to both SSDA and ADLC. The program determines which of these circuits is to handle incoming data.

- TD (Transmit Data Output). This signal consists of serial data from SSDA or ADLC.

- RSET (Receive Bit Clock) and TSET (Transmit Bit Clock). These signals are received from external equipment.

- RSET Out and TSET Out. These signals consist of bit clock signals generated in SCC by the timer.

- RTS (Request To Send). This signal is generated either by SSDA (SM/$\overline{\text{DTR}}$ output) or ADLC ($\overline{\text{RTS}}$ output) to prepare the data circuit terminating equipment (typically a modem) for transmission to the line from SCC.

- RFS (Ready For Sending) is a signal issued by the modem to indicate that it is ready to send the information received via the transmit data line from SCC. This signal proceeds to the CTS (Clear To Send) inputs on SSDA and ADLC.

- DTR (Data Terminal Ready) is a signal that can be used (among other things) as an enable signal for the modem. This signal indicates that the data terminal equipment is operable. It is generated when the supply voltages (±12 V and +5 V) for SCC are present and the program has indicated that it is "ready" by entering a 0 in PIA B bit 4.

- The DSR (Data Set Ready) signal is sent to PIA via the selector logic.

- The Calling Indicator input signal is sent to PIA via the selector logic.

X21/24/27 Interface

The X21/24/27 interface contains the following signals

- R (Receive Data). This signal is sent to both SSDA and ADLC. The program determines which of these circuits is to handle incoming data.

- T (Transmit Data). This signal carries serial data obtained from SSDA or ALDC.

- S (Signal Element Timing). This signal originates at the data circuit terminating equipment (DCE). Transmitted data is shifted out of SSDA/ADLC when signal S changes state (_┌‾). Received data is shifted into SSDA/ADLC when signal S changes state (‾└_).

- B (Byte Timing). This input signal is used to synchronize the activating of the C signal synchronously with DCE (in connection with the transmission of bytes and start-up operations from SSDA/ADLC). This is accomplished by activating the CTS (Clear To Send) signal.

- C (Control). This signal is used to indicate to DCE that (among other things)

_┌‾ I wish to establish a connection
‾└_ I wish to clear the connection

or

_┌‾ I accept a call
‾└_ I confirm a clear connection indiation from DCE

The C (Control) signal is generated by activating the DTR/Control signal (PIA B bit 4 = 0). C will go high in response to the next positive-going transition of B which is followed by a positive-going transition of S. C is deactivated in response to the first negative-going transition of S followed by the setting of PIA B bit 4.

- I (Indication). This signal is sent out from DCE to indicate that (among other things)

_┌‾ The connection I wanted is established
‾└_ DCE confirms my "clear connection" request

or

_┌‾ Connection wanted by "other end" is established
‾└_ "Other end" wishes to clear the connection

Due to the fact that the connection/disconnection sequences take some time, the I signal provides an indication of changes in state to the microprocessor software by interrupting MPU. I proceeds directly by CB1 in PIA, and also to CB2 and PIA B bit 7 via the selector logic. CB1 and CB2 are used to generate IRQ on transitions and PIA B bit 7 is used to sense the level of I.

Line Activity Logic

The line activity logic, see Fig. 9, indicates activity on the RD and TD lines by issuing the Line Activity signal. This signal is sent to the CP bus. If there are two SCCs in a communication processor, the Line Activity signal can be connected via jumpers so that each SCC transmits it own signal indicating line activity on the CP bus. The line activity logic also controls a red LED which indicates activity on the receive and transmit data lines as follows

- Extinguished when there is no activity on either line

- Glows steadily when data is being transmitted from SCC without any pauses longer than 17 seconds between transmissions

- Flashes when data is being received and pauses between transmissions are longer than 17 seconds

Data_Terminal_Ready_Logic_

This logic makes certain that the supply voltages (±12 V and +5 V) are present. This condition, together with the DTR/Control signal received from PIA B bit 4, indicates to the modem (via signal DTR) that the data terminal equipment is operable.

Asynchronous Communication Adapters,

ACA 4185/4193/4199

Contents

EE366-810B
1984-02-15

General

The Asynchronous Communication Adapters, ACA, ACA-A, -B, -C and -D, are used to adapt the parallel data bus of display processors to serial transmission lines using asynchronous communication (with start and stop bits). Occasionally they might be connected to other types of processors. The board exist in several versions with different serial interfaces:

- ACA (4193-001) with:
  – V.24 printer interface.
  – V.24/V.28 modem interface.
  – (Current loop interface. This interface shall not be used. Utilize instead the ACA-A.)
- ACA-A (4193-002) with:
  – V.24 printer interface.
  – V.24/V.28 modem interface.
  – Current loop interface.
- ACA-B (4199-001) with V.24 printer interface.
- ACA-C (4185-001) with fibre optic interface. The ACA-C is used as a printer interface but is also prepared for modem communication. Only the printer interface functions are fully described in this document.
- ACA-D (4199-002) with:
  – V.24 printer interface.
  – Personal computer interface. (A type of modem interface.)

ACA, ACA-A and -C can only be used either as a printer or as a modem (or current loop in the case of ACA-A) interface and must be strapped accordingly. Strapping information can be found in the Installation and Maintenance manual.

The reader needs understanding of a display processor as described in e.g. the Display Units, DU 4111 and 4112 chapter before this text is studied.

Please refer to the block diagrams in Appendices 2 – 6 when reading this text.

Brief Outline

Common Circuits

All asynchronous communication adapters are built-up around three large scale integrated, LSI, circuits, which are connected to the microcomputer bus of – normally – a display processor and form a link between the microcomputer bus and the specific logic, line drivers and line receivers of the different interfaces mentioned under General. In the ACA-D, however, two of these circuits (the ACIA and PTM, see below) are doubled in order to enable software controlled switching between printer and personal computer operation. The LSI circuits are:

- ACIA, asynchronous communication adapter, which converts 8-bit parallel data to serial data for transmission and received serial data to 8-bit parallel data. Some control signals to and status signals from the specific interface are also handled by the ACIA. The ACIA is used in connection with all specific interfaces.
- PTM, programmable timer module which is used to generate internal transmitter and receiver bit clocks i.e. when the clocks are not provided by an external source as might be possible in the modem interfaces. The PTM can also be used to produce time dependent interrupts without any real connection to the asynchronous communication functions.
- PIA, peripheral interface adapter, which, via the data bus, can be ordered to generate control signals to the specific interfaces or be used for tests of status conditions in the interfaces. The status is then available on the microcomputer bus.

All LSI circuits can make interrupt requests to the microcomputer under certain conditions.

The microcomputer address bus bits 7 – 0 are decoded in address decoding logic and in the LSI circuits, when an enable signal (Per Ø2) from the microcomputer is active, making all reachable registers in the LSI circuits accessible to the microcomputer.

Line Activity Indication of ACA, ACA-A and -C

The ACA, ACA-A and -C are equipped with circuits, which drive a light emitting diode (LED) indicating line activity and function like this:

- Extinguished when data is neither received nor transmitted.
- Blinks when data is only received (≈ 8 Hz).
- Glows steadily when data is transmitted and for about 16 seconds after the last activity on the transmitted data line.

Printer Interface of ACA, ACA-A, -B and -D

As said above the ACA and ACA-A can be hardware-strapped to function as printer interfaces. The ACA-B contains only a printer interface and the ACA-D can concurrently be used as a printer interface and a personal computer interface.

The interface transfers, via the Transmitted data output of the ACIA, data to a Data-to-printer line, which can be up to 600 m (1 970 feet). An Enable signal, which for example can be used to start a printer motor, can also be transmitted.

The printer interface can also receive four status signals from the printer, i.e. Ready A, Ready B, Busy and Parity error. The Parity error line can alternatively be used to receive data from the printer in the ACA-A, -B and -D.

Fibre Optic Printer Interface of ACA-C

Fibre Optic Link

The fibre optic interface is used when IPF (information protection feature) is wanted. The complete link system consists of:

- ACA-C which converts parallel signals to serial data on a fibre optic cable and vice versa.
- An up to 1 000 m (3 280 feet) long fibre optic cable with two optic fibres, one for each transfer direction.
- FCU, fibre optic communication unit converting serial data on the fibre optic cable to parallel signals for a printer interface and vice versa.

The system can handle four channels in each direction. The transmission speed is 0 to 19 200 baud depending on printer.

The cable and the FCU are not described in this document.

Fibre Optic Interface Functions

The fibre optic interface consists of a transmitter part and a receiver part.

The transmitter is strapped for two channels usage and transfers a Data-to-printer and an Enable signal. The signals are loaded in a parallel-in-serial-out register and then shifted out to a modulator. The modulator converts the signals to a frequency-shifted signal fed further to an optic transmitter, which drives the optic fibre.

The above mentioned register is controlled by a timing logic and is synchronized with Data-to-printer (or transmitter clock if strapped accordingly).

The receiver recognizes frequency-shifted serial data on an optic fibre. A demodulator demodulates the signal and generates a bitclock, which shifts the data into a serial-in-parallel-out register.

A timing logic checks when four new bits are stored in the register. The register contents are then transferred to a latch. The four bits of the latch are interpreted as:

- Parity error, PE, or Received data, RD.
- Busy.
- Ready A.
- Ready B.

In addition to the line activity indication mentioned above, a signal detector checks the quality of the received signal.

The circuit resets the latch and turns on a LED if:

- The optical input is overdriven by the use of not recommended fibre.
- The quality of the optical signal goes below a fixed limit.
- The frequency-shifted signal concept is not correct, i.e. if the attenuation of the cable is increased due to damage or the transmitting unit is turned off.

When the fault condition is ended all functions start again. No reset is necessary.

The frequency-shifted signal denotes, in both directions, a 0 with 300 kHz and a 1 with 150 kHz.

Modem Interface of ACA and ACA-A

As said above, the ACA and ACA-A can be hardware-strapped to function as a modem interface. This function is mainly used in TTY or VT 100 applications when a display processor communicates with a computer – with or without actual use of a modem.

The interface can transmit and receive serial data. It can also generate timing, control and status signals to the data communication equipment, DCE. They are: Transmitter signal element timing (internal clock), Request to send, Data terminal ready and in the ACA-A also Data signalling rate selector. The interface can also receive timing and status signals from the DCE. They are: Transmitter signal element timing (external clock), Data set ready, Ready for sending, Receiver signal element timing (external clock), Data carrier detector and Calling indicator. The transfer distance can be up to 15 m (50 feet).

Current Loop Interface of ACA-A

As said above the ACA-A can be hardware-strapped to function as a current loop interface. (Actually with the same strapping as for modem operation.) The interface can transmit data on one line and receive data on another line.

The interface can work in either of two modes, i.e. passive or active. Passive mode means that the needed currents are generated in an external source. See Fig. 1. The currents are generated on the ACA-A board in active mode. See Fig. 2. In passive mode the transfer distance depends on the equipment at the other side of the link (called Computer in the figures). In active mode the maximum distance is a function of the transfer speed:

Speed	Distance
9 600 baud	300 m (985 feet)
4 800 baud	600 m (1 970 feet)
2 400 baud	900 m (2 955 feet)
≤1 200 baud	1 500 m (4 925 feet)

Electrical isolation is obtained in passive mode but not in active mode.

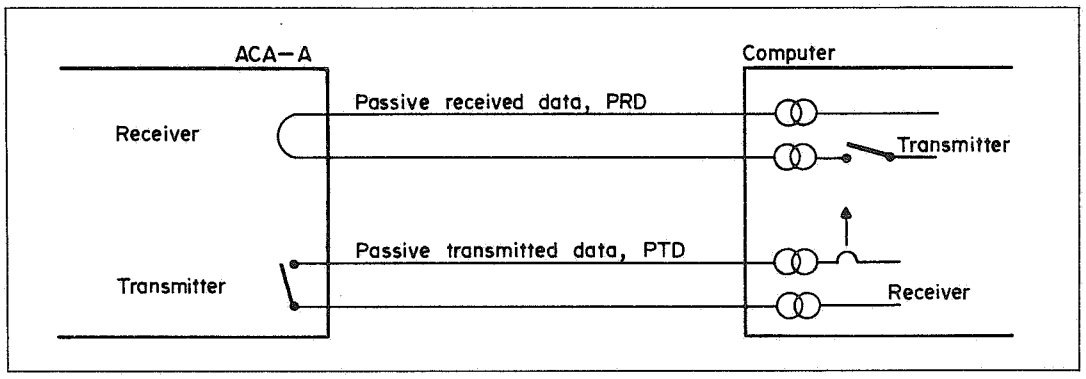


Fig. 1. Current loop, passive mode.

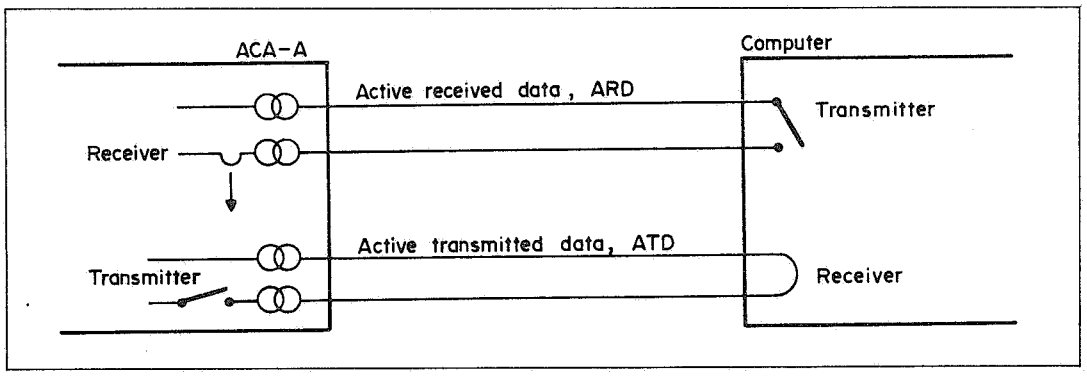


Fig. 2. Current loop, active mode.

Passive or active mode is selected by connecting the line to different pins in the connector. See Fig. 3. It is possible to mix passive and active mode, e.g. active transmitter and passive receiver, but not active and passive receiver.

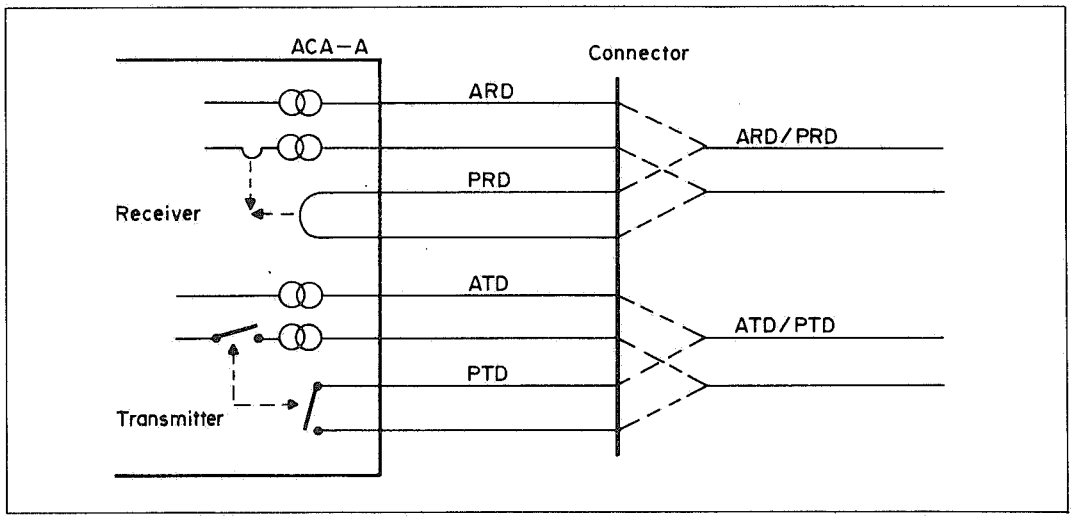


Fig. 3. Current loop, mode selection.

Personal Computer Interface of ACA-D

The personal computer interface of the ACA-D can be used concurrently with the printer interface on the board. The interface can with some limitations also be used as a modem interface.

The interface can transmit and receive serial data. It can also generate timing, control and status signals, i.e. Transmitter signal element timing (internal clock) Request to send and Data terminal ready. The interface can also receive timing and status signals, i.e. Transmitter signal element timing (external clock), Ready for sending and Receiver signal element timing (external clock). Internal or external clocks must be selected by hardware strapping. The transfer distance can be up to 15 m (50 feet).

Detailed Description

Microcomputer Interface

Signals

The microcomputer interface consists of the following logical signals:

- Data bus, DB7 – DB0.
- Address bus lines AB7 – AB0.

- Per I/O, decoded from the complete microcomputer address bus. (Activated by the addresses $F700_{(16)} - F7BF_{(16)}$ and $F7E0_{(16)} - F7EF_{(16)}$ in the DU 4110 and by $F700_{(16)} - F7FF_{(16)}$ in the DU 4111 and DU 4112.)
- Read/Write, R/W.
- Per Ø2, an 1.065 MHz clock signal synchronizing the ACA with the microcomputer.
- Interrupt requests, IRQ2 and IRQ4. One of these is selected by a jumper in the ACA, ACA-A and -C. Only IRQ2 is used in the ACA-B. Both are used in the ACA-D. (The numbers 2 and 4 above do not imply that the lines are connected to the corresponding interrupts in the microcomputer.)
- Reset, generated at power on of the microcomputer or by the microcomputer program.
- 9.585 MHz, generated by the microcomputer for timing purposes is only used in the ACA-C.

An internal power on reset, POR, is also generated in the ACA-A, -B and -D. POR is or-ed together with Reset from the microcomputer.

Power, −12 V, 0 V, +5 V and +12 V, is also supplied to the board. The −12 V and +12 V are not used in the ACA-C.

Addressing

Address bus bits AB7 – AB3 are decoded in an address decoder (two decoders in ACA-D) to chip select signals as shown below:

Type of ACA	Inputs						Outputs	Comments[3]	
	Per I/O	Address bus bits					Signal	Address (Hex)	Selects
		7	6	5	4	3			
ACA,	L	L[1]	H[1]	L	H	L	$\overline{CSP}$	F750-7	PIA (Printer)
ACA-A	L	L[1]	H[1]	L	H	H	$\overline{CST}$	F758-F	PTM (Printer)
and	L	L[1]	H[1]	H	L	L	$\overline{CSA}$	F760-7	ACIA (Printer)
ACA-C	L	H[2]	L[2]	L	H	L	$\overline{CSP}$	F790-7	PIA (Modem)
	L	H[2]	L[2]	L	H	H	$\overline{CST}$	F798-F	PTM (Modem)
	L	H[2]	L[2]	H	L	L	$\overline{CSA}$	F7A0-7	ACIA (Modem)
ACA-B	L	L	H	L	H	L	$\overline{CSP}$	F750-7	PIA (Printer)
	L	L	H	L	H	H	$\overline{CST}$	F758-F	PTM (Printer)
	L	L	H	H	L	L	$\overline{CSA}$	F760-7	ACIA (Printer)
ACA-D	L	L	H	L	H	L	$\overline{CSP}$	F750-7	Printer PIA
	L	L	H	L	H	H	$\overline{CST}$	F758-F	Printer PTM
	L	L	H	H	L	L	$\overline{CSA}$	F760-7	Printer ACIA
	L	H	L	L	H	H	$\overline{CST2}$	F798-F	Modem PTM
	L	H	L	H	L	L	$\overline{CSA2}$	F7A0-7	Modem ACIA

[1] If jumpered as printer interface.
[2] If jumpered as modem interface.
[3] Valid when connected to display processors.

When any of the chip select signals is active an ACA signal is activated, which enables a bidirectional data bus buffer. Read/Write, R/W, controls the direction of the transfer.

Address bus bits AB2 – AB0 are directly used for addressing of specific registers in the selected circuit.

The specific use of each display processor I/O address is stated in Appendix 1.

LSI Circuits

The LSI circuits are discussed separately as they may have a function in all I/O interfaces of any ACA board. Some of the signals mentioned here will therefore be further explained in the context of each specific I/O interface described later.

Peripheral Interface Adapter, PIA

A general survey of, and definitions relating to, the peripheral interface adapter, PIA, can be found under Peripheral Interface Adapter, PIA in the microcomputer chapter.

The PIA is connected to the microcomputer bus. One chip select signal, CSP, is fed from the address decoder (the one of the printer interface on the ACA-D board).

The CA and CB control lines of the PIA are programmed as inputs and normally for interrupt generation at positive transitions of the following signals:

CA1: Ready A.

CA2: Ready B.

CB1: Parity error, PE. In the ACA and ACA-A modem operation: Calling indicator, CI, instead of PE.

CB2: Data set ready, DSR, but only in the ACA-A.

The modem signals Data set ready, DSR and Data carrier detector, DCD are in the cabling presently connected to Ready A and Ready B respectively of the ACA and ACA-A boards.

The interrupt request outputs, IRQ A and IRQ B, are treated under Interrupt Request Lines later on.

The peripheral registers PA and PB are used in the following way:

PA7 Input Ready A. (1 = low printer line level, printer off or not connected.)

PA6 Input — Ready B. (1 = low printer line level, printer off or not connected.)

PA5 Output — 0 = Select internal clocks.
1 = Select external clocks.
(Only internal clocks exist in the ACA-B and the printer interface of the ACA-D. The clock source must be selected by strapping in the personal computer interface of the ACA-D.)

PA4 Output — 0 = Data terminal ready, DTR. This signal is used in the ACA and ACA-A and can be used in the ACA-C if strapped so.

PA3 Input — Transmitted data. (Test input.) In the ACA, ACA-A and -C this signal is supplied by the line activity logic.

PA2 Input — Transmitter clock. (Test input.) In the ACA-A is only the internal clock supplied.

PA1 Input — Receiver clock. (Test input.)

PA0 Output — 0 = Reset parity error flip-flop or parity logic is not used (but a parity error signal from the printer can be detected if CB1 is enabled).
1 = Enable parity logic. (See also PB3.) This bit shall be 0 at modem operation.

PB7 Input — Printer operation:
1 = Parity error, printer off or not connected.

Modem operation:
1 = Calling indicator active.
This signal is only used in the ACA and ACA-A.

PB6 Input — 0 = Data set ready, DSR. This input is only connected in the ACA-A and -C.

PB5 Output — Not used. (Default value = 0.)

PB4 Output — Printer operation:
0 = High line level interpreted as Busy. (Default value.)
1 = Low line level interpreted as Busy. This signal is not used in the ACA. The selection can also be made by strapping in the ACA-C.

Modem operation:
0 = Data carrier detect, DCD, enabled to the DCD input of the ACIA. This signal is only used in the ACA-A.

PB3 Output — 0 = Parity error and Busy are used.
1 = Receive data from printer and deactivate the parity flip-flop and the busy logic 1 and 2. This bit is not used in the ACA, can be strapped 0 in the ACA-A and -C and shall be 1 at modem operations.

PB2 Input — In the ACA:
External transmitter clock, TSET. (Test input.)

In the ACA-A, -B, -C and -D:
0 = Transmitter clock is disabled by activated busy logic 1. (Test input.)

PB1 Input — In the ACA:
External receiver clock, RSET. (Test input.)

In the ACA-A, -B, -C and -D:
0 = Transmitter clock is disabled by transmitted data = 1 and activated busy logic 1 or parity error. (Test input.)

PB0 Output — Printer operation:
0 = Enable busy logic 1.
1 = Disable busy logic 1.
(Busy logic 2 functions always unless when PB3 = 1 in the ACA-A, -B and -D.)

Modem operation:
0 = Set Data signalling rate selector, DRS, to high rate.
1 = Set DRS to low rate. This signal is only used in the ACA-A.

When there has been a reaction to the Parity error signal (e.g. a reloading of the erroneous byte into the ACIA transmit data register) transmit bit clock for the ACIA must be enabled again. This may be done by sending the Reset signal but, preferably, PIA PA0 is set to a 0 to enable the clock and afterwards to a 1, to enable further parity error detection.

In the description above only the hardware is considered. Some functions are therefore not used in all applications. Note also that some inputs are only used for program controlled tests of the hardware, e.g. PA3 – PA1 and PB2 – PB1.

Asynchronous Communication Interface Adapter, ACIA

A general survey of, and definitions relating to, the asynchronous communication interface adapter, ACIA, can be found under Asynchronous Communication Interface Adapter, ACIA in the Communication chapter.

The ACIA is connected to the microcomputer bus. One chip select signal, CSA, is fed from the address decoder on the ACA, ACA-A, -B and -C boards and one chip select signal, CSA or CSA2, from each address decoder to each ACIA on the ACA-D board. The interrupt request output, IRQ, is treated under Interrupt Request Lines later on.

All ACIA inputs and outputs of the serial interface side can be used except the Data carrier detect input, DCD, which only can be used in the ACA-A and if strapped so in the ACA-C.

Programmable Timer Module, PTM

A general survey of, and definitions relating to, the programmable timer module, PTM, can be found under Programmable Timer Module, PTM in the Microcomputer chapter.

The PTM is connected to the microcomputer bus. One chip select line, CST, is fed from the address decoder on the ACA, ACA-A, -B and -C boards and one chip select signal, CST or CST2, from each address decoder to each ACIA on the ACA-D board. The interrupt request output, IRQ, is treated under Interrupt Request Lines below.

The outputs of timer 1 and 2 can be used as internal transmitter and receiver clocks respectively. The 03 output, C1 – C3 and G1 – G3 inputs are not used but timer 3 may be used for interrupt generation.

Interrupt Request Lines

The interrupt request lines are tied to the microcomputer bus in different ways on the various boards.

The two interrupt request outputs of the PIA (IRQ A and B) and the interrupt request outputs of the ACIA and the PTM (printer ACIA and PTM on the ACA-D board) are or-wired, thus generating a combined interrupt. This interrupt request can be strapped to IRQ 2 (printer interrupt) or IRQ 4 (modem interrupt) in the ACA, ACA-A and -C and is fed to IRQ 2 in the ACA-B and D. Another or-wired interrupt request is fed from the ACIA and the PTM of the personal computer interface to IRQ 4 in the ACA-D.

Programming Example

Assume that there is a need to produce a receive clock for Received data with a bit rate of 9 600 Hz.

1. Internal clocks are selected by writing PIA PA5 to a 0.
2. Receive (and transmit) clock rate is selected to be 16 times the bit rate and internal synchronization is selected by writing ACIA control register bits 1 and 0 to $01_{(2)}$.
3. Timer 2 of the PTM is programmed to produce receive clock: Output T2 is enabled, interrupt disabled, continuous operation, dual 8-bit counting mode and clock on E input is selected. This may be accomplished by writing, say, 10000111 into control register 2 of the PTM.
4. The clock rate is selected by writing $00_{(16)}$ into the MSB buffer and $06_{(16)}$ into the timer 2 LSB latches.

This programming will make the PTM produce a receiver clock on output T2 with a low time of one ∅ cycle time and a high time of six ∅ cycles, i.e. with a rate of about one seventh of 1.065 MHz which is about equal to 16 times the Receive data bit rate (9 600 Hz).

Line Activity Logic of ACA, ACA-A and -C

The line activity logic in the ACA, ACA-A and -C contains two monostable flip-flops. These are triggered by 0 to 1 transitions on Received data and Transmitted data respectively. They are retriggerable and active during about 275 ms after the last triggering. The received data monostable flip-flop turns, when set, on an 8 Hz signal to a LED, which then will blink.

A transmitted data latch is preset when the transmitted data monostable flip-flop is activated and is reset 16 s after the monostable flip-flop is deactivated. The LED will glow steadily as long as the transmitted data latch is set. The output of the latch can also be checked by the program via PIA PA3.

Printer Interface of ACA, ACA-A, -B and -D

It is, under this heading, assumed that the ACA and ACA-A are strapped for printer operation.

The V.24 printer interface consists of the following logical signals:

- Data to printer (±12 V), originating from the Transmitted data output, TXD, of the ACIA.
- Enable, originating from the Request to send, RTS, output of the ACIA. The signal is activated when the ACIA control register bits 6 and 5 are set to $00_{(2)}$, $01_{(2)}$ or $11_{(2)}$.
- Ready A and B. A is fed to the PIA CA1 and PA7 inputs, B to CA2 and PA6 inputs. Printer status can be indicated via these two lines and the PIA can be programmed to react with an interrupt request. If the printer is turned off or not connected, the signals will be logical 1 on the PIA inputs.
- Busy. This signal functions in different ways on the ACA board and the ACA-A, -B and -D boards.

  The following is valid for the ACA. Busy is fed to the Clear to send, CTS, input of the ACIA, and will thus inhibit an interrupt request when the transmitted data register is empty. This path is called busy logic 2. Note that two bytes, one in the shift register and one in the transmitted data register may be transmitted after the activation of the Busy signal. An additional path for the Busy signal is therefore provided and enabled if PIA PB0 = 0. An active Busy will then inhibit the transmitter clock to reach the Transmitter clock, TXC, input of the ACIA as soon as the Transmitted data, TXD, output goes high, i.e. latest at the first stop bit of the byte just being shifted out. This path is called busy logic 1 and will of course override busy logic 2.

  The following is valid for the ACA-A, -B and -D. The active polarity of Busy can be selected by PIA PB4. If not inhibited by PIA PB3 = 1 the Busy signal will then give the same functions as said above about the ACA.

  The transmission starts again when the Busy condition ends.

- Parity error (or Received data, see below). A parity error shall be indicated by the printer with a positive transition in the middle of the parity bit of the erroneous character. This will set a parity error flip-flop and thereby inhibit the transmitter clock (to the TXC input of the ACIA) at the following stop bit. The flip-flop is reset by PIA PA0 = 0 and, in the ACA-A, -B and -D, also by PIA PB3 = 1.

  The Parity error signal is also directly fed to PIA CB1 and PB7 and may thus lead to an interrupt request and an indication mark to the microcomputer. Note that interrupts from PIA CB1 must be inhibited by the PIA programming if the parity error function is not used.

- Received data (or Parity error, see above). This line can be used for receiving asynchronous serial data instead of parity error in the ACA-A, -B and -D if PIA PB3 = 1. (The data is then fed to the Received data, RXD, input of the ACIA.)

Only internal clocks (from the PTM) are used in the printer interface.

Fibre Optic Printer Interface of ACA-C

It is, under this heading, assumed that the ACA-C is strapped for printer operation.

The external interface contains an optical output for transmitting and an optical input for receiving. The following signals are transferred in time multiplex via the optical output:

- Data to printer, originating from the Transmitted data output, TXD, of the ACIA.

- Enable, originating from the Ready to send, RTS, output of the ACIA. The signal is activated when the ACIA control register bits 6 and 5 are set to $00_{(2)}$, $01_{(2)}$ or $11_{(2)}$. Enable can for instance be used to start a printer motor.

Note that the ACA-C shall be strapped for transmission on two channels.

The following signals are transferred in time multiplex via the optical input:

- Ready A and B. A is fed to the PIA CA1 and PA7 inputs, B to CA2 and PA6 inputs. Printer status can be indicated via these two lines and the PIA can be programmed to react with an interrupt request. If the printer is turned off or not connected, the signals will be logical 0 on the PIA inputs.

- Busy. The active polarity can be selected by PIA PB4 or by strapping. (PB4 or strap shall be set to 0 if Busy is not used.)

  The Busy signal is fed to the Clear to send, CTS, input of the ACIA and will thus inhibit an interrupt request when the transmitted data register is empty. This path is called busy logic 2. Note that two bytes, one in the shift register and one in the transmitted data register may

be transmitted after the activation of the Busy signal. An additional path for the Busy signal is therefore provided and enabled if PIA PB0 = 0. PIA PB3 must also be 0 or disconnected in the strapping with the strap connected to 0 V. An active Busy will then inhibit the transmitter clock to reach the Transmitter clock, TXC, input of the ACIA as soon as the Transmitted data, TXD, output goes high, i.e. latest at the first stop bit of the byte just being shifted out. This path is called busy logic 1 and will of course override busy logic 2.

The transmission starts again when the active Busy condition ends.

- Parity error, PE (or Received data, see below). If PIA BP3 = 0 (and is strapped effective) or the parity error function is strapped constantly active an parity error indication will set a parity error flip-flop and thereby inhibit the transmitter clock (to the TXC input of the ACIA) at the following stop bit. The flip-flop is reset by PIA PA0 = 0.

  The Parity error signal is also (depending on PB3 and strapping) directly fed to PIA CB1 and PB7 and may thus lead to an interrupt request and an indication mark to the microcomputer.

- Received data (or Parity error, see above). This signal can be used for receiving asynchronous serial data instead of parity error if PIA PB3 = 1 (and is strapped effective) or the receiving function is strapped constantly active. The data is then fed to the Received data, RXD, input of the ACIA.

An additional signal is generated in the signal detector (see Fibre Optic Interface Functions) and is fed to the Data carrier detect, DCD, input of the ACIA if strapped accordingly. Thus, the ACIA can be programmed to make an interrupt request on the same conditions as the indicator LED is lit.

Modem Interface of ACA and ACA-A

It is, under this heading, assumed that the ACA and ACA-A are strapped for modem/current loop operation.

The CCITT V.24/V.28 modem interface consists of the following signals:

- Transmitted data, TD, output fed from the Transmitted data, TXD, output of the ACIA.
- Transmitter signal element timing, TSET, input, i.e. the external transmitter clock fed to a clock selector. Either of the external or internal (see below) transmitter clocks must be selected.
- Transmitter signal element timing, TSET out, output, i.e. the internal transmitter clock fed from the O1 output of the PTM.
- Request to send, RTS, output fed from the Request to send, RTS, output of the ACIA.
- Data set ready, DSR, input fed to the PB6 input of the PIA in the ACA-A. Not used in the ACA.
- Ready for sending, RFS, input fed to the Clear to send input of the ACIA.

- Received data, RD, input fed to the Received data, RXD, input of the ACIA.
- Receiver signal element timing, RSET, input, i.e. the external receiver clock fed to the clock selector.
- Data carrier detector, DCD, input fed to the Data carrier detected, DCD, input of the ACIA if PIA PB4 = 0. Only used in the ACA-A.
- Data terminal ready, DTR, output fed from the PA4 output of the PIA.
- Data signalling rate selector, DRS, output fed from the PB0 output of the PIA in the ACA-A. Not used in the ACA.
- Calling indicator, CI, input fed to the CB1 and PB7 inputs of the PIA in the ACA-A. Not used in the ACA.

Current Loop Interface of ACA-A

It is, under this heading, assumed that the ACA-A is strapped for modem/ current loop operation.

The current loop interface consists of the following logical signals:

- Passive transmitted data, PTD, output fed from the Transmitted data, TXD, output of the ACIA.
- Passive received data, PRD, input fed to the Received data, RXD, input of the ACIA.
- Active transmitted data, ATD, output fed from the Transmitted data, TXD, output of the ACIA.
- Active received data, ARD, input fed to the Received data, RXD, input of the ACIA.

Personal Computer Interface of ACA-D

Note that there is a separate programmable timer module, PTM, asynchronous communication interface adapter, ACIA, and address decoder for the personal computer interface but there is no peripheral interface adapter PIA and no connections to the PIA of the printer interface of the board.

The personal computer interface consists of the following logical signals:

- Transmitted data, TXD, output fed from the Transmitted data, TXD, output of the ACIA.
- Transmitter signal element timing, TSET, input, i.e. the external transmitter clock that (if strapped so) is fed to the Transmitter clock, TXC, input of the ACIA.
- Transmitter signal element timing, TSET out, output, i.e. the internal transmitter clock that (if strapped so) is fed from the O1 output of the PTM.

- Request to send, RTS, and Data terminal ready, DTR, outputs, both fed from the Request to send, RTS, output of the ACIA.
- Ready for sending, RFS, input fed to the Clear to send, CTS, input of the ACIA.
- Received data, RXD, input fed to the Received data, RXD, input of the ACIA.
- Receiver signal element timing, RSET, input, i.e. the external receiver clock that (if strapped so) is fed to the Receiver clock, RXC, input of the ACIA.

Appendix 1

ACA I/O Addresses, $F7A7_{(16)} - F790_{(16)}$, $F767_{(16)} - F750_{(16)}$

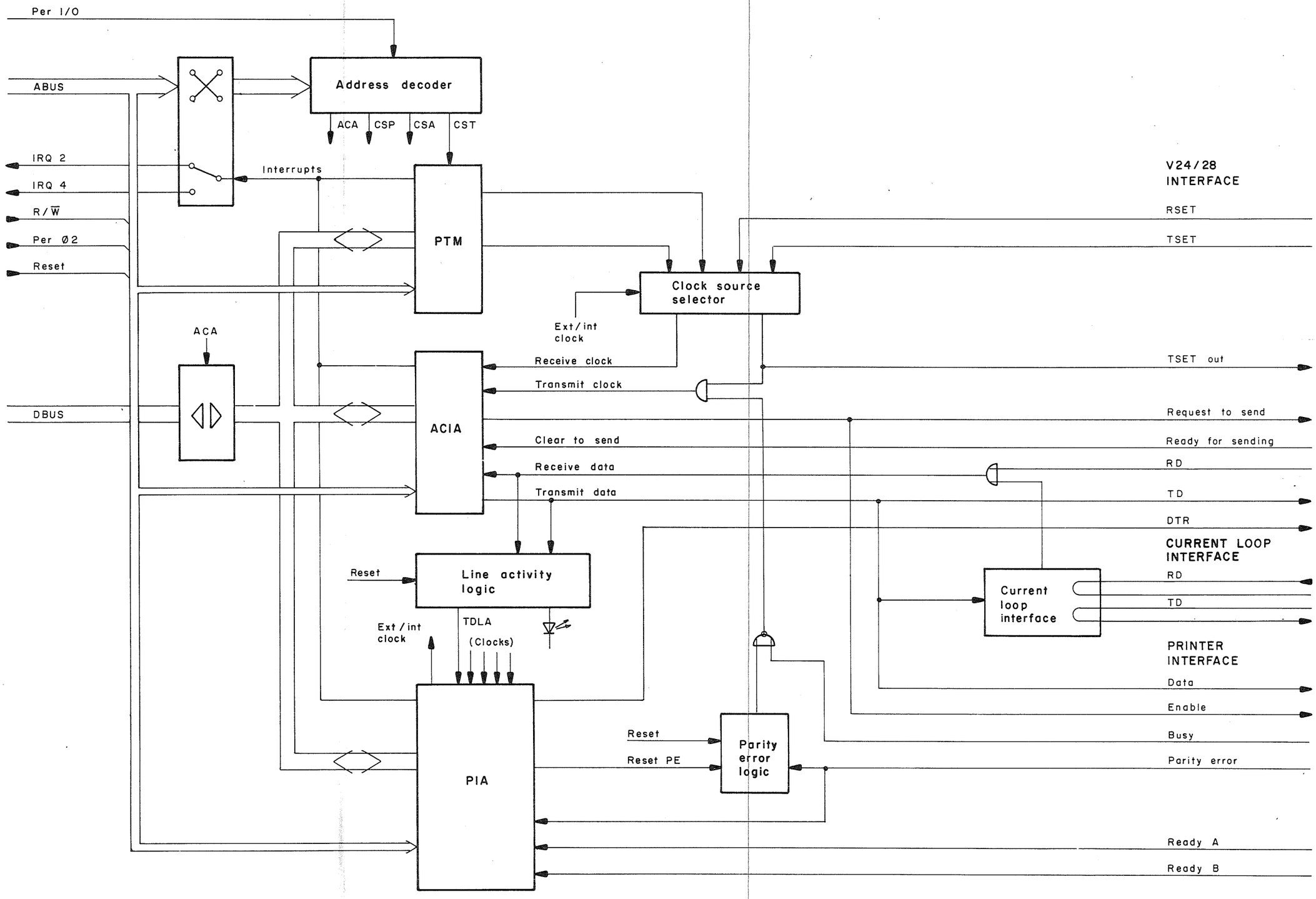
This is only valid when the ACA boards are used in S41 display units.

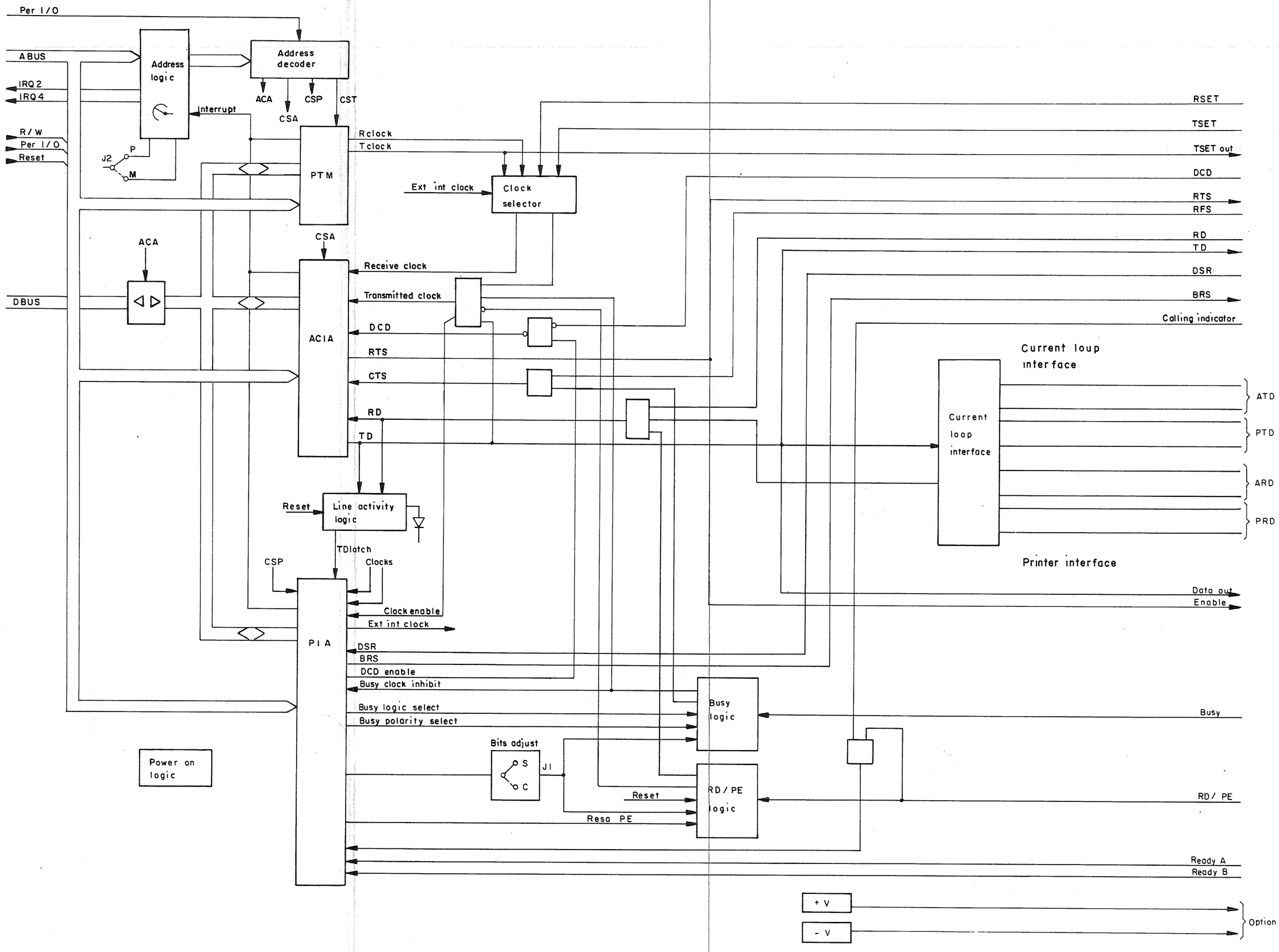
(The first two figures, $F7_{(16)}$, are omitted in the address column below. R/W = W means write, R/W = R means read and R/W = R/W means read or write as seen from the MPU.)

Address (Hex)	R/W	Other condition	Addressed unit and register
			ACA ACIA (Modem interface, not used in ACA-B and -D)
A7	→		= A1
A6	→		= A0
A5	→		= A1
A4	→		= A0
A3	→		= A1
A2	→		= A0
A1	W		Transmitted data register
A1	R		Received data register
A0	W		Control register
A0	R		Status register
			ACA PTM (Modem interface, not used in ACA-B and -D)
9F	W		Timer 3 LSB latches
9F	R		LSB buffer (timer 3)
9E	W		MSB buffer (timer 3)
9E	R		Timer 3 counter MSB
9D	W		Timer 2 LSB latches
9D	R		LSB buffer (timer 2)
9C	W		MSB buffer (timer 2)
9C	R		Timer 2 counter MSB
9B	W		Timer 1 LSB latches
9B	R		LSB buffer (timer 1)
9A	W		MSB buffer (timer 1)
9A	R		Timer 1 counter MSB
99	W		Control register 2 (CR2)
99	R		Status register
98	W	CR2:0 = 0	Control register 3
98	W	CR2:0 = 1	Control register 1
			ACA PIA (Modem interface, not used in ACA-B and -D)
97	R/W		= 93
96	R/W	→	= 92
95	R/W		= 91
94	R/W	→	= 90
93	R/W		Control register B (CRB)
92	R/W	CRB2 = 0	Data direction register B
92	R/W	CRB2 = 1	Peripheral register B
91	R/W		Control register A
90	R/W	CRB2 = 0	Data direction register A
90	R/W	CRB2 = 1	Peripheral register A
			ACA ACIA (Printer interface)
67	→		= 61
66	→		= 60
65	→		= 61
64	→		= 60
63	→		= 61
62	→		= 60
61	W		Transmitted data register
61	R		Received data register
60	W		Control register
60	R		Status register

Address (Hex)	R/W	Other condition	Addressed unit and register
			ACA PTM (Printer interface)
5F	W		Timer 3 LSB latches
5F	R		LSB buffer (timer 3)
5E	W		MSB buffer (timer 3)
5E	R		Timer 3 counter MSB
5D	W		Timer 2 LSB latches
5D	R		LSB buffer (timer 2)
5C	W		MSB buffer (timer 2)
5C	R		Timer 2 counter MSB
5B	W		Timer 1 LSB latches
5B	R		LSB buffer (timer 1)
5A	W		MSB buffer (timer 1)
5A	R		Timer 1 counter MSB
59	W		Control register 2 (CR2)
59	R		Status register
58	W	CR2:0 = 0	Control register 3
58	W	CR2:0 = 1	Control register 1
			ACA PIA (Printer interface)
57	R/W		= 53
56	R/W	→	= 52
55	R/W		= 51
54	R/W	→	= 50
53	R/W		Control register B (CRB)
52	R/W	CRB2 = 0	Data direction register B
52	R/W	CRB2 = 1	Peripheral register B
51	R/W		Control register A
50	R/W	CRB2 = 0	Data direction register A
50	R/W	CRB2 = 1	Peripheral register A

MICRO COMPUTER
(DTC) INTERFACE

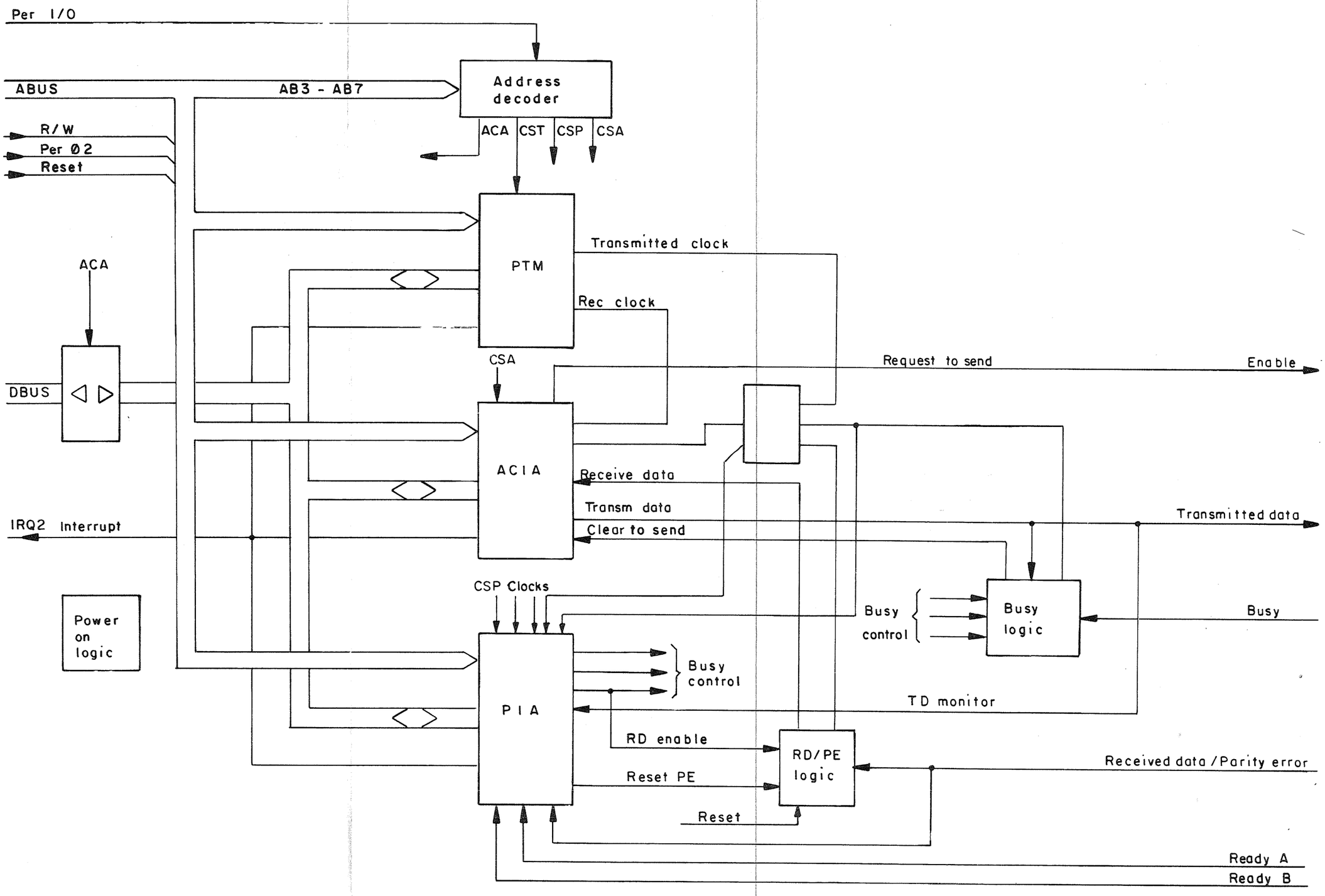




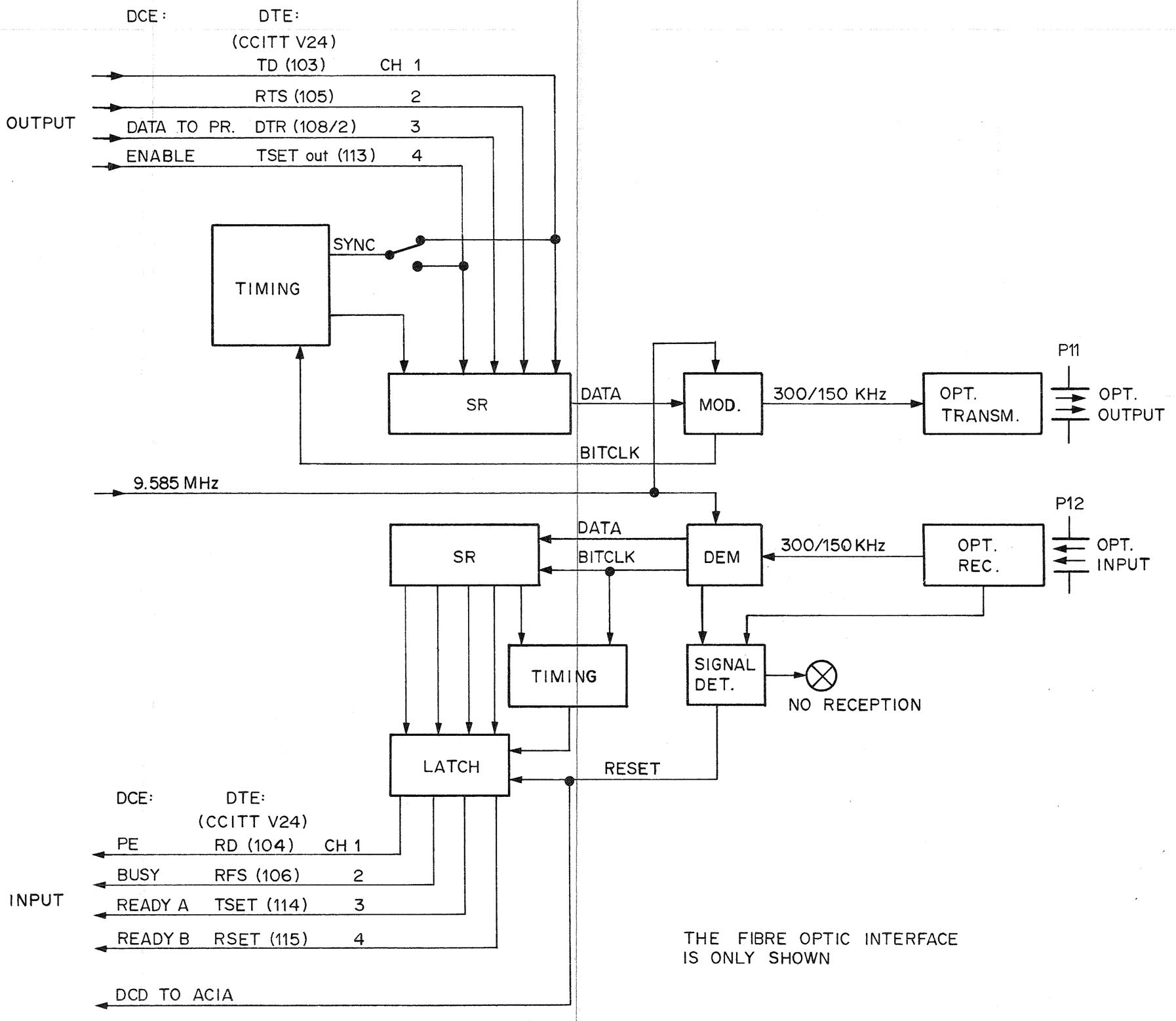
EE366-810B

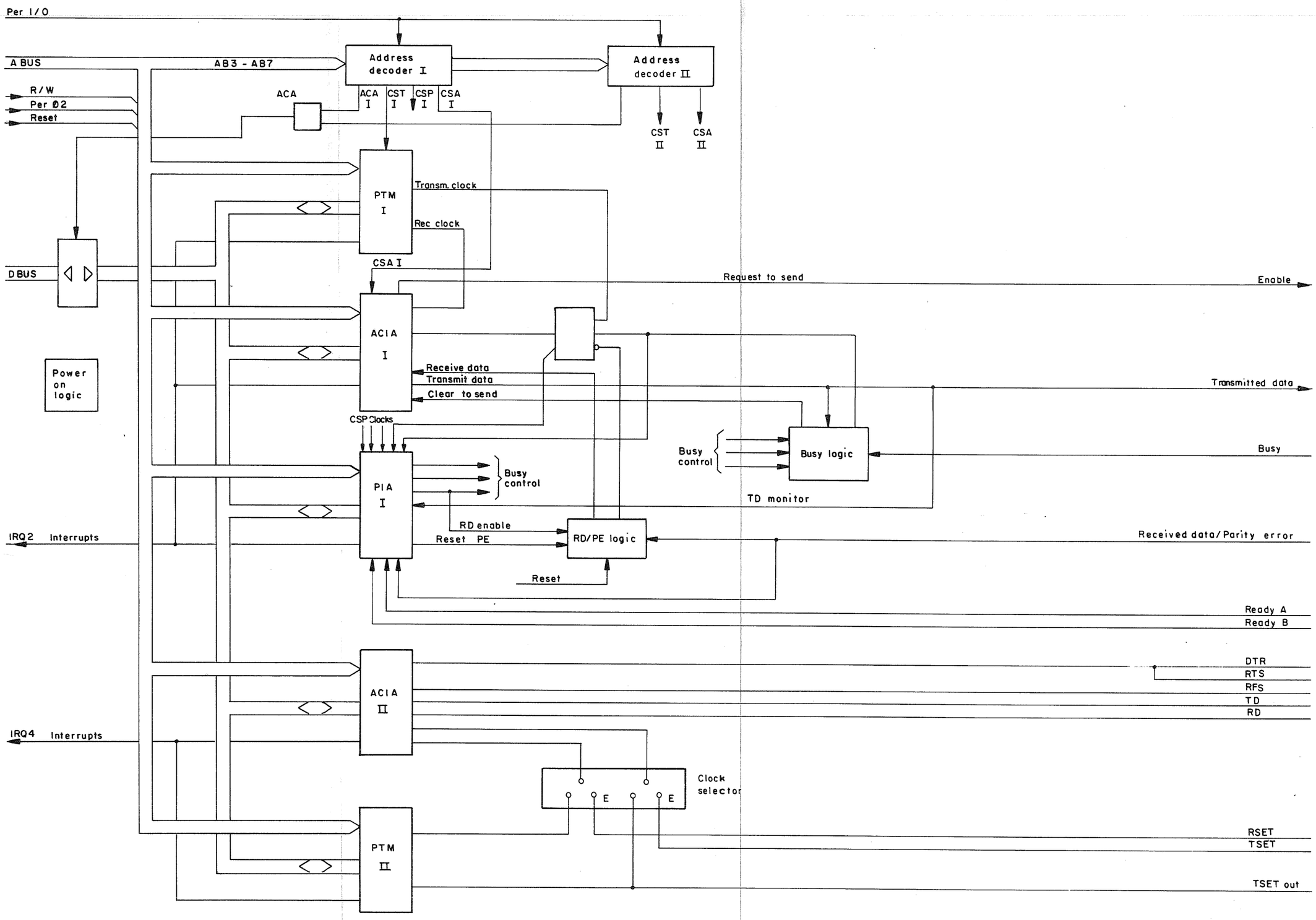
Block Diagram, ACA-A

Appendix 3



Block Diagram, ACA-B
Appendix 4


DCE:
DTE:
(CCITT V24)
TD (103)
CH 1
RTS (105)
2
OUTPUT
DATA TO PR.
DTR (108/2)
3
ENABLE
TSET out (113)
4
TIMING
SYNC
SR
DATA
MOD.
300/150 KHz
OPT. TRANSM.
P11
OPT. OUTPUT
BITCLK
9.585 MHz
P12
DATA
SR
BITCLK
DEM
300/150 KHz
OPT. REC.
OPT. INPUT
TIMING
SIGNAL DET.
NO RECEPTION
LATCH
RESET
DCE:
DTE:
(CCITT V24)
PE
RD (104)
CH 1
BUSY
RFS (106)
2
INPUT
READY A
TSET (114)
3
READY B
RSET (115)
4
DCD TO ACIA
THE FIBRE OPTIC INTERFACE
IS ONLY SHOWN



EE366-810B

Block Diagram, ACA-D

Appendix 6

DATASAAB

Memory Board, R/W

Contents

Figure

GENERAL

The MRW, memory board, read/write memory, is used in the display unit or the communication processor when more RWM (read/write memory) than 32 kbytes is needed. The board may be equipped with 16 or 32 kbytes RWM. A strap is provided to make 28, 24 or 20 kbytes of the memory available. (Four kbytes are used by each synchronous communication controller, SCC). See memory map of DU and CP for further information.

The MRW is coupled to the external address and data buses of the microcomputer.

DYNAMIC RWM AND ROW/COLUMN/REFRESH ADDRESS SELECTOR

Addressing from Address Bus

The RWM chips used contain 16 kbits organized in a 128 rows by 128 columns matrix. (To provide 16 kbytes eight chips are coupled equally i.e. activated by the same strobe signals.) The 16 kbits are organized as 128 rows by 128 columns. As can be seen in the block diagram, there are only seven address lines to the RWM. These lines are used both for row and column address. Thus, to select a specific byte, at first the row address is applied and strobed in by one of the two Row address strobes and then the column address is applied and strobed in by the Column address strobe.

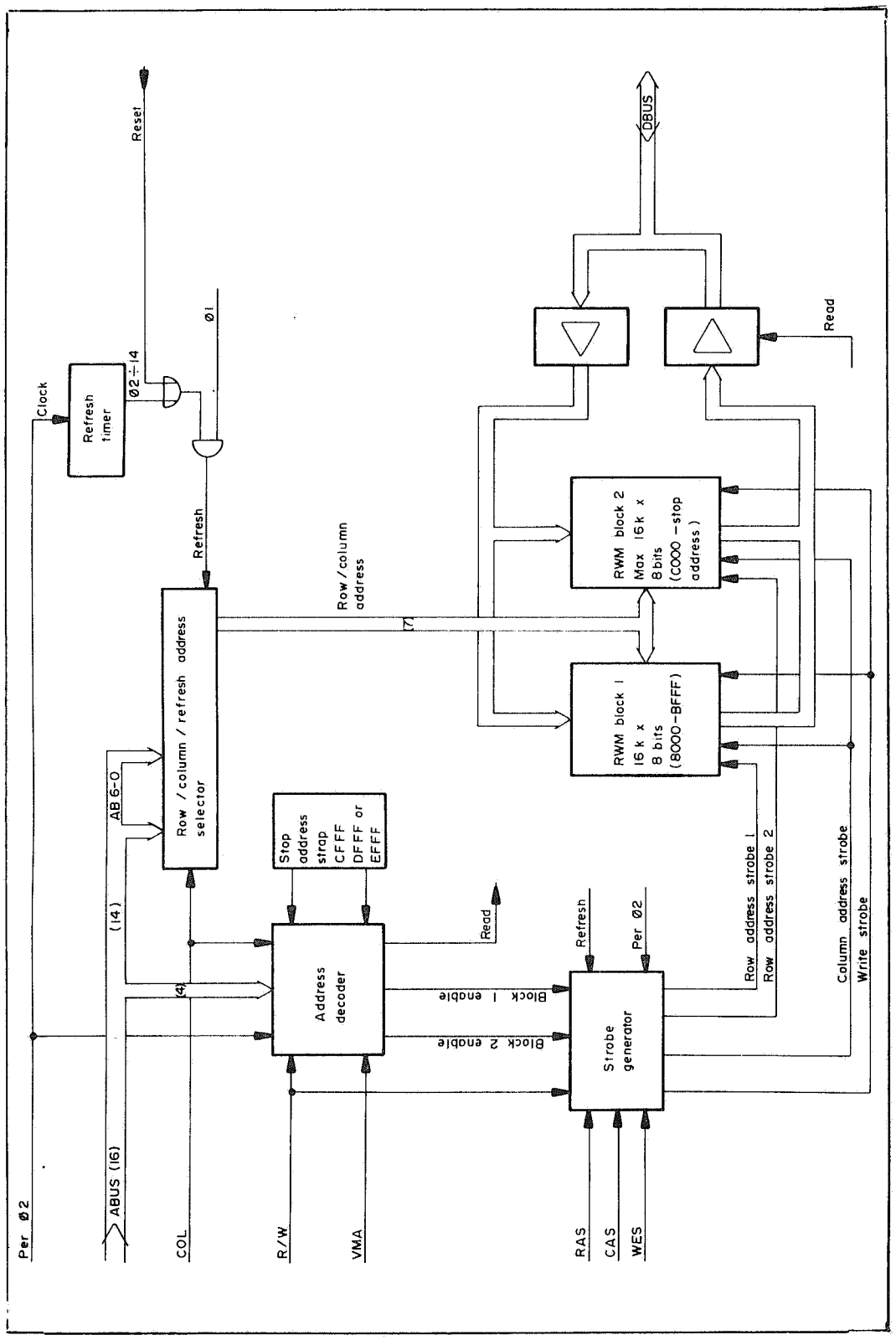


Fig. 1. MRW block diagram

Refresh Addressing

The dynamic RWM that is used needs periodic addressings if it is not to lose its contents. As one cannot be sure that the MPU or DMA logic addresses the cells at regular intervals, a refresh logic is needed. The row/column/refresh address selector contains a divide by 128 refresh address counter which provides consecutive row addresses at periodic intervals. Note that only the row addressing is needed to refresh the memory. When the microprocessor is operating (Not Reset), one refresh address is provided every fourteenth clock cycle. During Reset, one refresh address is applied every clock cycle, to prepare the memory for operation. The function of the Refresh signal is to

- Select the row address from the refresh address counter.
- Step the refresh address counter.
- Make the strobe generator provide Row address strobes for both RWM blocks at the same time.

The refresh addressing takes place during the Ø1 half of the system clock cycle and MPU or DMA addressing takes place during the Ø2 half of the clock cycle. Thus the periodic refresh addressing does not interfere with memory accesses from the bus.

ADDRESS DECODER

The address decoder contains a ROM. From the four most significant address bus bits, the R/W signal and the two lines defining the memory area stop address, the address decoder ROM judges which enable signals that should be provided. In short, Block 1 enable is generated if A15 = 1 and A14 = 0. Block 2 enable is generated if A15 = 1 and A14 = 1.

A13 and A12 are used together with the stop address lines to define whether Block 2 enable should be provided. E.g. when the stop address is CFFF only the combination A15 · A14 · A13 · A12 = 1100 should activate the Block 2 enable.

The Read pulse is in the same way generated for addresses between 8000 and the strapped stop address when the R/W signal is in the read state.

All three enable signals are generated during Ø2 (Bus time).

STROBE GENERATOR

- Write strobe is generated during the later part of the Per. Ø2 (Peripheral Ø2) period i.e. during the period when MPU or DMA is working on the buses.
- During Refresh (Ø1 period) the Row address strobes 1 and 2 are both generated. One of them is generated during Ø2 if one of the Block enable signals is present, i.e. if a byte in the defined memory area of the MRW board is addressed from the bus.
- The Column address strobe is only generated during the Ø2 period and if one of the Block enable signals is present.

Note that a Column address strobe alone does not affect the contents of the memory. Neither does a Write strobe without preceding Row address and Column address strobes.

For further details of timing, see paragraph "Timing" in the "Microcomputer" chapter.

DATASAAB

Memory Board, RO

Contents

Figures

EE368-810
1978-11-24

GENERAL

The MRO (Memory board, read-only memory) is used in display unit or communication processor to add program memory capacity to the basic memory of max 32 kbytes and IPL ROM of 2 kbytes. The MRO may be equipped with a maximum of 24 kbytes read-only memory.

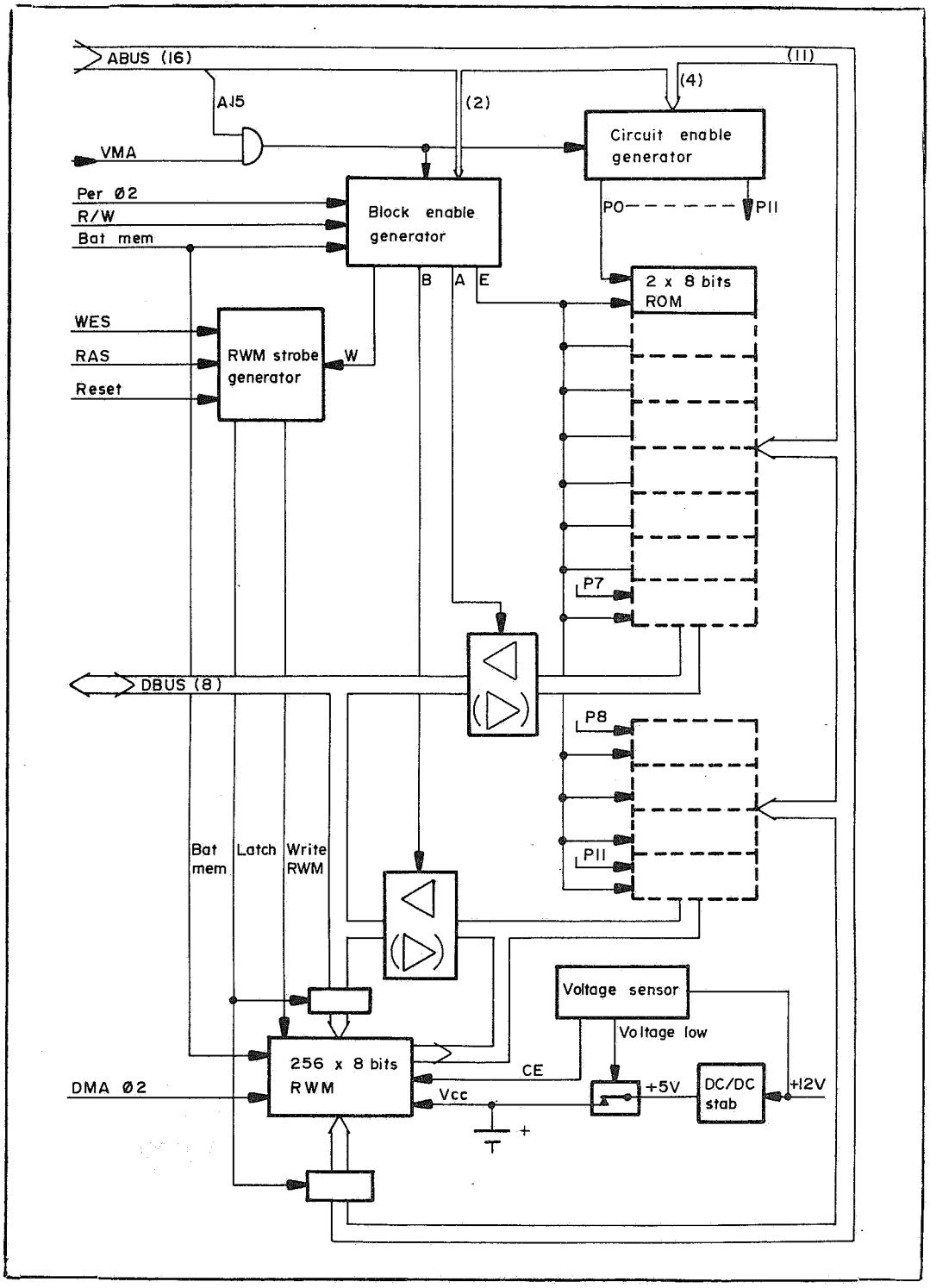


Fig.1. MRO block diagram

Each memory chip contains 2 kbytes (2k x 8 bits) of memory, and the RO memory area is thus expandable from 2 kbytes to 24 kbytes in steps of 2 kbytes. Furthermore, the MRO holds 256 bytes of static RWM, typically used for software strappings. The RWM should not be used for program storage.

Besides memory circuits and bus buffers, the MRO contains three groups of strobe generating circuits;

- Block enable generator consisting of a ROM, providing enable signals for the output buffers and the associated ROM circuit outputs.

- Circuit enable generator, consisting of two decoder circuits, providing different enable signals for the twelve possible ROM circuits.

- RWM strobe generator, only activated at write RWM operations, causing data and address for the RWM to be latched and then data to be written into the addressed RWM cell.

DETAILED DESCRIPTION

Addressing. Block Enable Signals

The locations of the different memory circuits of MRO in the total microcomputer memory map is shown in Fig. 2. The occurence of some enable signals is also indicated. The block enable signals, A, B, C and W, are all generated during the Per. Ø2 time. A, B and C are read strobes. Note that C is generated for read from RWM operations (Bat. mem. signal active and R/W in the high (read) state). W is activated for write operations (R/W = low) into the RWM (Bat. mem. signal active).

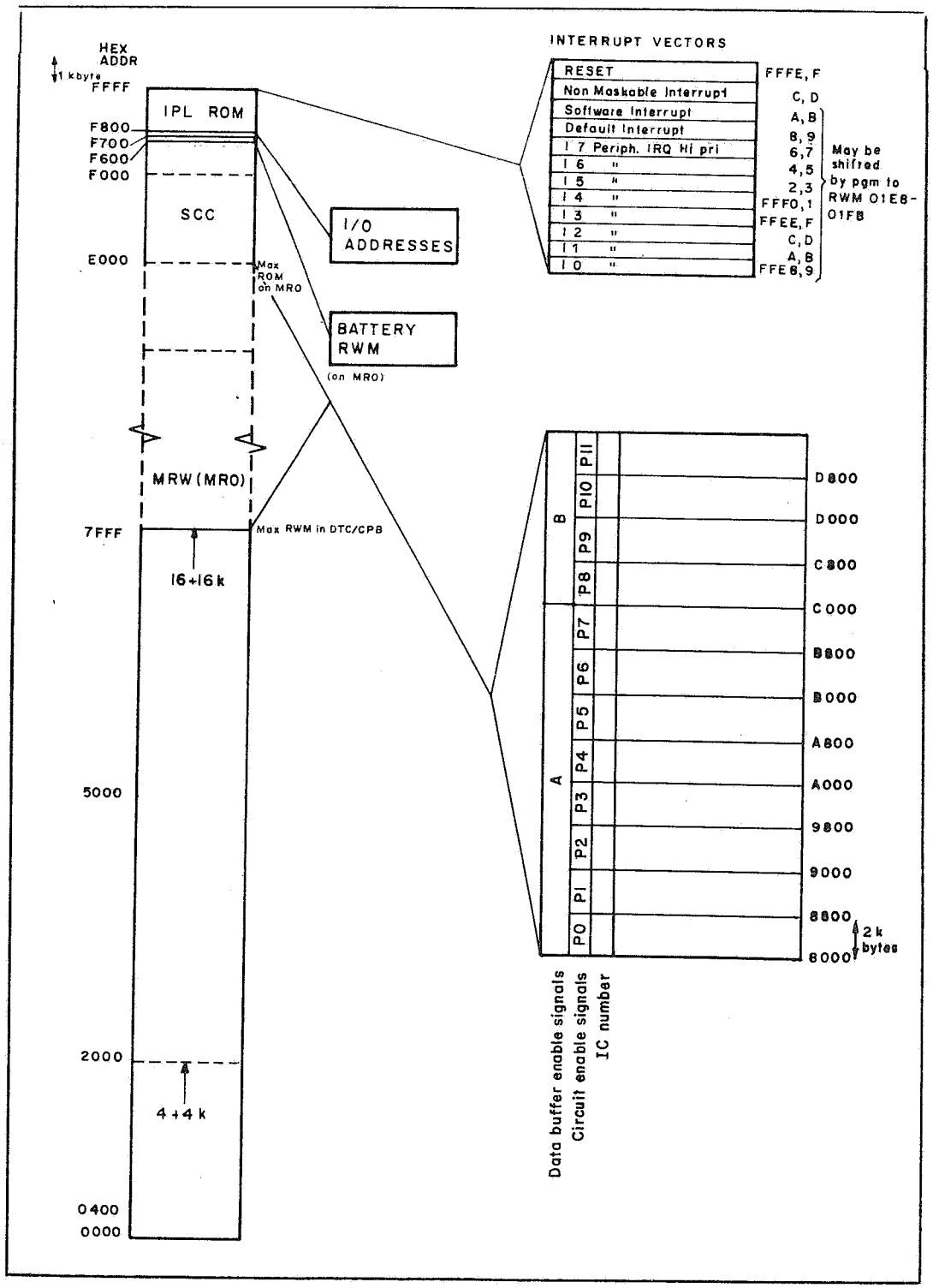


Fig. 2. Memory organization of basic controller and MRO

RWM Write Timing

The generation of the Latch and Write RWM signals is illustrated in Fig. 3.

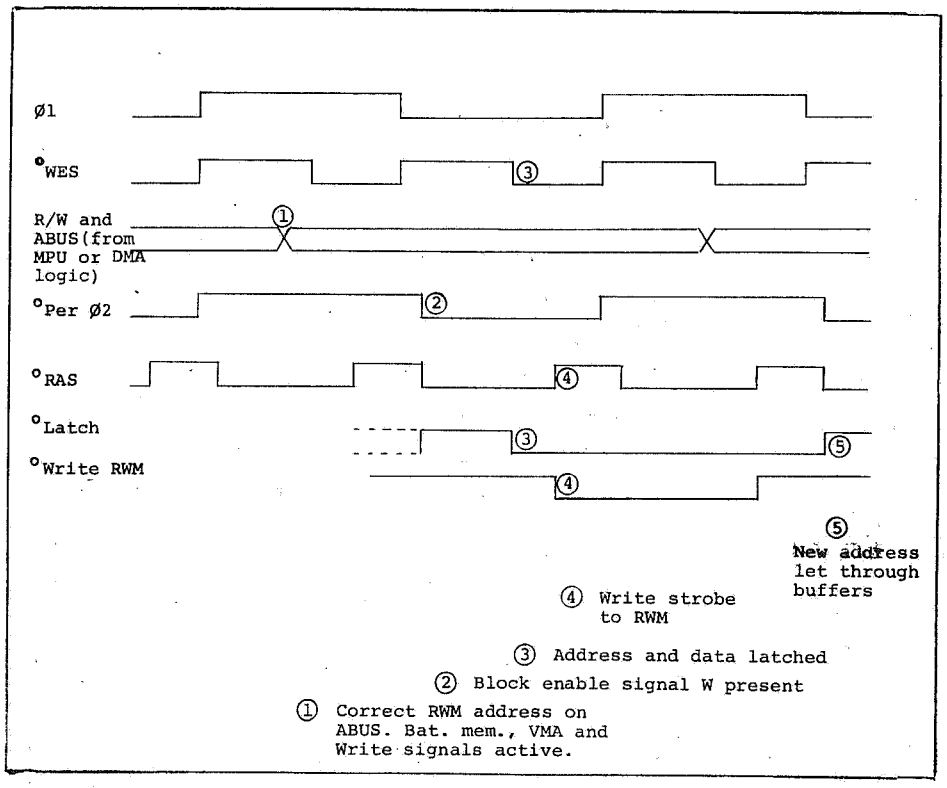


Fig. 3. RWM write cycle

As the needed data and address setup and hold times for write operations into the static RWM circuits are quite long a write cycle (Ø) intrudes on next clock cycle. Consequently, the RWM should not be used so that it could be affected by a read cycle directly following a write cycle. This is because the access time for read operations is longer than the time between point (5) in Fig. 3 and next trailing edge of Ø2. The sequence write-read may only occur if the RWM area is used both for register (e.g. stack) purposes and to hold program.

Power Supplies

Contents

Brief Outline

A power supply converts the mains AC voltage to a number of DC voltages which are required in the units.

The power supplies described here are of the type SMPS (Switch Mode Power Supply).

Common for all switching power supplies is, that the incoming voltage is rectified and then chopped to a high frequency square wave voltage. The regulation of the final output voltage is made in the chopper by varying the pulse width, and thus the duty cycle, of the square wave pulses, see Fig. 1. The pulse width varied voltage is transformed to suitable levels and then rectified to get the required DC output voltages.

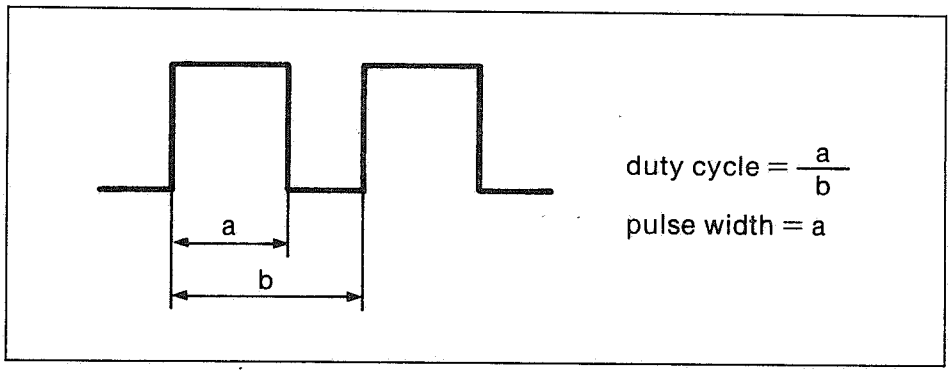


Fig. 1. Definition of duty cycle

There are a number of different power supplies. In some cases they contain the same circuit board but have different mechanical layouts to suit the units they are built into. In other cases they differ electrically, e.g. by means of input voltages, output voltages, internal frequency, or temperature requirements. See further under Technical Data.

A complete list of the power supplies, their circuit boards and the units they are included in, is enclosed in Appendix 1.

The description of the power supplies is divided into two sections, each one describing a group of power supplies with similar principle design and properties.

Section 1

The power supplies with the following circuit diagrams can be regarded as one "family" with principally the same design and with the same numbering of components:

E34068 2001	E34068 2010	E34068 2040
E34068 2004	E34068 2011	E34068 2050
E34068 2005	E34068 2030	E34068 2060

These power supplies are described together, see the block diagram, Fig. 2.

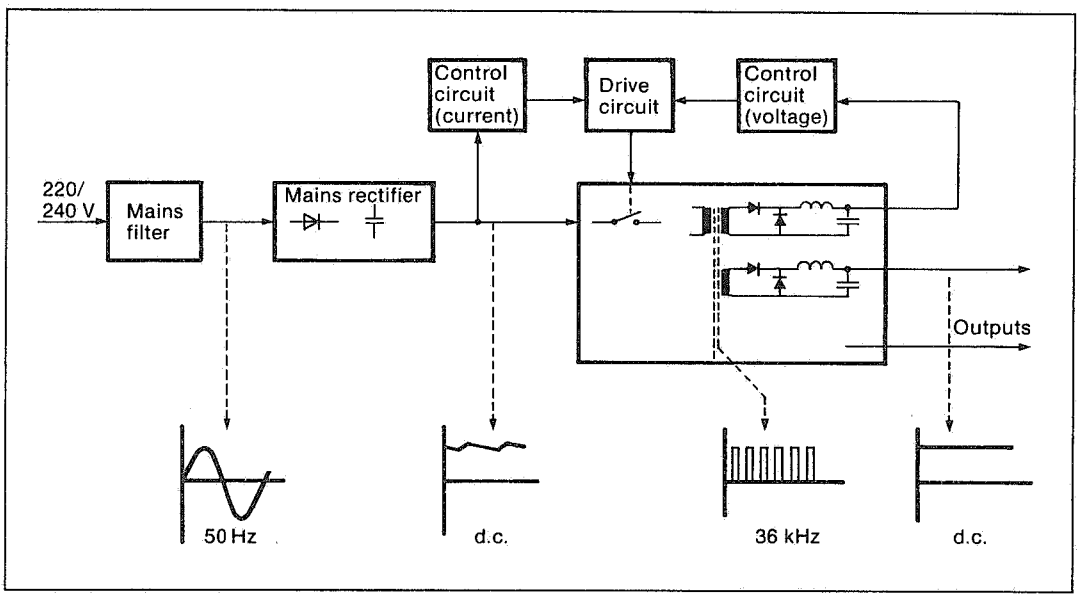


Fig. 2. Block diagram, Section 1 power supplies

The mains voltage is rectified and smoothed in the mains rectifier. The rectified voltage is converted to a 36 or 40 kHz square wave voltage in a chopper (the frequency differs between the power supplies).

The chopped frequency is applied to the primary side of a transformer. The output voltages are taken from the secondary windings of the transformer. They are rectified and filtered before they are provided at the DC output terminals.

The power supply also contains control circuits for voltage and current control.

The voltage control is governed by a voltage from a separate winding of the transformer that gives a correction signal which varies the duty cycle and thus compensates for any change in input voltage.

The current is controlled by a circuit that decreases the duty cycle if the current rises, and finally turns off the power supply if the current rises too much.

A separate circuit is transforming and rectifying the mains voltage to the DC voltage needed for the circuits internally in the power supply.

Section 2

The power supplies ROA 117202/1 – 2, ROA 117203/1 – 2, and BMJ 160003/1 (ROA 117205/1) belong to another "family" of power supplies with the same design principle.

See the block diagrams, Fig. 3 for ROA 117202/1 – 2, Fig. 4 for ROA 117203/1 – 2 and Fig. 5 for BMJ 160003/1 (ROA 117205/1).

The mains voltage is filtered, rectified and smoothed in the mains filter and rectifier. The rectified voltage is converted to an about 45 – 50 kHz square wave voltage by the main transistor.

The chopped voltage is applied to the primary side of a transformer. The output voltages are taken from the secondary windings of the transformer. They are rectified and filtered before they are provided at the DC output terminals.

A circuit senses the output voltages and gives a signal via an optocoupler to the control circuit, which adjusts the pulse width and thus the primary voltage. The control circuit also contains the oscillator that determines the switching frequency.

The power supply ROA 117203/1 – 2, see Fig. 4, has a monitoring circuit that gives a warning signal (PW) at first and then a reset signal (PR) if the mains voltage or the output voltages are too low.

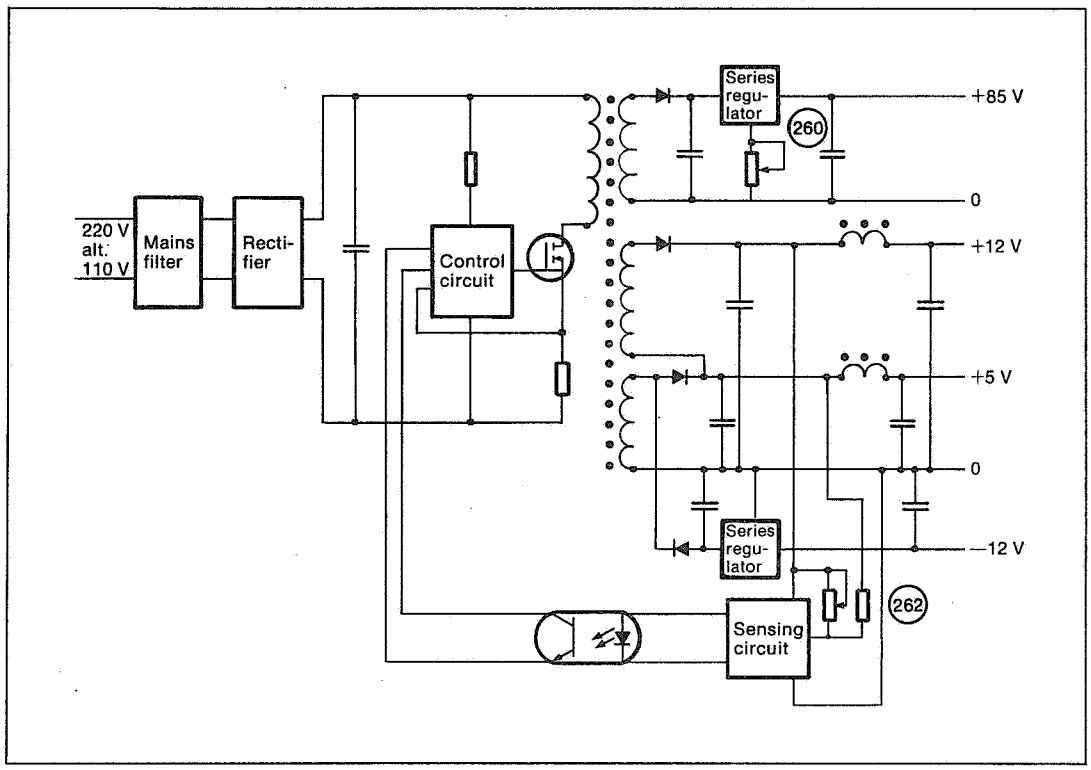


Fig. 3. Block diagram, ROA 117202/1 – 2

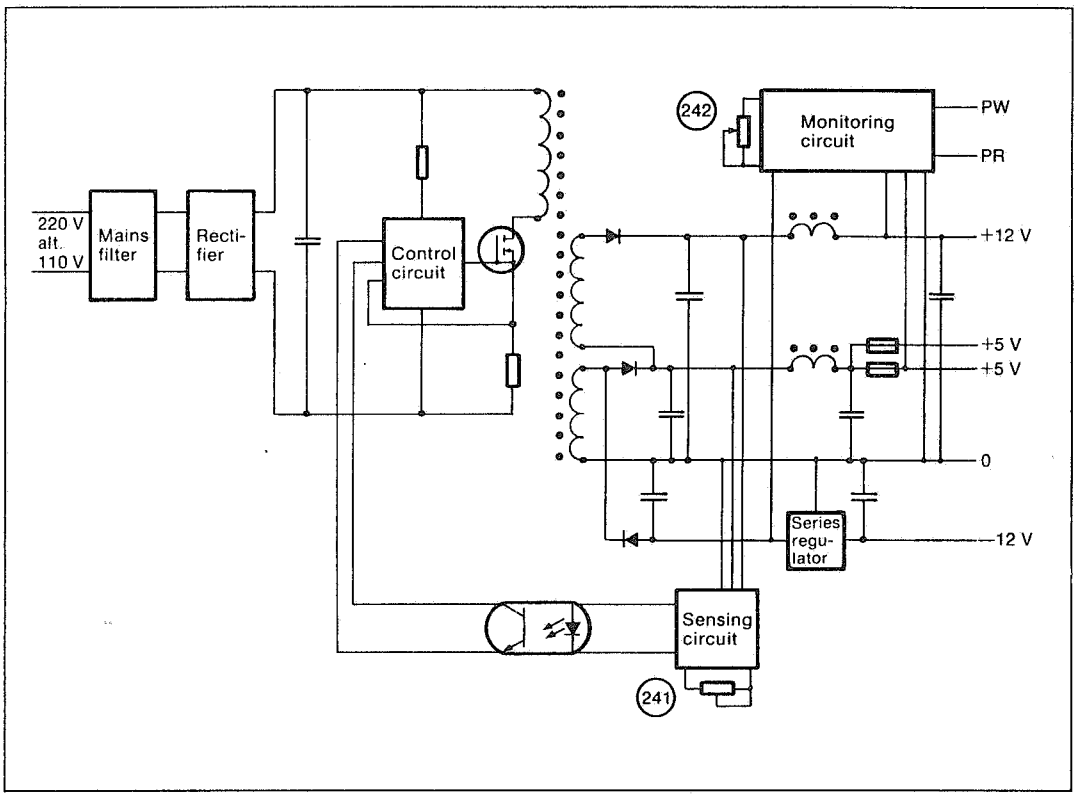


Fig. 4. Block diagram, ROA 117203/1 – 2

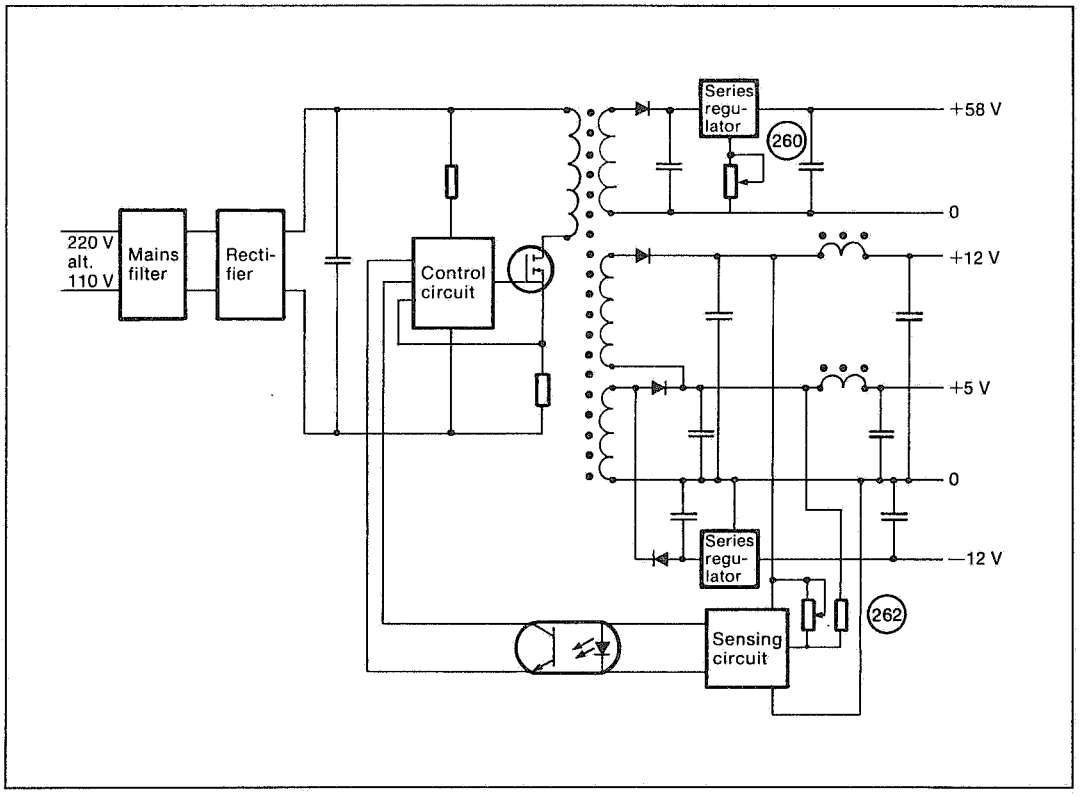


Fig. 5. Block diagram, BMJ 160003/1 (ROA 117205/1)

Detailed Description

Section 1

The following description refers to circuit diagrams:

E34068 2001	E34068 2010	E34068 2040
E34068 2004	E34068 2011	E34068 2050
E34068 2005	E34068 2030	E34068 2060

These circuit diagrams are designed with the same component denomination, in general, and have, in principle, the same function. The differences are described under a separate heading.

Input Rectifier and Filter Circuit

The main voltage is first filtered by the inductor L1 (not included in all power supplies) and then rectified by the bridge rectifier Z1. The rectified current is filtered by the smoothing capacitor C5.

The current is limited at the start-up. This is done with the NTC (negative temperature coefficient) resistor R1 and resistor R2.

Internal DC Voltage

The mains voltage is transformed to about 13 V RMS (Root-Mean-Square = effective voltage) in the transformer T1. It is rectified in Z2 and smoothed to about 15 V DC by C7 and then supplied as drive voltage to the circuits in the power supply.

Depending on whether the mains voltage is 220 or 240 V this voltage should be connected to different points of the primary winding of T1. This is done with contact P5.

Power Switching and Output Stage

The DC voltage from the mains rectifier circuit is chopped by the power transistors Y2 and Y3. These transistors are switching elements which are turned on and off together at a frequency determined by the oscillator in IC1 via the driver circuit (Y3 is excluded in some of the power supplies, see further on).

When the transistors are turned on, current flows through the primary winding of transformer T3.

The output stages are connected for forward converter transformation, which means that when current flows through the primary winding, current will also flow through the secondary winding out to the load. When the switching transistors are off, the stored energy in the output choke will continue to deliver current. The transformer's magnetizing current is returned to the supply through Z7 and Z8 during the off period.

Driver Circuit

Transistor Y1, which is controlled by IC1, drives the flyback transformer T2. The transformer has two windings which are driving the power transistor(s) Y2 (and Y3) that provide the chopped current for the primary winding of transformer T3.

Control Circuits

- Output voltage control. The voltage from the feed back winding of transformer T3 is rectified and used to regulate the output voltages. It gives a correction signal which varies the duty cycle and thus compensates for any change in the input voltage. The normal level of the correction signal is set with potentiometer R6.

- Current limit. The primary current is sensed as a voltage over resistor R25 coupled in parallell with R26 and potentiometer R27. A level for IC1 pin 11 is set with potentiometer R27. If the current passes a certain level, the duty cycle will decrease. If the current passes another, higher, level the output transistors will be turned off immediately. Restart will occur in 0.3 seconds' intervals. When the current level is acceptable the power supply will start again. These functions are governed by IC1 pin 15 and the driver circuit.

- Maximum duty factor. The maximum duty factor is set to about 0.45 by potentiometer R14. (If the duty factor exceeds 0.5 the transformer core will be DC magnetized.)

- Switching frequency. The frequency of the oscillator in IC1 is set by potentiometer R11.

- Soft start. At start of the power supply the duty factor begins at zero and increases slowly up to a level depending on the input voltage. The time constant depends on R12 and C13.

Individual Differences from the Description

- Circuit diagrams E34068 2002, E34068 2004 and E34068 2005.

  These power supplies apply fully to the description. The switching frequency is 36 kHz.

- Circuit diagrams E34068 2010, E34068 2011 and E34068 2060.

  These power supplies have a switching frequency of 40 kHz. The drive circuit is changed (transistor Y3 is replaced by a new circuit) to make the power supply able to work with a higher frequency at a higher input voltage and a higher load. This in turn means that the different inputs for 220/240 V in the internal voltage circuit is no longer necessary. Also the inductor L1 at the input is excluded.

  At the outputs, a circuit (K1, Z22) is added to make sure that the −12 V voltage comes before the other voltages.

- Circuit diagram E34068 2030 and E34068 2040.

  In these power supplies some components are changed to get a higher efficiency, a lower cost and a possibility to use them in a unit without a fan. The output power and the number of output voltages are reduced. An integrated circuit, IC2, provides the pre-start of −12 V. L1 and Y3 are excluded as above.

- Circuit diagram E34068 2050.

  This power supply has a +85 V voltage instead of +24 V. The +85 V circuit contains an over voltage protection, Y4, and a series regulator that consists of Y5, Y3 and IC3. L1 and Y3 are excluded as above.

Section 2

The following description refer to circuit diagrams:

ROA 117203/1 – 2
ROA 117202/1 – 2
BMJ 160003/1 (ROA 117205/1)

The power supplies with these circuit diagrams are designed in principally the same way. Differences are pointed out in the text.

Input Filter and Rectifier

The mains voltage is filtered in the mains filter and rectified in the rectifier bridge. The input capacitor of 220 μF or 330 μF serves as an energy buffer which allows a short power failure of about 20 ms.

Internal DC Voltage

The voltage needed internally for the circuits in the power supply is taken out over the diode 51 (58 in ROA 117203).

Power Switching

The DC voltage from the mains rectifier is chopped by the main transistor and applied to the transformer. The main transistor is a FET transistor. It has a high power gain and needs no special driving circuit, but is driven from the control circuit.

Output Circuits

The output voltages, except the −12 V voltage, are produced in circuits operating according to the fly-back converter principle.

The −12 V voltage ciruit is working at the forward converter principle, which makes it come first at start-up.

Control Circuits

The voltage is controlled via a sensing circuit that senses the +5 V output voltage. The sensing circuit gives a signal via the optocoupler to the voltage control input of the control circuit. The control circuit contains the oscillator that drives the main transistor. The pulse width of the oscillator is changed by the signal from the sensing circuit.

A circuit that limits the maximum pulse width is also included in the control circuit.

The +12 V voltage is controlled together with the +5 V voltage. The −12 V voltage is series regulated by a fixed regulator. The +85/+58 V voltages are series regulated and insulated from other voltages.

The current control is provided by sensing the small voltage over the series resistor 20 (13 in ROA 117203).

The circuit diagram ROA 117202 contains an extra current control circuit, where the output currents are sensed by the transformer 253.

The circuit diagram ROA 117203 contains a monitoring circuit that gives a warning and a reset signal when the mains or output voltages are too low. The mains voltage is sensed indirectly over the capacitor 187 in the −12 V circuit. The reset signal appears about 20 ms after the warning signal.

Degaussing Circuit

The circuit diagram ROA 117202 also contains a degaussing circuit used to demagnetize the cathode ray tube. Its output terminals are 277 and 276. At start-up, when the unit is cold, the PTC (positive temperature coefficient) resistors allow a high current from the degaussing circuit. At power on, the resistors rapidly increase their resistance and in about 0.2 seconds the current has decreased to about 10% of the initial value.

Technical Data

Circuit diagram	Input voltage	Output voltages Max. current	Output power	Efficiency
E34068 2001	198 – 264 V	+5 V, 14 A +12 V, 1 A −12 V, 1 A		Appr. 70%
E34068 2004	198 – 264 V	+5 V, 7 A +12 V, 0.5 A −12 V, 0.3 A +24 V, 2 A +24 V, 2 A		Appr. 70%
E34068 2005	198 – 264 V	+5 V, 14 A +12 V, 1 A −12 V, 0.5 A +24 V, 2 A +24 V, 2 A		Appr. 70%
E34068 2010	198 – 264 V	+5 V, 12 A +12 V, 1.3 A −12 V, 1 A +24 V, 2 A +24 V, 2 A		>65%
E34068 2011	198 – 264 V	+5 V, 12 A +12 V, 1.3 A −12 V, 1 A +24 V, 2 A +24 V, 2 A	Max. 120 W	>65%
E34068 2030	198 – 264 V	+5 V, 4.5 A +12 V, 0.4 A −12 V, 0.1 A +24 V, 1.5 A	70 W	>75%
E34068 2040	99 – 132 V	+5 V, 4.5 A +12 V, 0.4 A −12 V, 0.1 A +24 V, 1.5 A	70 W	>75%
E34068 2050	198 – 264 V	+5 V, 4.5 A +12 V, 0.4 A −12 V, 0.1 A +85 V, 0.8 A, av.	90 W	>65%

Circuit diagram	Input voltages	Output voltages Max. current	Output power	Efficiency
E34068 2060	99 – 132 V	+5 V, 12 A +12 V, 1.3 A −12 V, 1 A −24 V, 2 A +24 V, 2 A	Max. 120 W	>65%
ROA 117202/1	198 - 264 V	+5 V, 7.5 A +12 V, 0.4 A −12 V, 0.1 A +85 V, 0.92 A	125 W	>76%
ROA 117202/2	99 – 132 V	+5 V, 7.5 A +12 V, 0.4 A −12 V, 0.1 A +85 V, 0.92 A	125 W	>76%
ROA 117203/1	198 – 264 V	+5 V, 4.5 A +12 V, 2.5 A −12 V, 0.1 A	55 W	>75%
ROA 117203/2	99 – 132 V	+5 V, 4.5 A +12 V, 2.5 A −12 V, 0.1 A	55 W	>75%
BMJ 160003/1 (ROA 117205/1)	198 – 264 V	+5 V, 4.5 A +12 V, 0.4 A −12 V, 0.25 A +58 V, 0.7 A	70 W	>80%

Appendix 1

Table of Available Power Supplies

(This table will not be updated for each new unit but may nevertheless be helpful and is therefore included. Current information can be found in the S41 Product Catalogue.)

Unit	Part No.	Circuit Diagram	Included in
CPS	E34068 001	CPS E34068 2001	CPR 4101-001/002
FPS	E34068 004	FPS E34068 2002	FD 4120-001/002/003
DPS	E34068 005	DPS E34068 2005	DU 4110-001/002/003
UPS EU 220 V	E34068 0010	UPS E34068 2010	CPR 4101-101/102 DU 4110-101/102/103 FD 4120-101/102/104
DPS EU 220 V	E34110 0210	UPB E34068 2011	CPR 4101-001/002/003 CPR 4101-101 DU 4110-002/003 DU 4110-101/102/103 DU 4110-202/203 FD 4120-101/102/103/104
DPS EU 220 V	E34110 0211	DPB E34068 2030	DU 4111-001 DU 4114-001
DPS US 115 V	E34110 0212	DPE E34068 2040	DU 4111-011 DU 4114-011
DPS EU 220 V	E34110 0213	DPB E34068 2050	DU 4112-001
DPS EIS art. No. 71505033 US 115 V	E34110 0214	ROA 117202/2	DU 4112-011 DU 4113-011/012
DPS EIS art. No. 71505032 EU 220 V	E34110 0215	ROA 117202/1	DU 4113-001/002
DPS US 115 V	E34110 0216	DPB E34068 2060	DU 4110-212/213

Unit	Part No.	Circuit Diagram	Included in
FPS EU 220 V	E34120 0210	UPB E34068 2011	CPR 4101-102/103 CPL 4102-001/002/003 CPR 4103-001/002 FD 4120-001/002/003 CPR 4105-001 CPL 4106-001
FPS US 115 V	E34120 0211	UPB E34068 2060	CPR 4101-013 CPL 4102-013 CPR 4103-011/012 FD 4120-013 CPR 4105-011 CPL 4106 011
FPS EU 220 V	E34122 0200	ROA 117203/1	FD 4122-002 CPR 4105-001 CPL 4106-001 PCU 4171-001
FPS US 115 V	E34122 0210	ROA 117203/2	FD 4122-012 CPR 4105-011 CPL 4106-011 PCU 4171-011
DPPS EU 220 V	E34173 0200	BMJ 160003/1 (ROA 117205/1)	DPU 4173-001