Data Systems

PHILIPS

Service Manual

P856M/P857M CPU

A PUBLICATION OF
PHILIPS DATA SYSTEMS
APELDOORN, THE NETHERLANDS.

PUB. NO. 5122 991 26953

DATE October 1978

Great care has been taken to ensure that the information contained in this handbook is accurate and complete. Should any errors or omissions be discovered, however, or should any user wish to make a suggestion for improving this handbook, he is invited to send the relevant details to:

PHILIPS DATA SYSTEMS
SERV. DOC. AND TRAINING DEPT.
P.O. BOX 245, APELDOORN,
THE NETHERLANDS.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION I

GENERAL DESCRIPTION

1.1 SYSTEM ORGANIZATION

The P856M/P857M systems consist of a basic central processor unit (CPU) and various independent elements interconnected via a General Purpose Bus. The CPU is contained on a single printed-circuit card. The same card includes a Serial Control unit for connecting an operator's device. Some of the other system elements are: memory modules, control units, input/output processor (IOP), the P857 options floating-point processor (FPP) and memory management unit (MMU).

1.2 A block diagram of the P856M/P857M systems is shown in Figure 1-1. All system elements are interconnected via the GP Bus. The P852M system elements are plug compatible with the P856M/P857M systems and may be connected via the same GP Bus. P850/P855 control units may also be used with the 856/857 system; the 850/855 cards are mounted in their own chassis and connected to the GP Bus via a bus converter unit. Chassis information and wiring is included in Section IV.

1.3 The P857M and P856M systems use the same basic logic design and both systems are based on a single-card CPU and operator's control unit. The two systems use different microprogram control, and the P857M performs an expanded number of functions, including operations with the FPP and MMU options. The specific logic differences are noted throughout the Logic description, Section II. Also, the P857M and P856M systems have different standard control panels as shown in Section III.

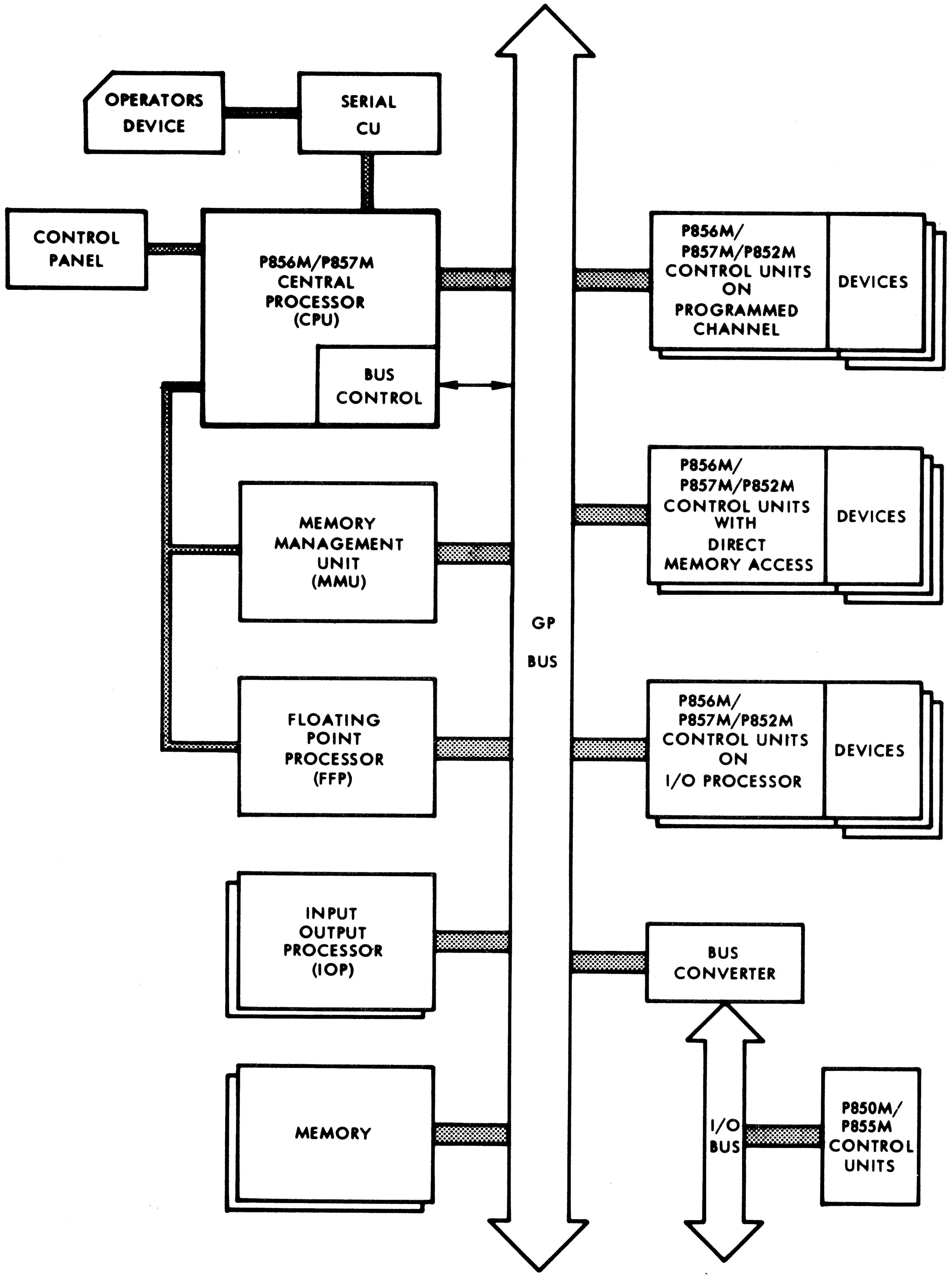


Figure 1-1 P856M/P857M System Block Diagram

1.4 Memory

The P856M maximum memory size is 32k words. The P857M maximum memory size is 32k words with the basic system configuration, 64k words with the MMU option, or 128k words with the MMU and the M5 chassis options. Memories are provided in modular form with up to 16k words per card.

1.5 Memory addressing on the GP Bus lines is by character address, and usually shown in hexadecimal. CPU logic which deals with memory word addresses simply places the word code on the GP Bus lines shifted one bit to the left. This then accesses an even-numbered memory character address. Some of the first (low numbered) memory locations are reserved for hardware-addressed functions, as follows:

Address		Function
Decimal Word	Hexadecimal Character	
0 62	000 07C	Interrupt list words
63	07E	Trap routine list word
64 127	080 0FE	
128	100	

OVERFLOW OF STACK

1.6 Memory Management Unit (MMU)

The MMU is a P857 single-card hardware option which uses Virtual Addressing. This option extends the main memory from 32k to 128k words, while still using the 16-bit addressing. The Virtual Addressing system also allows software extension of main store to backing store, via the Direct Memory Access channel. The MMU card uses a dedicated slot adjacent to the CPU card. The MMU is described in Appendix C.

1.7 Floating Point Processor (FPP)

The FPP is a P857 single-card hardware option which performs floating-point arithmetic operations. The FPP uses a dedicated slot (beside the MMU position)

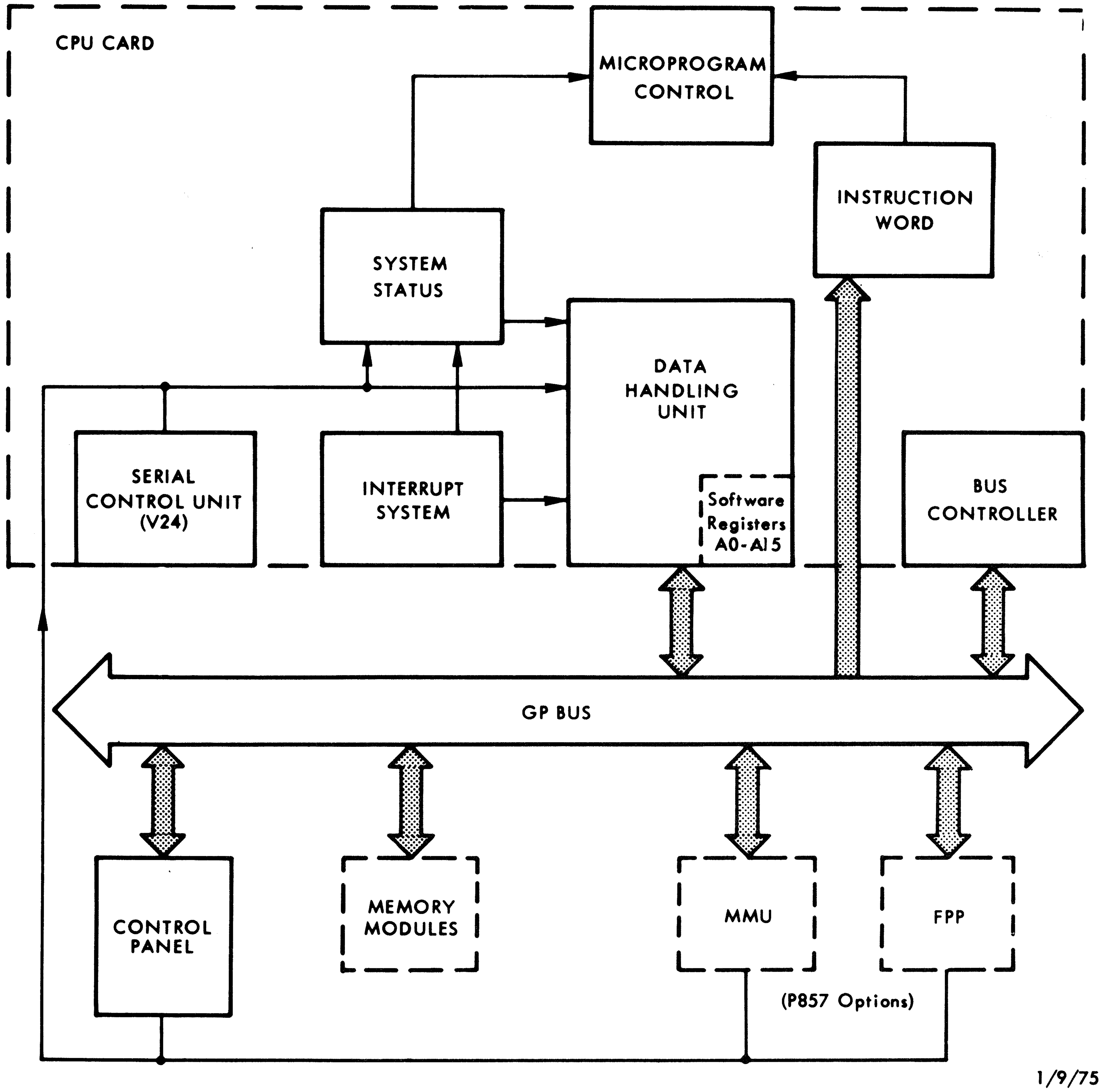


Figure 1-2 P856M/P857M CPU Block Diagram

in the cabinet. Dedicated wiring between the FPP and CPU is used to increase operating speed for the floating point calculations. The FPP is considered an extension of the CPU arithmetic section rather than an independent unit. The FPP is described in Appendix D.

1.8 I/O Processor (IOP)

The IOP is a hardware I/O channel that manages direct data transfers between control units and memory. The IOP multiplexes a number of control units for memory data transfer (eight CUs with the IOP type A). The IOP card can be inserted in any slot of the basic cabinet. The IOP is described in Appendix B. Additional information about I/O channels is given in paragraph 1.28.

1.9 GP BUS

The General Purpose Bus is a 57-line communicating link between all system elements, such as the CPU memory modules, I/O processors, and control units (Figure 1-1). System elements use the GP Bus on a master-slave basis. The CPU operates only as a master; the memory, external registers, and most device control units are slaves; the I/O processor may operate as master or slave (for CU with integrated DMA channel, the DMA can operate as master or slave).

1.10 The Bus Controller logic in the CPU regulates access of masters to the GP Bus. Whenever the Bus is free, the Bus Controller scans the masters in a specific sequence for a Bus-access request. The CPU has direct access to the memory at the completion of each instruction.

1.11 CENTRAL PROCESSOR UNIT

The P856/P857 CPU card contains the complete central processor, the GP-Bus control logic, and a serial control unit. The main CPU logic units and data paths are shown on the block diagram, Figure 1-2. A more detailed block diagram and complete logic diagrams are provided in Section II, CPU Logic description.

1.12 The Data Handling Unit does the processing of all data words accessed by the CPU. This unit also handles the addressing for both data transfers and instruction-word transfers. The Microprogram Control is a read-only memory and associated logic which controls all CPU operations.

1.13 CPU SOFTWARE REGISTERS

A scratchpad comprising sixteen 16-bit registers (A0 to A15) is directly accessible to software. The scratchpad contains 15 working registers (A0 to A14) and a stack pointer (A15). The working registers are used as an operand for some instructions. The scratchpad is located in the Data Handling section of the CPU logic and is connected to the operand-A input of the arithmetic logic unit.

1.14 A 2-bit condition register (CR) is provided for testing operation results. This register is also located in the Data Handling section of the CPU logic.

1.15 BUS CONTROLLER

The Bus Controller scans the GP Bus priority chain for selecting a master, provides control for the memory, and gates input/output data between the GP Bus and the CPU. The Bus Controller is included on the CPU card and is interconnected with the CPU logic.

1.16 INTERRUPT SYSTEM

The Interrupt System is a hardware feature which allows a running program to be interrupted by a higher-priority program. The Interrupt System is used for CPU logic functions and for control-unit I/O channel operations. There are 63 interrupt levels, divided into two groups: internal and external. There is a separate interrupt-request associated with each of the 63 levels.

1.17 Internal Interrupts

The internal interrupts (Figure 1-3A) use the eight highest-priority levels, 0-7. Four of the internal interrupts are pre-wired from CPU logic functions; the other four may be connected to other units in the basic chassis, but are not

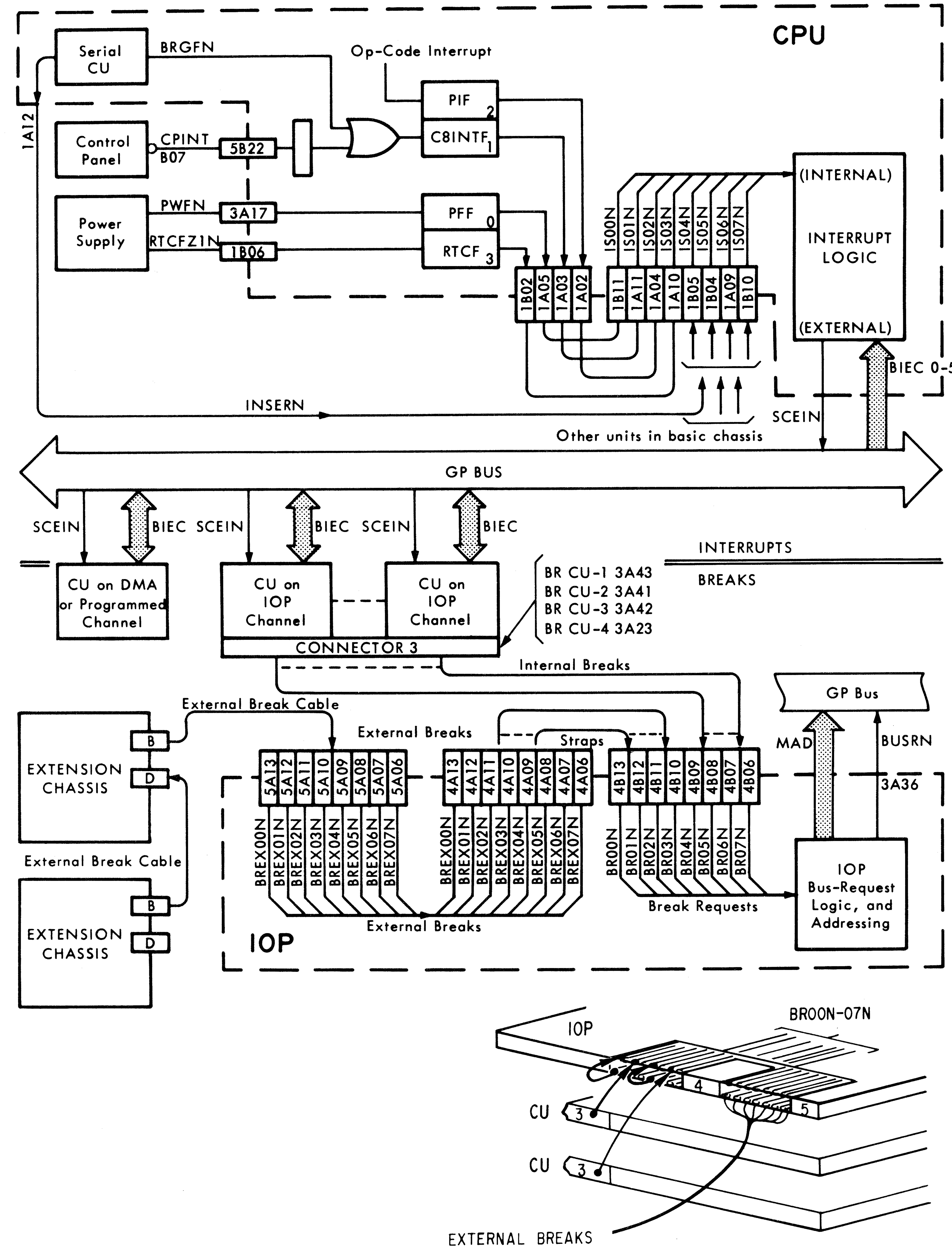
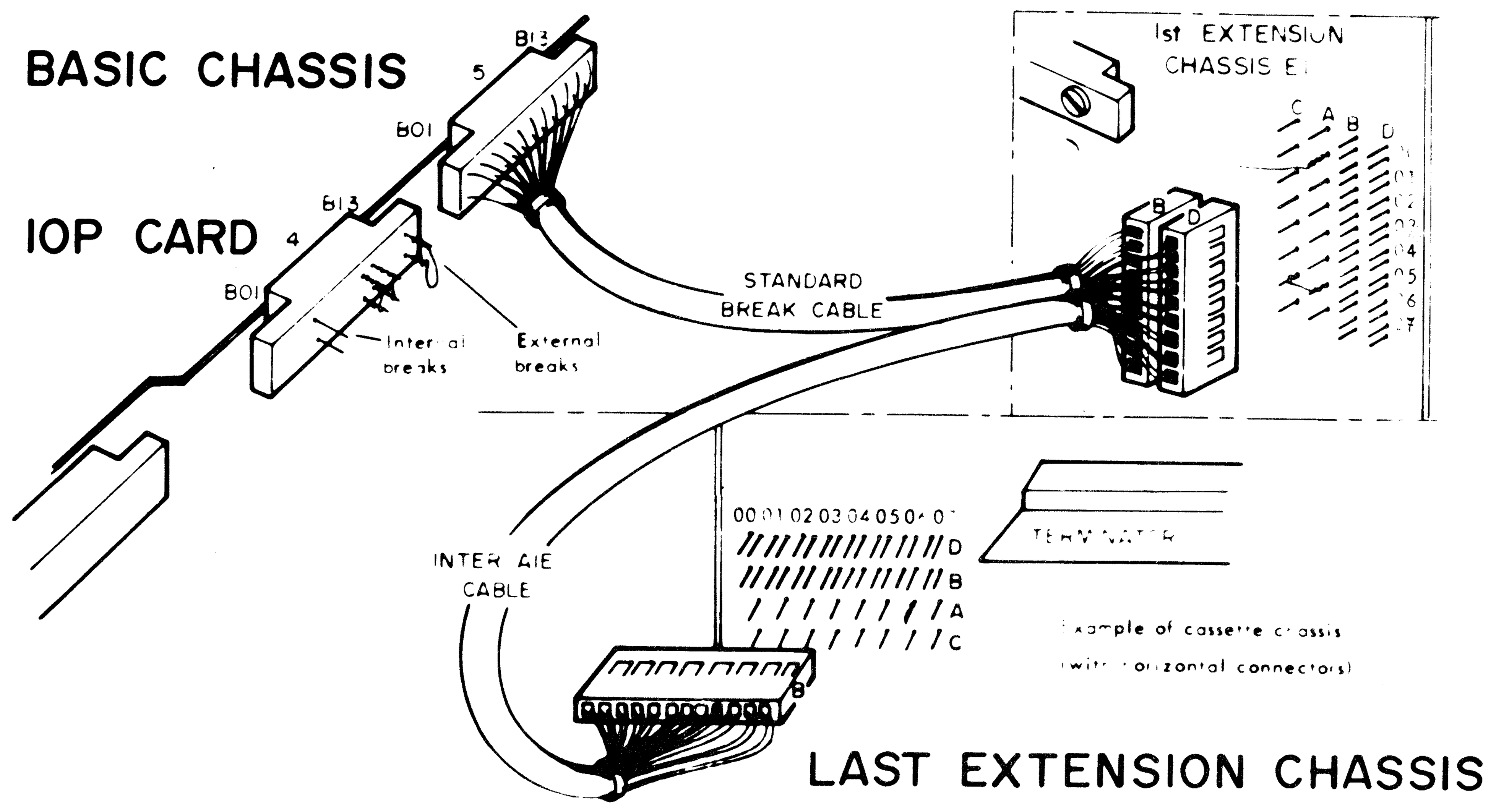
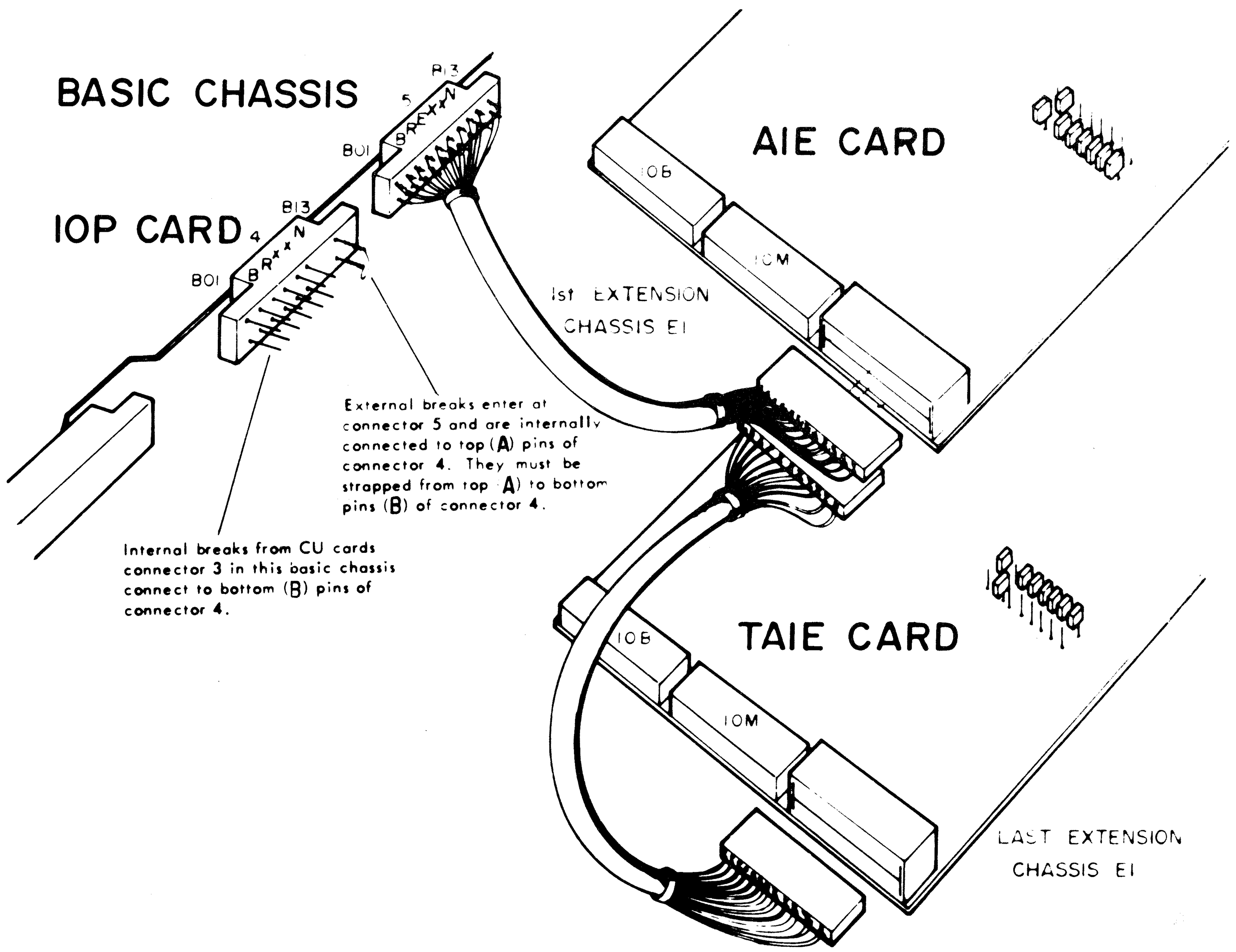


Figure 1-3A Interrupts and Breaks

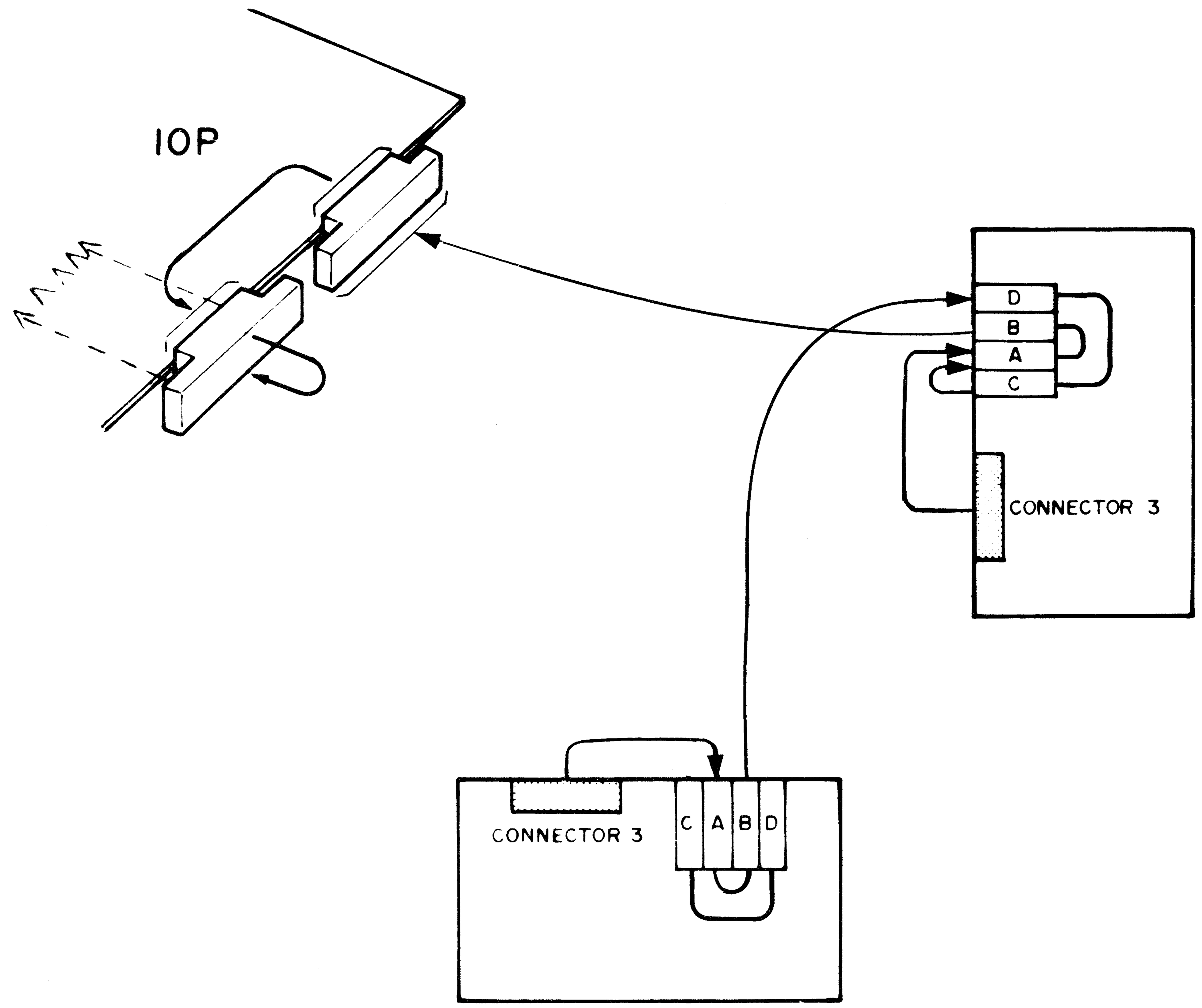
EXTENSION CHASSIS E2 (P852/856/857)



EXTENSION CHASSIS E1 (P850/855)



MULTIPLE-CHASSIS DETAILS (P852/856/857)



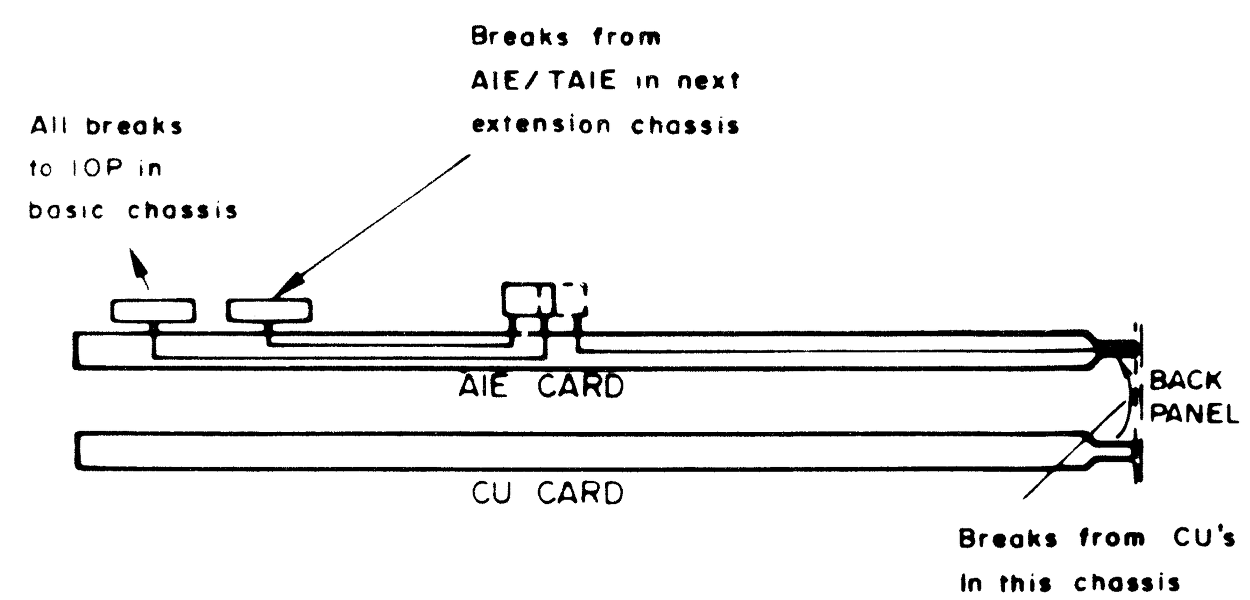


Figure 1-3B Break Cable Details

used for CU I/O channel operations. If any of the levels four through seven are not used for internal interrupts, they may be used by the external interrupts on the BIEC lines. The internal interrupts are assigned as follows:

level 0 - Power failure, Automatic restart.
1 - Operator's interrupt (Control Panel or I/O Console).
2 - Operation Code interrupt (Link to Monitor and Stack Overflow).
3 - Real Time Clock interrupt.
4-7 - Available for other internal interrupts within the basic chassis (serial CU, FPP interrupts, etc.) or external interrupts on the BIEC lines.

The four CPU logic interrupts are set into CPU flip-flops as they occur. They are reset individually by the RIT instruction in the corresponding interrupt program.

1.18 External Interrupts

The external interrupts (Figure 1 -3) are assigned priorities 4 to 62, although 4 to 7 may be used by internal interrupts. All I/O control units (except the CPU-integral serial controller) use the external interrupts, including those control units mounted in the basic chassis. Control units on the programmed channel use an external interrupt to transfer each word. Control units on all three channels use the external interrupt to request a status transfer at the end of a data-block transfer. For control units on the IOP channel, word transfers are initiated by break requests (BR) to the IOP; the IOP then makes a Bus Request to obtain control of the GP Bus for the word transfer. A break request is part of the IOP channel (paragraph 1.33) and not part of the interrupt system.

1.19 External interrupt requests are connected to the CPU interrupt logic via a 6-bit code on the GP-Bus BIEC lines. The priority level of a control unit is established by a priority encoder (with a set of jumpers) on the CU itself. Any pending interrupts on the BIEC lines are sampled at the end of the instructions by the scan-interrupt signal SCEIN. The CU priority encoders sample the BIEC lines and only the highest priority external interrupt request is coded onto the lines.

1.20 Interrupt Control

An Enable Interrupt (ENB) instruction is used to enable the CPU interrupt system. The entire interrupt system can be blocked with the Inhibit Interrupt (INH) instruction. The hardware flip-flops generating the internal interrupts are reset individually by the RIT instruction. The hardware that is generating the external interrupts is reset by appropriate CIO instructions. At power-on time, and at every master clear from the control panel:

- the CPU is set to Enable Interrupt mode,
- the current program level is established at 63, and
- all internal and external interrupt requests are reset.

1.21 Interrupt Sequence

The internal/high-priority interrupts (levels 0-7) are sampled at the end of each instruction execution (except Move Table which is sampled early). If there is no internal interrupt, and there has been no external sampling within 2 microseconds, the highest-priority external interrupt (which is coded on the BIEC lines) is sampled. The highest-priority interrupt request is then compared with the priority level of the running program. If the running program is of higher or equal priority to the interrupt request, the program continues. If the interrupt request is of higher priority than the running program, the interrupt sequence is started:

- The current instruction (except Move Table) is completed. For Move Table, registers are updated to allow resuming the instruction at the point it was suspended.
- The program counter (P) is stored in the memory-stack location specified by stack pointer A15. P contains the address of the next instruction (except for Move Table, where P points to the instruction itself). A15 is decremented by 2.
- The program status word is stored in the memory location adjacent to (P), specified by stack pointer A15. A15 is decremented by 2.
- The Inhibit Interrupt state is set.
- The system User Mode flag is reset (unless already reset).
- The priority level register (PLR) is loaded with the new level number.

- An indirect branch is made to the corresponding memory location (paragraph 1.23).
- The interrupt routine is executed.

Note: A Return instruction with a pointer other than A15 can be used independently of the interrupt routine to switch from any program to another under a supervisory program control.

1.22 Interrupt Routine

The following operations must be performed by the interrupt-routine program:

- Some or all of the accumulators (A0 - 15) are saved in the interrupt memory stack.
- The interrupt itself is treated, including a RIT, SST, or other instruction to reset the interrupt signal.
- The accumulators are re-loaded from the stack at the end of the interrupt routine.
- A Return instruction, referring to stack pointer A15, is programmed. During this instruction, the contents of A15 are used (with incrementing) to restore the program counter into P and retrieve the program status word.

1.23 Interrupt Addresses

The first 63 word locations in memory are used for the interrupt-routine list words. The CPU interrupt logic generates a direct six-bit word address for the accepted interrupt. This address code is shifted left one position onto the address lines to produce the memory character address, as follows:

Interrupt Level	Address Code from Interrupt Logic	Bit 0 added, for mem. char. add.	
0	0 0 0 0 0 0	0	0
1	0 0 0 0 0 1	0	2
2	0 0 0 0 1 0	0	4
3	0 0 0 0 1 1	0	6
4	0 0 0 1 0 0	0	8
5	0 0 0 1 0 1	0	A
6	0 0 0 1 1 0	0	C
	etc. to		
62	1 1 1 1 1 0	0	7C

1.24 Memory Stack Operation

The interrupt system utilizes a memory stack with automatic handling. The hardware uses this system stack during the interrupt sequence to save the program status word and the instruction counter of the interrupted program. The software uses the stack: to save and later restore any other parameters of the interrupted program; to link a program to a subroutine; and to return to the main program. The system stack is also used by software for Traps and page faults.

1.25 The stack operates on a last-in first-out basis, controlled by the automatic updating of stack pointer A15. Load, Store, Multiple Load, and Multiple Store can be used as stack-handling instructions when their effective address refers to A15. The Call Function (CF) instruction is a branch with automatic saving of PSW and P into the stack. The Return (RTN) instruction is used at the end of an interrupt routine or a subroutine to restore PSW and P.

1.26 Stack Overflow is signalled by an Operation Code interrupt (level 2, internal interrupt). This signal is generated by hardware when the stack pointer decrements to less than 128_{10} (word address) to indicate that the stack is almost full and to prevent overwriting in the dedicated low address memory locations.

1.27 Additional memory stacks may be used by software, using the scratchpad accumulators A0 to A14 for stack pointers. These software stacks will not have the automatic handling and updating like the system stack which uses A15 as the pointer. All references here to the stack pertain to the automatic-handling stack. All memory stacks may have software limits to stack size established at system generation time.

1.28 INPUT/OUTPUT CHANNELS

The P852M/P856M/P857M systems have three different input/output channels:

- programmed channel,
- multiplex-type I/O-processor channel, and
- direct memory access channel.

All I/O data transfers are via the GP Bus and are timed by the Bus Controller logic on the CPU card. For all three types of I/O transfers, the program initiates control-unit operation with a CIO Start command. The CU must reply with an interrupt request or a break request (depending on the type of I/O channel) when it is ready for the first word (or character) transfer. The data are then transferred according to the channel type. When a data-block transfer is ended (either complete or early), the CU signals to the CPU with an interrupt request. The program must then issue a Send Status (SST) command to obtain the status word from the CU.

1.29 For transferring a block of data via any I/O channel, the program provides the memory address of the first word and the length of the block to be transferred. During the transfer, these two control words must be updated: the memory address is incremented to select sequential locations in the data block; the block length is decremented to determine when the complete block has been transferred (length = zero). The method of handling this pair of control words depends on the type of I/O channel.

1.30 Programmed Channel

This is an input/output exchange between a CU and memory via the CPU, under complete program control (Figure 1-4). The exchange is word-by-word or character-by character at up to 40,000 characters per second. On the programmed channel, the address/length control-word set is located in program registers. For each data word or character transferred, the program must access both of these registers to up-date the control words.

1.31 For an output transfer, a Load Register instruction loads the first word from memory into the CPU scratchpad register. The CU signals that it is ready with an Interrupt Request to the CPU. An OTR instruction then transfers the word from the CPU to the CU, while another Load instruction obtains the next word from memory. This procedure continues until the last word of the block is transferred. When the program loads the last data word, its block length is counted to zero. The CPU then sends CIO Halt to the CU along with the last OTR data transfer, and the transfer is ended with the status transfer.

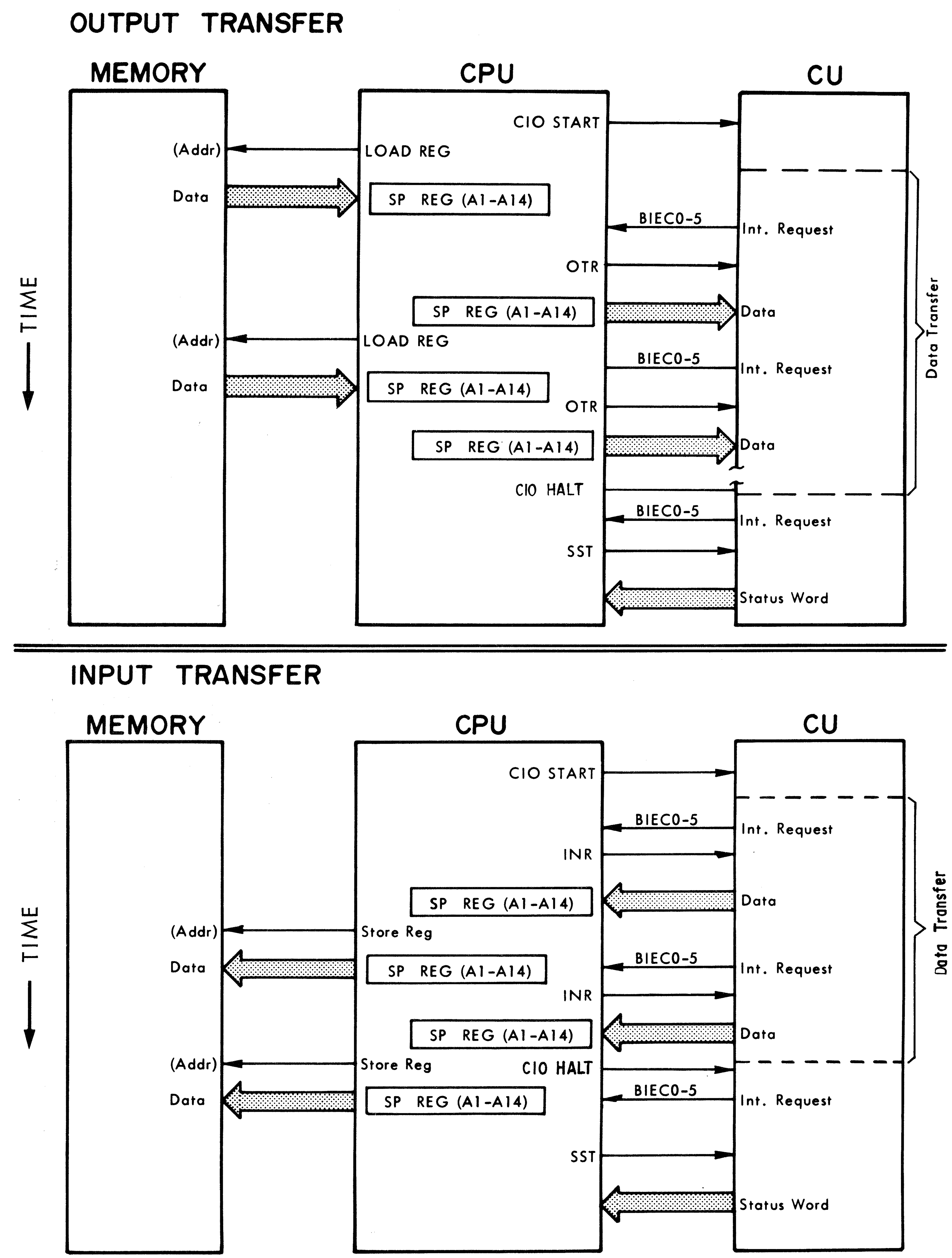


Figure 1-4 Programmed Channel Transfers

1.32 The input transfer sequence is essentially the same as the output transfer. The input sequence, however, begins with an INR command which transfers data from the CU into the CPU scratchpad register. A Store Register instruction then transfers the word from the scratchpad register into the memory.

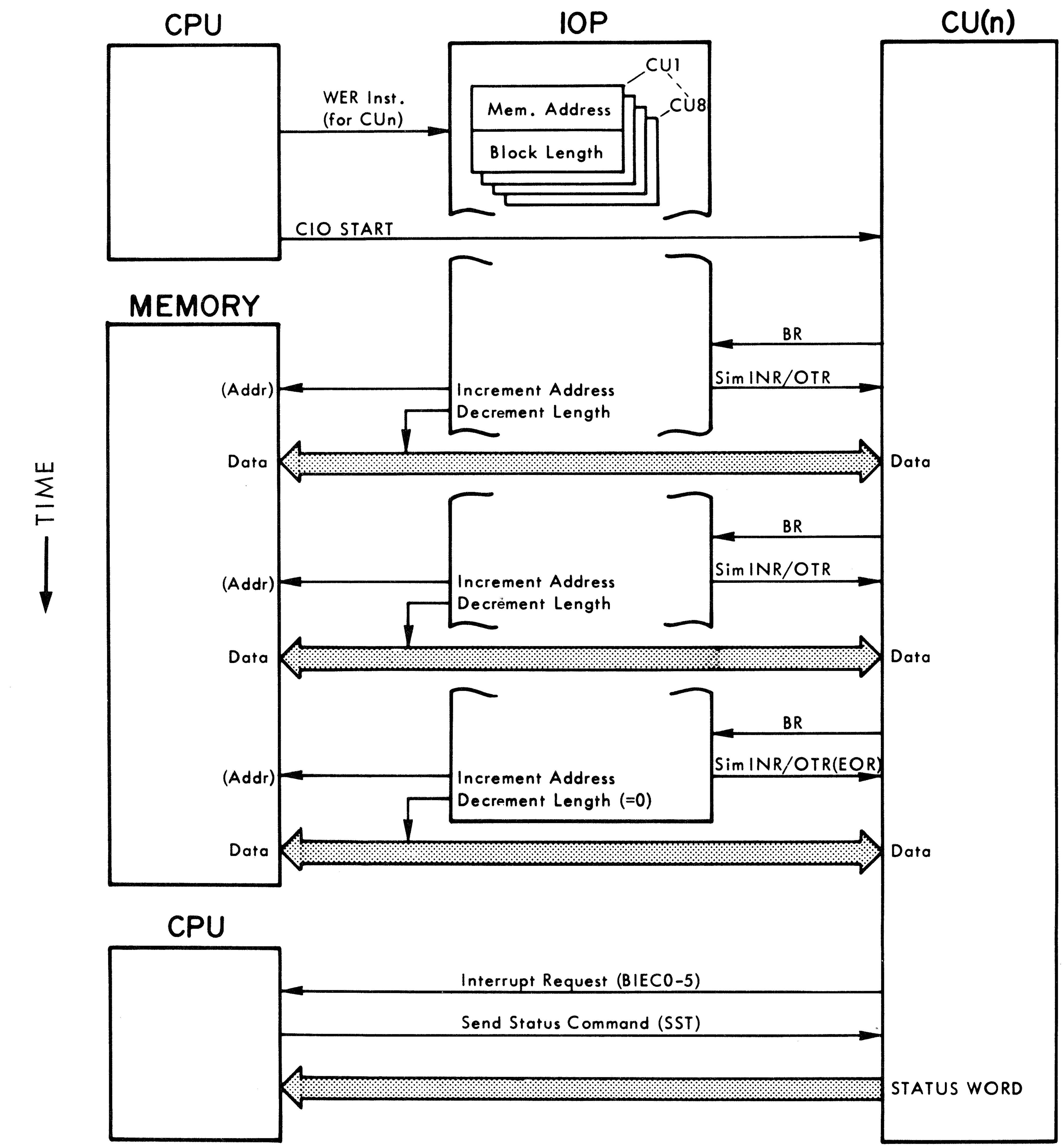
1.33 I/O Processor Channel

The Input/Output Processor (IOP) channel manages data transfers directly between memory and a number of multiplexed control units (Figure 1-5). CPU registers are not used, and there is no need for program control except for starting the exchange and testing status at the completion. The exchange is in blocks at up to one million words per second.

1.34 The IOP is a hardware option that contains a pair of address/length control-word registers for each of its CU channels. At the beginning of a transfer to one CU, the program uses two WER instructions to load this register pair with the starting address and the block length. The IOP logic then provides all GP Bus timing signals to control the data transfers directly between the memory and the CU.

1.35 For input or output transfer, the CU signals that it is ready with a Break Request (BR) to the IOP. The IOP makes a Bus Request to obtain control of the GP Bus (Figure 1-3). The IOP then sends a simulated INR or OTR command to the CU to initiate one word transfer. The INR/OTR command is simulated in that it is generated by the IOP and is not a programmed instruction. The IOP logic updates its control-word registers for each data word. When the block length is counted to zero, the IOP sends End Of Record (EOR) to the CU along with the last simulated INR/OTR data transfer. The data block transfer is ended with an SST command and status transfer between the CU and CPU.

1.36 The IOP can be loaded with the data-exchange control information for a number of control units (up to eight for the type-A IOP). The IOP then multiplexes the exchanges between the CUs and the memory.



Note -- This diagram shows the sequence for transferring a block of data between memory and a single CU, (n). The IOP can multiplex up to eight CUs; the operation shown is thus duplicated for each CU.

Figure 1-5 Multiplexed I/O Processor Channel Transfers

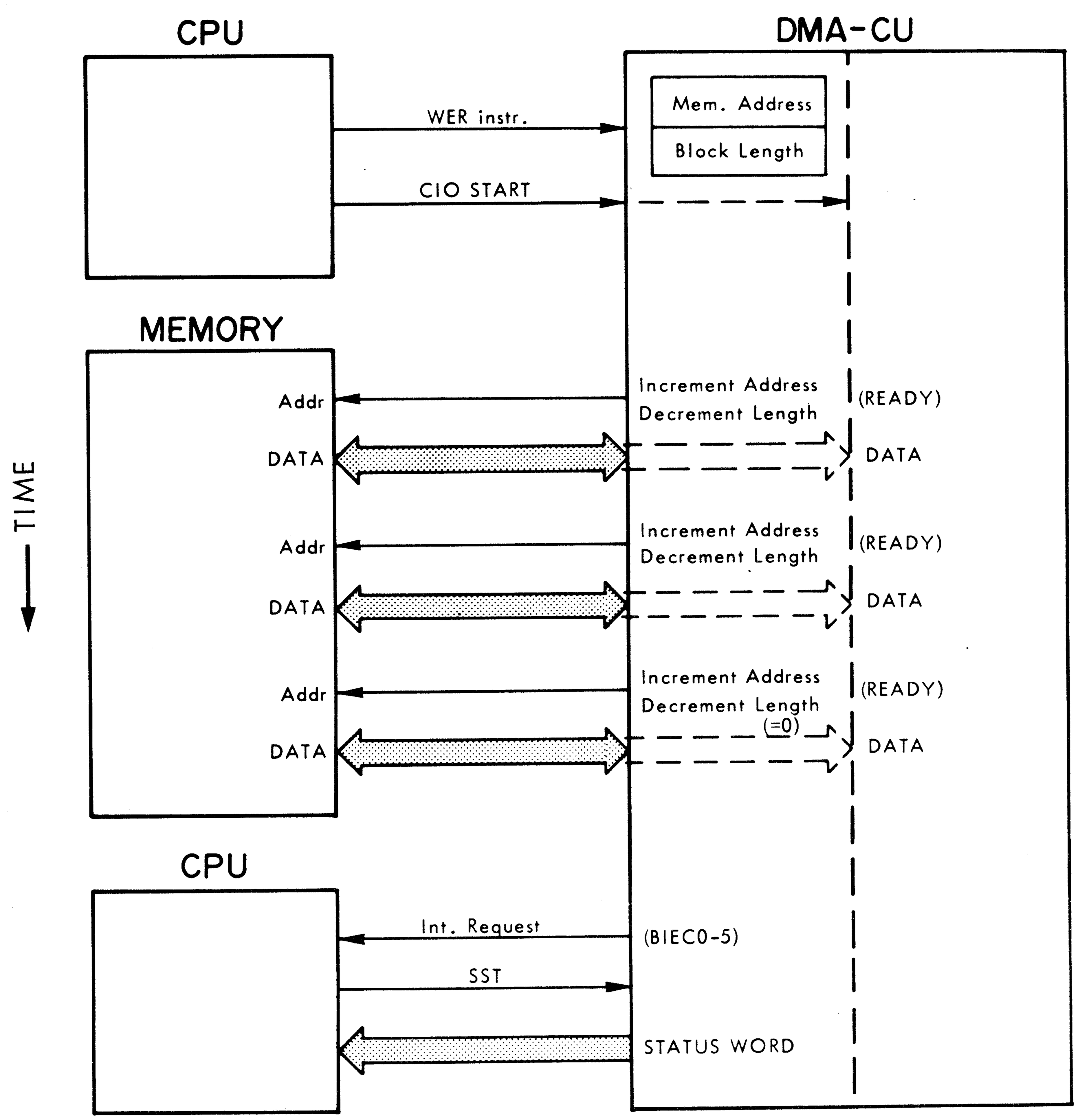


Figure 1-6 Direct Memory Access Channel Transfer

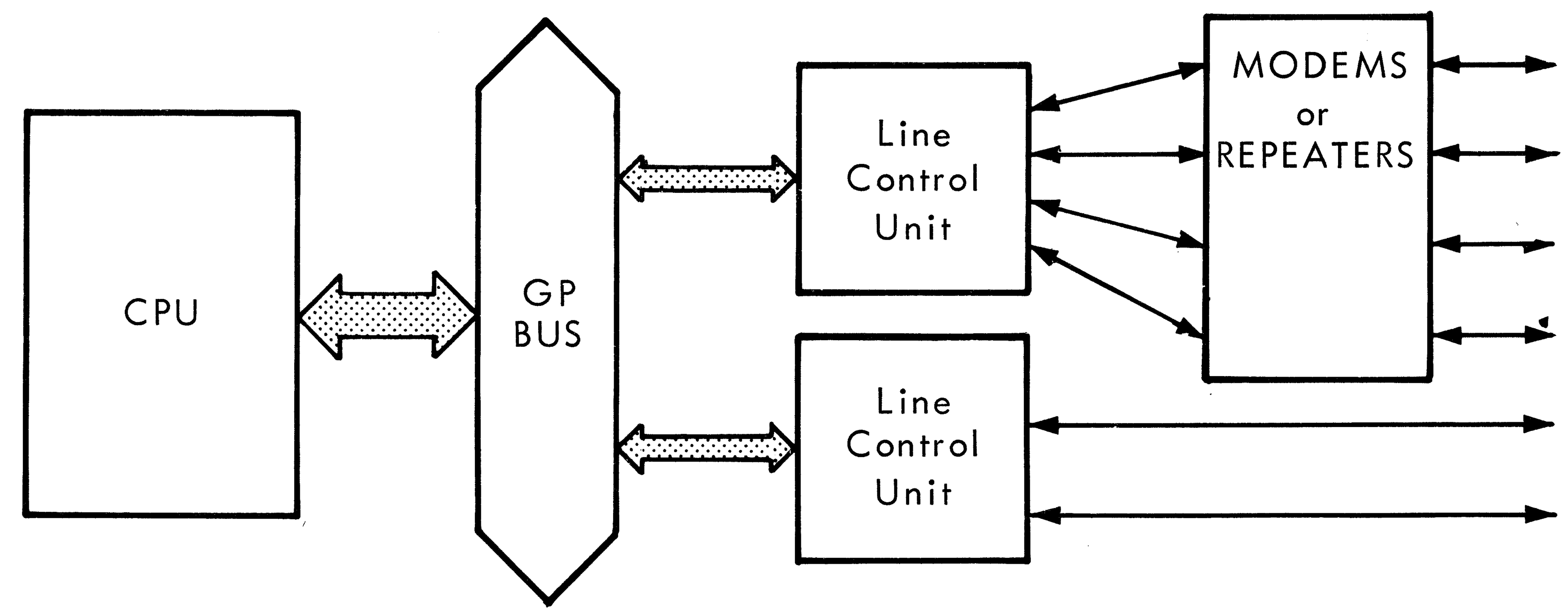
1.37 Direct Memory Access Channel

The DMA channel manages data transfers directly between memory and a single high-speed control unit (Figure 1-6). CPU registers are not used, and there is no need for program control except for starting the exchange and testing status at the completion. The DMA channel is a hardware channel included as a part of the high-speed control unit (DMA-CU card).

1.38 At the beginning of a transfer, the program uses two WER instructions to load the starting address and block length into the DMA control-word register. The DMA logic then provides all GP Bus timing signals to control the data transfers directly between the memory and the DMA CU. The DMA logic also updates the control-word register for each data word and detects when the complete block has been transferred. The data block transfer is ended with an SST command and status transfer between the CU and CPU.

1.39 Data Communications Channels

Data communication with the P856M/P857M system is accomplished with various Line Control Units operating via the programmed channel or the multiplexed IOP channel. The Line Control Units connect to the GP Bus in the same manner as any other control unit cards. The Line Control units are connected to the communications lines, either directly or via modem interface or repeater units (following diagram).



Some Line Control Units used with the system are:

- SLCU Synchronous line control units
- ALCU Asynchronous line control units
- ALM Asynchronous low speed multiplexers

1.40 CLOCKS

1.41 CPU Timing Clock

The CPU logic timing is controlled by the Sequensor which provides the clock signals AP, BP, and T1 through T10. The Sequensor is described in Section II.

1.42 Real Time Clock

A Real Time Clock produces an interrupt signal every 20ms. The real-time clock pulse is produced directly from the power supply and is described in Section V.

1.43 CONTROL PANEL

1.44 The standard control panel for the P856M system is the same as for the P852M. This panel is directly interchangeable between the two systems. The Extended control panel used by the P857M is available as an option for the P856M system. Description and operation of the control panel is provided in Section III.

1.45 The standard control panel for the P857M system is the Extended Control Panel. This panel has, in addition to the standard Data and Control facilities, complete Addressing facilities, including the ability to stop on preset addresses. The Data/Control half of the panel is the same as the P852M/P856M control panel, but with the addition of a TEST position on the key switch for performing automatic microdiagnostic tests. Description and operation is provided in Section III.

1.46 TESTING

The P856/7 contains an automatic testing feature in the form of a microprogrammed diagnostic built in to the CPU logic. This automatic microdiagnostic operates through the control panel to test approximately 70% of the CPU logic. Successful running of the microdiagnostic indicates that sufficient parts of the CPU function for loading of test programs. The automatic microdiagnostic tests and the test programs are described in Section III.

1.47 STATUS

System status is contained in a sixteen-bit Program Status Word (PSW). The use and operation of the status word is described in Section II (Paragraph 2.84). The status word contents are shown in the following diagram.

Program Status Word

PLR						CR		GF							
0	1	2	3	4	5	0	1	RUN	ENB	CPF	PFF	RTCF	PIF		FU
										Interrupts					1 = User Mode 0 = System Mode

1.48 DATA FORMAT

The data are handled in 16-bit words with 15 magnitude bits and bit 0 as a sign bit. Bit 1 is the most-significant data bit. The data word is handled in the CPU and on the GP Bus with negative logic: a logic 1 is a low level (0V) and a logic 0 is a high level (+5V).

Data Word

Bit	0	1	15
	Sign Bit 0 = positive 1 = negative	Magitude Bits	Least-significant bit

1.49 Double Precision

Double precision is obtained by utilising two successive words to obtain 30 magnitude bits. The sign bit and the 15 most-significant bits are in the first word; the 15 least-significant bits are in the second word. Bit 0 of the second word is not used, and is always zero.

1.50 Double precision is used for the product of the multiplication of two single-precision words, and for some other operations.

1st Word

Bit:	0	1	15
	S	1	
	Sign Bit	Most-significant Bit	Most-significant Magnitude Bits

2nd Word

Bit:	0	1	15
	0	1	
	not used	Least-significant Magnitude Bits	Least-significant Bit

1.51 Logical Data

Logical data, such as the condition of sixteen binary indicators, can be stored in a single data word. This type of data is generally not treated arithmetically by the program but logically by means of boolean operations such as AND, OR, and Exclusive-OR. In this case, bit 0 of a word is not used as a sign bit.

1.52 Floating-Point Data

Real, floating-point numbers are contained in three successive words. The mantissa is stored in the first two words as a double-precision integer. The third word contains the exponent, represented as a single-precision integer.

1st Word

Bit:	0	1 - 15
	S	Mantissa
	Sign	Most-significant Bits (1-15)

2nd Word

Bit:	0	1 - 15
	0	Mantissa
	Not Used	Least-significant Bits (16-30)

3rd Word

Bit:	0	1 - 15
	S	Exponent
	Sign	

1.53 Character Handling

Character handling is performed by some instructions. The right character of a word is the least significant (bits 8-15) and the left character is the most significant (bits 0-7).

1.54 OPERATING MODES

The CPU can operate in two basic modes: System Mode or User Mode. The mode is specified by bit 15 of the program status word (0 = system; 1 = user).

1.55 System Mode

The System Mode is reserved for the monitor program and system programs. In the System Mode, execution of the complete instruction set is allowed. This mode assures the system resource allocation, protected by the following privileged instructions:

- Control Instructions which modify the CPU state:
  HLT -- Halt
  INH -- Inhibit Interrupt
  RIT -- Reset Internal Interrupt
- All I/O Instructions:
  CIO, OTR, INR, SST, TST
- External Register Instructions:
  WER, RER
- (P857M) MMU-related instructions, Extended Load or Store, and Segment Table Load or Store:

ELR, EL, ESR, ES,

TLR, TL, TSR, TS

- Some other instructions are reserved for System Mode when they modify the contents of the stack pointer (A15). Refer to Table 1-1.

1.56 User Mode

This mode is reserved for the user program execution. If a program in User Mode attempts to execute a privileged instruction, a Trap routine is performed: the program parameters (P and PSW) are saved in the stack, and the program branches to the address contained in memory location 7E.

1.57 When a program running in User Mode needs system allocation, a Link to Monitor (LKM) instruction executes a call to the monitor which sets the CPU to System Mode.

1.58 When a program running in System Mode is complete, and ready to allow user programs to run, either a Set Mode (SMD) instruction or a Return (RTN) instruction with R2=15 can be used. The User-Mode indicator in the PSW is then set to 1.

1.59 INSTRUCTIONS

The P856/857 instruction set is divided into ten groups:

- load and store
- arithmetic
- logical
- character handling
- branch
- shift
- table handling
- control
- input/output
- external transfers

Table 1-1 lists the instructions and indicates the operating flow diagram for each instruction. The instruction formats are described in paragraph 1.62. Instructions are specified by addressing mode as well as the op-code. The addressing mode is described in paragraph 1.65 and listed in Table 1-2.

1.60 Invalid Instructions

An invalid instruction code initiates a Trap microprogram (paragraph 1-55, Figure 1-12). The program must determine what action (such as interrupt) to take following a Trap.

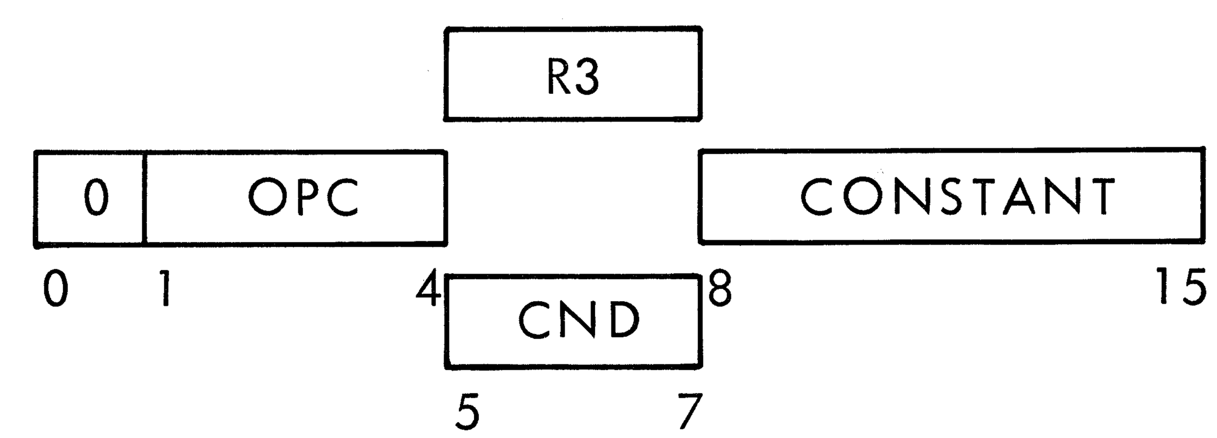
1.61 Some P857 instructions require the Floating Point Processor (FPP) or the Memory Management Unit (MMU). Both of these system extensions are optional. If the FPP is not present, the corresponding instructions initiate the Trap. If the MMU is not present, the corresponding instructions give no significant result when they are executed.

1.62 Instruction Format

There are two instruction formats, indicated by bit 0 of the instruction word:

- <u>Format 0</u> Constant handling instructions.
- <u>Format 1</u> Memory reference or register-to-register operations.

1.63 <u>Format 0 Instructions (Type T8)</u>



OPC : Operation code

R3 : Register (scratchpad A0-A7) on which the operation is performed.

CND : Condition for relative branching (when specified by OPC).

Table 1-1 P856/857 Instruction List

Flow Diagram 1:	Mnemonic	OPC	L/S (Bit 15)	Address Type	Instruction Name
					Load and Store Instructions
15	LDK	0	0	T8,2	Load constant
	LDR	0	0	T1,3	Load reg/reg; update stack pointer
	LD	0	0	T4-7	Load register
	STR	0	1	T3	Store reg/reg; update stack pointer
	ST	0	1	T4-7	Store register
23	MLK	7	0	T2	Multiple load constant
	MLR	7	0	T3	Multiple load register
	ML	7	0	T4-7	Multiple load
	MSR	7	1	T3	Multiple store register
	MS	7	1	T4-7	Multiple store
29	ELR	10	0	T3	Extended load register (MMU)
	EL	10	0	T4-7	Extended load (MMU)
	ESR	10	1	T3	Extended store register (MMU)
	ES	10	1	T4-7	Extended store (MMU)
					Arithmetic Instructions
17	ADK	2	0	T8,2	Add constant
	ADR	2	0/1	T1,3	Add reg/reg
	AD	2	0/1	T4-7	Add
	IMR	2	1	T3	Increment memory/reg
	IM	2	1	T4-7	Increment memory
18	SUK	3	0	T8,2	Subtract constant
	SUR	3	0/1	T1,3	Subtract reg/reg
	SU	3	0/1	T4-7	Subtract
	NGR	3	1	T1	Negate register
	C2R	3	1	T3	Two's complement/reg
	C2	3	1	T4-7	Two's complement
25	MUK	8	0	T2	Multiply with constant
	MUR	8	0	T1,3	Multiply reg/reg
	MU	8	0	T4-7	Multiply
26	DVK	9	0	T2	Divide with constant
	DVR	9	0	T1,3	Divide reg/reg
	DV	9	0	T4-7	Divide
29	DAK	10	0	T2	Double add with constant
	DAR	10	0	T1,T3	Double add reg/reg
	DA	10	0	T4-7	Double add

Flow Diagram 1:	Mnemonic	OPC	L/S (Bit 15)	Address Type	Instruction Name
					Arithmetic Instructions
30	DSK	11	0	T2	Double subtract with constant
	DSR	11	0	T1,3	Double subtract reg/reg
	DS	11	0	T4-7	Double subtract
28	FFL	9	0	T1	Integer to floating point (FPP)
	FFX	9	1	T1	Floating point to integer (FPP)
27	FADR	9	0/1	T3	Floating-point addition/reg (FPP)
	FAD	9	0/1	T4-7	Floating-point addition (FPP)
	FSUR	9	0/1	T3	Floating-point subtract/reg (FPP)
	FSU	9	0/1	T4-7	Floating-point subtract (FPP)
	FMUR	9	0/1	T3	Floating-point multiply/reg (FPP)
	FMU	9	0/1	T4-7	Floating-point multiply (FPP)
	FDVR	9	0/1	T3	Floating-point divide/reg (FPP)
	FDV	9	0/1	T4-7	Floating-point divide (FPP)
					Logical Instructions
19	ANK	4	0	T8,2	Logical AND with constant (R3≠0, R1≠0)
	ANR	4	0/1	T1,3	Logical AND reg/reg (R1≠0)
	AN	4	0/1	T4-7	Logical AND (R1≠0)
20	ORK	5	0	T8,2	Logical OR with constant (R3≠0, R1≠0)
	ORR	5	0/1	T1,3	Logical OR reg/reg (R1≠0)
	OR	5	0/1	T4-7	Logical OR (R1≠0)
	XRK	6	0	T8,2	EXclusive OR with constant (R1≠0)
	XRR	6	0/1	T1,3	EXclusive OR reg/reg (R1≠0)
	XR	6	0/1	T4-7	EXclusive OR (R1≠0)
32	CWC	13	X	T8	Compare word to short constant
	CWK	13	0	T2	Compare word with constant
	CWR	13	0/1	T1,3	Compare word reg/reg
	CW	13	0	T4-7	Compare word
37	C1R	15	0/1	T1,3	One's complement reg/reg
	C1	15	0/1	T4-7	One's complement
19	TM	4	1	T1	Test mask
20	TNM	6	1	T1	Test not mask
19	CMR	4	1	T3	Clear memory/reg
	CM	4	1	T4-7	Clear memory

Flow Diagram 1:	Mnemonic	OPC	L/S (Bit 15)	Address Type	Instruction Name	Bits 8	9	10
Character Handling Instructions								
31	ECR	12	0	T1	Exchange character reg/reg (R1≠0)			
	LCK	12	0	T2	Load character with constant (R1≠0)			
	LCR	12	0	T3	Load character/reg (R1≠0)			
	LC	12	0	T4-7	Load character			
	SCR	12	1	T3	Store character/reg (R1≠0)			
	SC	12	1	T4-7	Store character			
32	CCK	13	1	T2	Compare character/constant (R1≠0)			
	CCR	13	1	T3	Compare character/reg (R1≠0)			
	CC	13	1	T4-7	Compare character (R1≠0)			
Branch Instructions								
16	AB	1	0	T8,2	Absolute conditional branch			
	ABR	1	0/1	T1,3	Absolute conditional branch to reg			
	ABI	1	0	T4-7	Absolute branch			
29	RF	10	0	T8	Relative forward conditional branch			
30	RB	11	0	T8	Relative backward conditional branch			
35	CF	14	1	T2	Call function (direct) (R1≠0)			
	CFR	14	1	T1,3	Call function/reg (R1≠0)			
	CFI	14	1	T4-7	Call function (via memory) (R1≠0)			
33	RTN	14	0	T3	Return (R1=0)			
34	EXK	14	1	T2	Execute constant (R1=0)			
	EXR	14	1	T1,3	Execute/register (R1=0)			
	EX	14	1	T4-7	Execute (R1=0)			
Shift Instructions						8	9	10
21	SLA	7	X	T8	Single left arithmetic (R3≠0)	0	0	0
	SRA	7	X	T8	Single right arithmetic (R3≠0)	0	0	1
	SLL	7	X	T8	Single left logical (R3≠0)	0	1	0
	SRL	7	X	T8	Single right logical (R3≠0)	0	1	1
	SLC	7	X	T8	Single left circular (R3≠0)	1	1	0
	SRC	7	X	T8	Single right circular (R3≠0)	1	1	1
	SLN	7	0	T8	Single left, normalize (R3≠0)	1	0	0
	SRN	7	0	T8	Single right, normalize (R3≠0)	1	0	1
22	DLA	7	X	T8	Double left arithmetic (R3=0)	0	0	0
	DRA	7	X	T8	Double right arithmetic (R3=0)	0	0	1
	DLL	7	X	T8	Double left logical (R3=0)	0	1	0
	DRL	7	X	T8	Double right logical (R3=0)	0	1	1
	DLC	7	X	T8	Double left circular (R3=0)	1	1	0
	DRC	7	X	T8	Double right circular (R3=0)	1	1	1

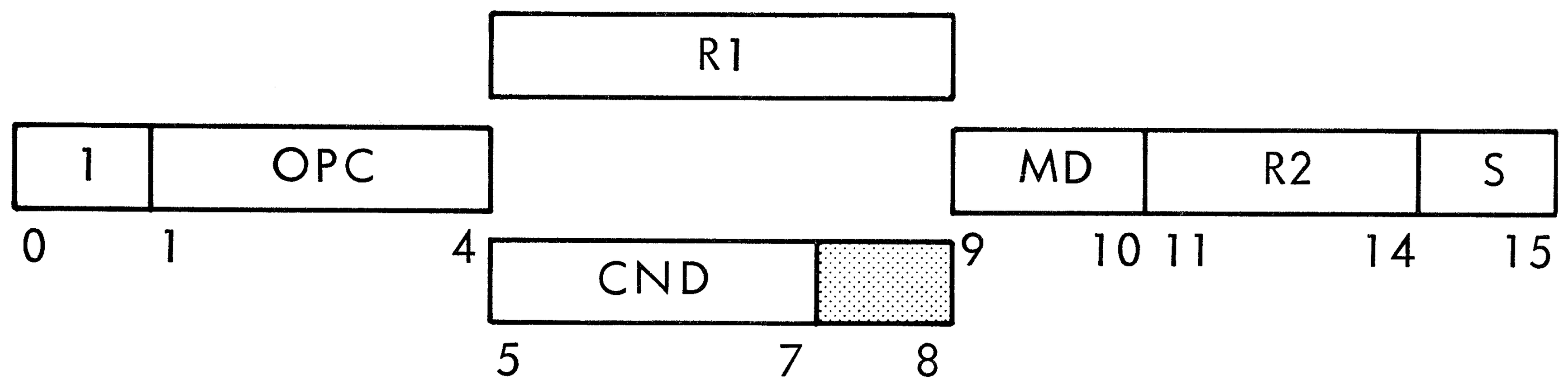
Flow Diagram 1:	Mnemonic	OPC	L/S (Bit 15)	Address Type	Instruction Name	Bits 8	9	10	11	12	13	14	15
Shift Instructions						8	9	10					
22	DLN	7	0	T8	Double left, normalize	1	0	0					
	DRN	7	0	T8	Double right, normalize	1	0	1					
Table Handling Instructions													
36	MVF	14	0	T8	Move table forward	0	0	0					
37	MVB	15	0	T8	Move table backward	0	0	0					
36	MVSU	14	0	T8	Move table, system to user (MMU)	1	0	0					
37	MVUS	15	0	T8	Move table, user to system (MMU)	1	0	0					
Control Instructions						8	9	10	11	12	13	14	15
19	HLT	4		T8	Halt	0	1	1	1	1	1	1	1
	INH	4		T8	Inhibit interrupt	1	0	1	1	1	1	1	1
	RIT	4	1	T8	Reset internal interrupt	1	1	DA					1
20	ENB	5		T8	Enable interrupt	0	1	0	0	0	0	0	0
	LKM	5		T8	Link to monitor	0	0	0	0	0	1	0	0
	SMD	5		T8	Set mode	0	0	0	0	0	0	0	1
Input/Output Instructions (R3≠0)						8	9						
24	CIO	8	X	T8	Control I/O (DA≠0)	1	n						
	OTR	8	X	T8	Output from register (DA≠0)	0	n						
	INR	9	X	T8	Input to register	0	n						
	SST	9	X	T8	Sense status	1	1						
	TST	9	X	T8	Test status	1	0						
External Transfer Instructions													
33	WER	14	X	T8	Write external register (R3≠0)								
37	RER	15	X	T8	Read external register (R3≠0)								
27	FLDR	8	0	T3	Floating-point load/reg (R1=2) (FPP)								
	FLD	8	0	T4-7	Floating-point load (R1=2) (FPP)								
	FSTR	8	1	T3	Floating-point store/reg (R1=2) (FPP)								
	FST	8	1	T4-7	Floating-point store (R1=2) (FPP)								
23	TLR	7	0	T3	Segment table load/reg (R1=0) (MMU)								
	TL	7	0	T4-7	Segment table load (R1=0) (MMU)								
	TSR	7	1	T3	Segment table store/reg (R1=0) (MMU)								
	TS	7	1	T4-7	Segment table store (R1=0) (MMU)								

| = P857 only

CND 5 6 7	Branch
0 n n	if CR = bits 6,7
1 n n	if CR ≠ bits 6,7
1 1 1	unconditional

CONSTANT: An 8-bit, positive constant, branch displacement, or OPC extension.

1.64 Format 1 Instructions (Types T1-T7)



OPC : Operation Code

R1 : Register (scratchpad A0-15) on which the operation is performed:
bit 8 = 0 : A0-7
bit 8 = 1 : A8-15

CND : Condition for absolute branching (when specified by OPC). (conditions same as for format 0)

MD : Addressing mode (Table 1-2).

R2 : Register (scratchpad A0-15) of 2nd operand or address of 2nd operand:
bit 14 = 0 : A0-7
bit 14 = 1 : A8-15

S : Store bit for memory reference instructions:
0 = store result in R1
1 = store result in memory

1.65 Addressing

Format 0 instructions (type 8) address the operand with the R1 field; no second operand is used. Format 1 instructions address the first operand with the R1 field; the second operand is addressed according to the addressing type, T1-T7, as listed in Table 1-2.

Table 1-2 Addressing Types

Type	Format 1 (K00) MD 9 10 (K9 K10)	R2 11	12	13	14	Effective Address of Operand	
T1	0 0	x	x	x	x	R2	Register-to-Register. R2 contains the operand .
T2	0 1	0	0	0	0	P	Long Constant. The following word(after the instruction) is the operand.
		(0R2)					
T3	0 1	non-zero $\overline{(0R2)}$				(R2)	Address in Register. R2 contains the address (A0-A15) of the operand.
T4	1 0	0	0	0	0	(P)	Address in Next Word. The following word is the operand address.
T5	1 0	non-zero $\overline{(0R2)}$				(P) + (R2)	Indexed Address. The following word, indexed by (A0-A15), as specified by R2, contains the operand address.
T6	1 1	0	0	0	0	[(P)]	Indirect Address. The following word specifies the location containing the operand address.
T7	1 1	non-zero $\overline{(0R2)}$				[(P) + (R2)]	Indirect Indexed Address. The following word, indexed by (A0-A15), specifies the location containing the operand address.
T8	Format 0 ($\overline{K00}$)						Short Constant. No 2nd operand is used.

1.66 All addressing uses a 16-bit address word, although only the 15 high-order bits are used for memory selection and for word-handling instructions. For character-handling instructions, the least-significant address bit specifies the character, as follows: bit 15 = 0: left character ; bit 15 = 1: right character.

1.67 OPERATION SEQUENCES

1.68 Microprograms, Microinstructions

CPU operations are controlled by microinstructions (μ Inst) located in the Microinstruction Store (Control ROM). Each microinstruction comprises a single 48-bit word divided into 14 command fields.

1.69 A microprogram performs one part of an instruction (e.g. indirect addressing or execution) or one operating sequence (e.g. Initial Program Loading or an Interrupt routine). The microprogram may consist of a single microinstruction (Fetch, SUK, etc.) or a group of microinstructions accessed in a specified sequence. This sequence may vary according to conditions specified in the microinstructions. An example of microinstructions and microprograms grouped into an instruction is shown in Figure 1-7, Operation Terminology.

1.70 Flow Diagrams

General operational flow for the CPU is shown in Figure 1-8. Detailed flow diagrams for each block are referenced on Figure 1-8 and, for the execute-instruction sequences, in the instruction list (Table 1-2). A key to the flow diagrams is provided in Figure 1-7.

1.71 Figure 1-8 shows the general sequence of computer operations. The computer continuously performs microinstructions. The sequences in which the microinstructions are performed are determined by microinstruction addressing (paragraph 2.26).

1.72 If no program is running and the computer is not being operated, it cycles through the Idle loop. RUNF is described with the PSW, paragraph 2.90. Machine-state-pointer control is described with the microinstruction addressing. RUNF is set by pressing START on the control panel. If the control panel is used (Section III), PUP is set in the CPU (logic diagram CC) and the operation now cycles through the control-panel path, beginning with microinstruction /010. Either automatic restart or IPL may be initiated through this path.

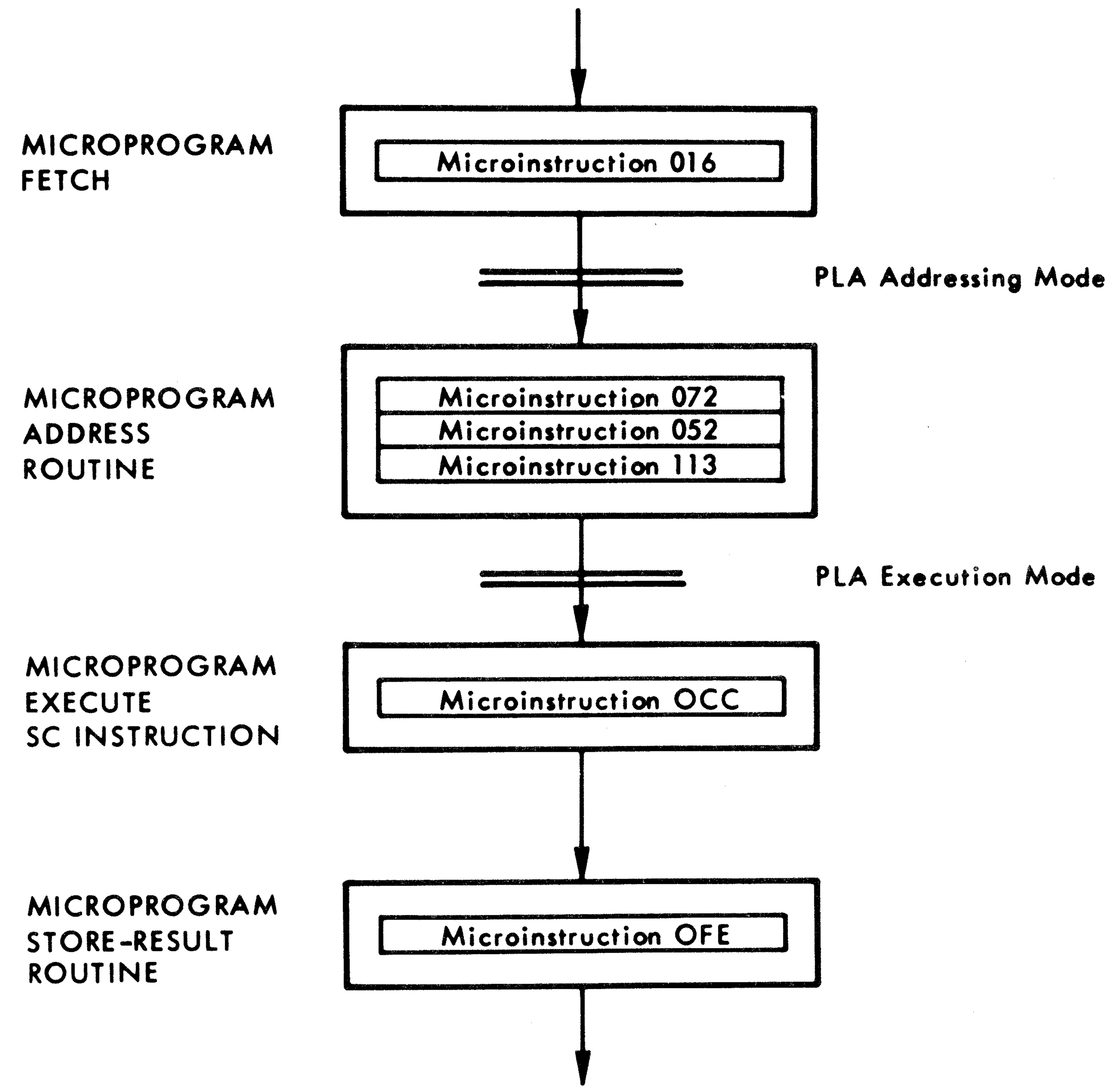
1.73 If RUNF is set and the control panel is not selected, the machine-state-pointer tests for interrupts, and executes the interrupt routine if necessary. KRY is set when the K-register is loaded with the instruction word. KRY is set by the Fetch command during the Fetch routine, some control-panel routines, EX instruction, and some tests.

1.74 Before KRY is set, the computer loops directly to Fetch to load the instruction word into the K register. The instruction must be requested from memory during a preceeding microinstruction. When an instruction is loaded, KRY is set and the PLA addressing mode is used to select the next microinstruction address. For most instructions, the addressing routine is used to obtain the operand. Instruction word addressing and decoding is described in paragraphs 2.37 and 2.48.

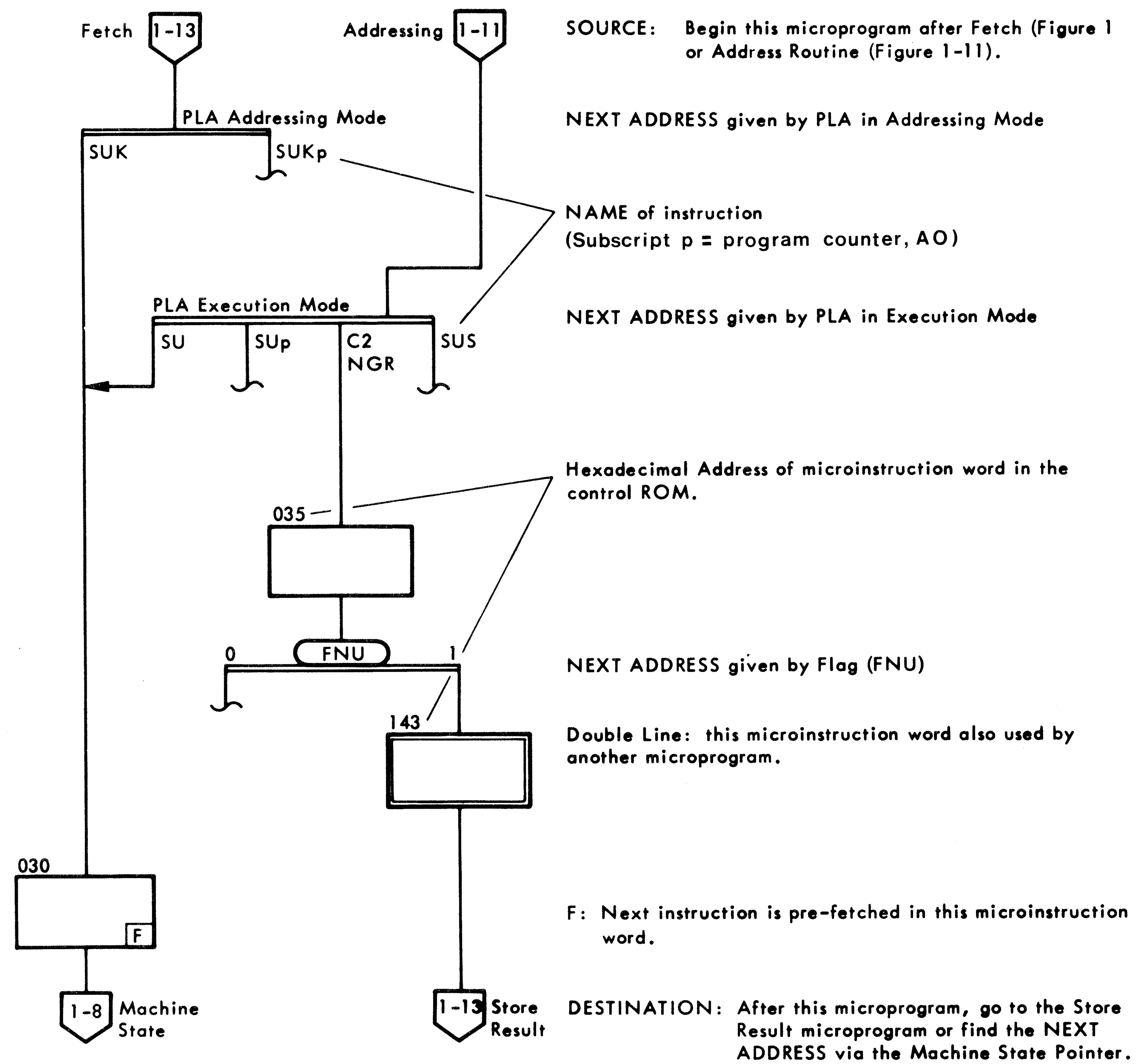
1.75 Repeated microinstructions of a routine are selected with explicit addressing (the microinstruction includes the address of the next microinstruction). Microprogram decision tests may modify the explicit addressing with a flag bit (SNA Flag mode). At the end of a microprogram routine, a Bus Request is made for the next instruction word and the Fetch routine is then used to load the instruction. If the machine state pointers allow (program still running, no control-panel operations or interrupts pending), the computer selects the next microinstruction address via the PLA addressing, and continues with the next microprogram.

1.76 Data paths are shown in Figures 2-1 and 2-4. The microinstructions within each microprogram or routine control the data paths and control the operations performed on the data. These controls are described (in Section II) for each of the logic blocks shown on the data-path diagrams. The data handling is also listed on the microprogram flow charts with terminology such as: $M + 2 \rightarrow M$, where the data in the M register loops through the data paths shown on Figures 2-1, 2-4 and back to M, with a left-shift being performed enroute to produce the +2.

Example of one instruction (SC-Store Character) using addressing mode T6 (indirect)



Part A



Address Type Designations	
T1 (2nd Operand in R2):	T2-7
RT1 RT1P - R2=0, thus operand in P. RT1D - Double Length; (R2) → Q, (R2+1) → M RT1DP - Double Length and R2=0	RT2-7 Result → R1 RT2-7S - Store; Result → memory RT2-7C - Constant T3A — R2 ≠ A15 T3B — R2 = A15

Part B

Figure 1-7 Operation Terminology, Flow Diagram Key

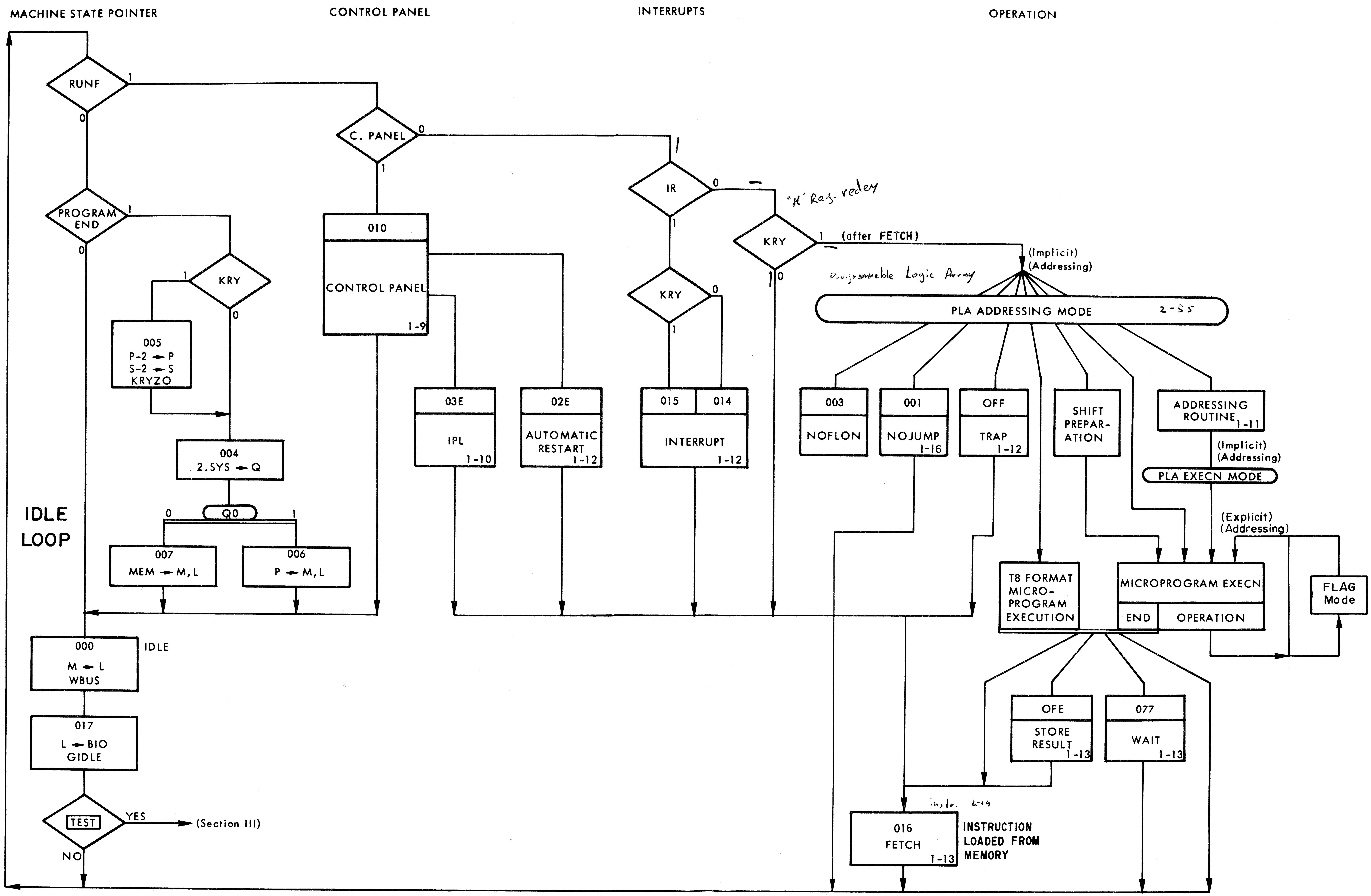


Figure 1-8 CPU Operational Flow and Machine State Pointer

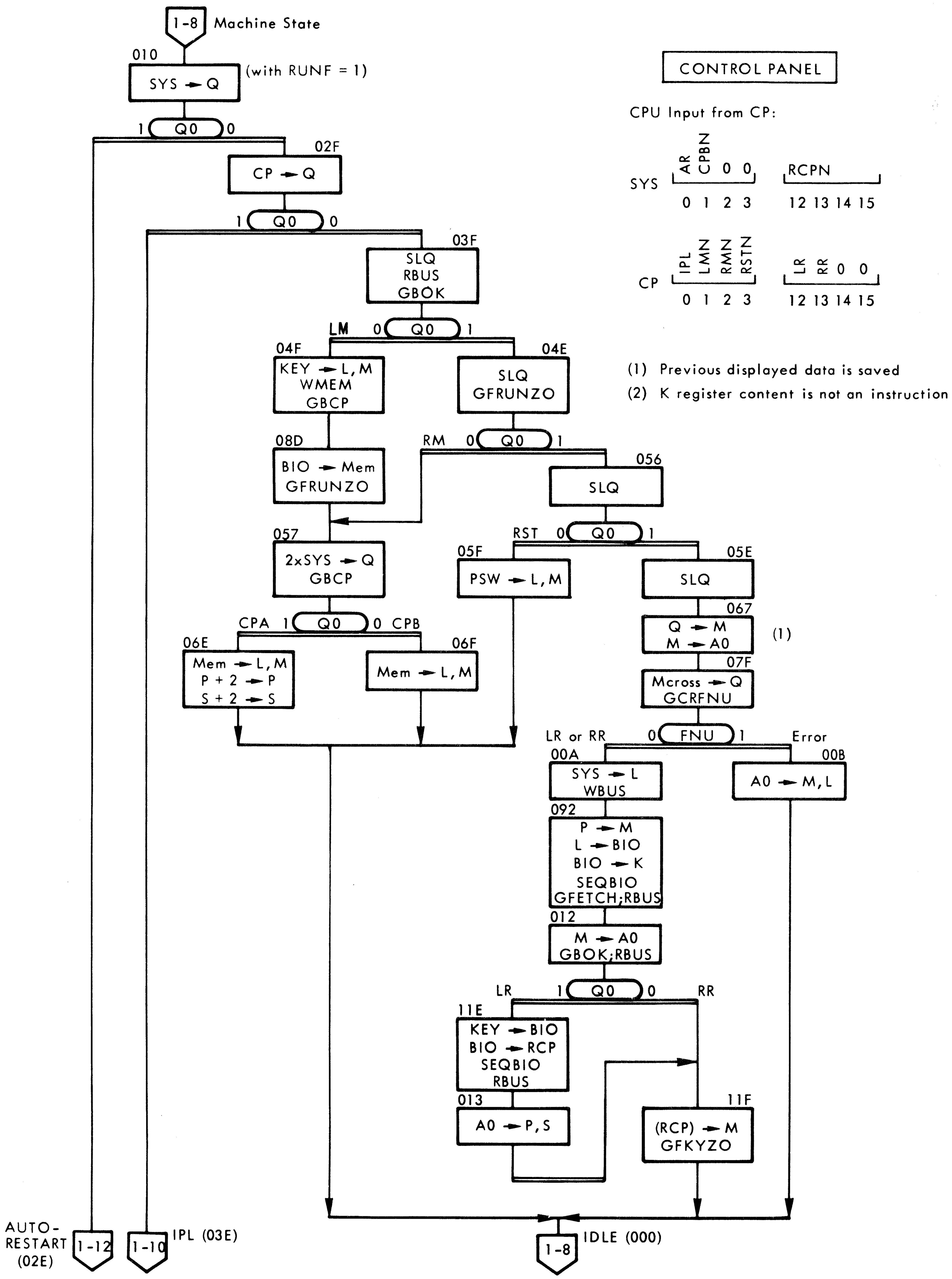
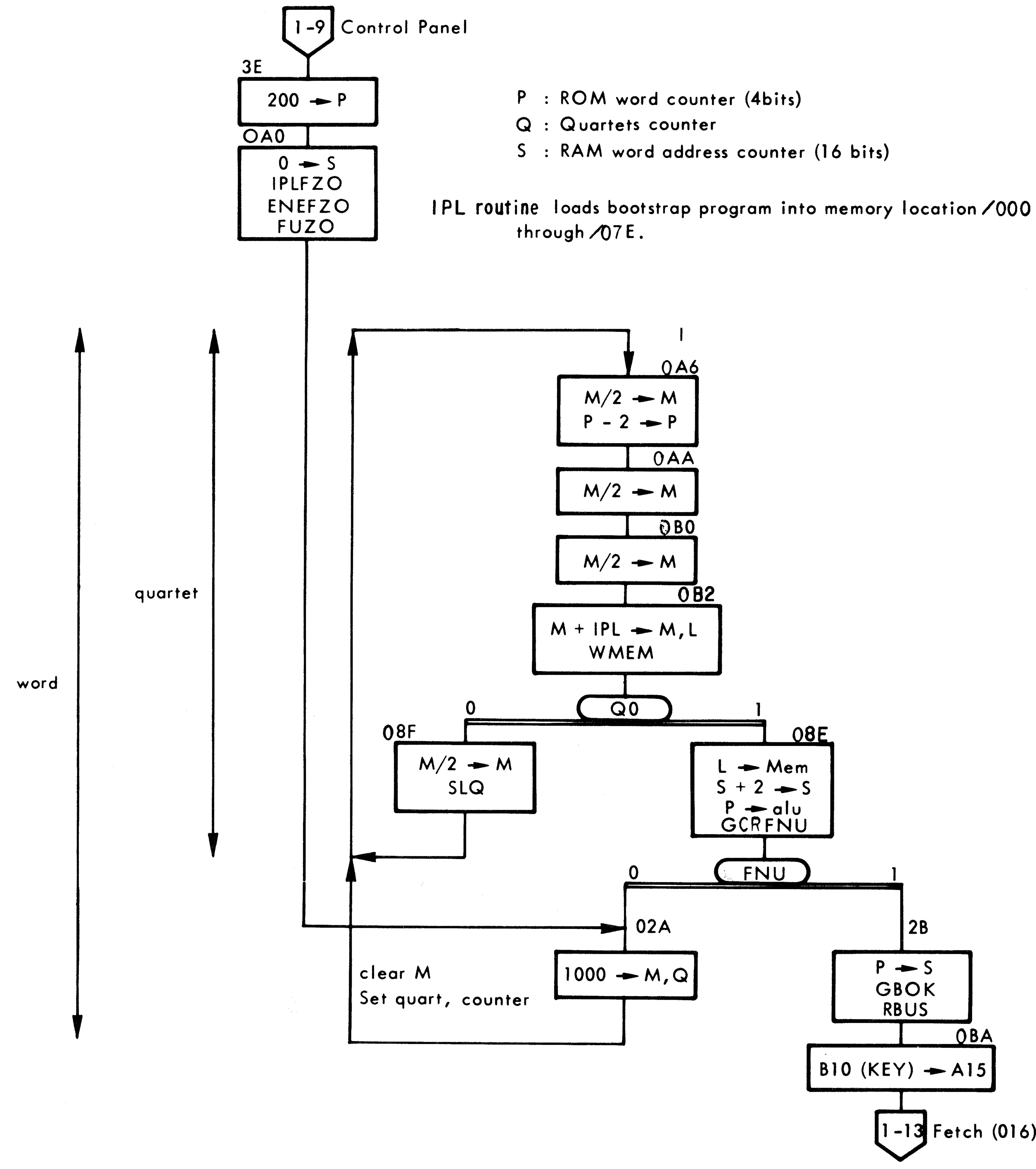


Figure 1-9 Control Panel



NOTE:

- Bootstrap listing in Table 2 - 10
- Loading routine in Paragraph 3.14

Figure 1-10 IPL

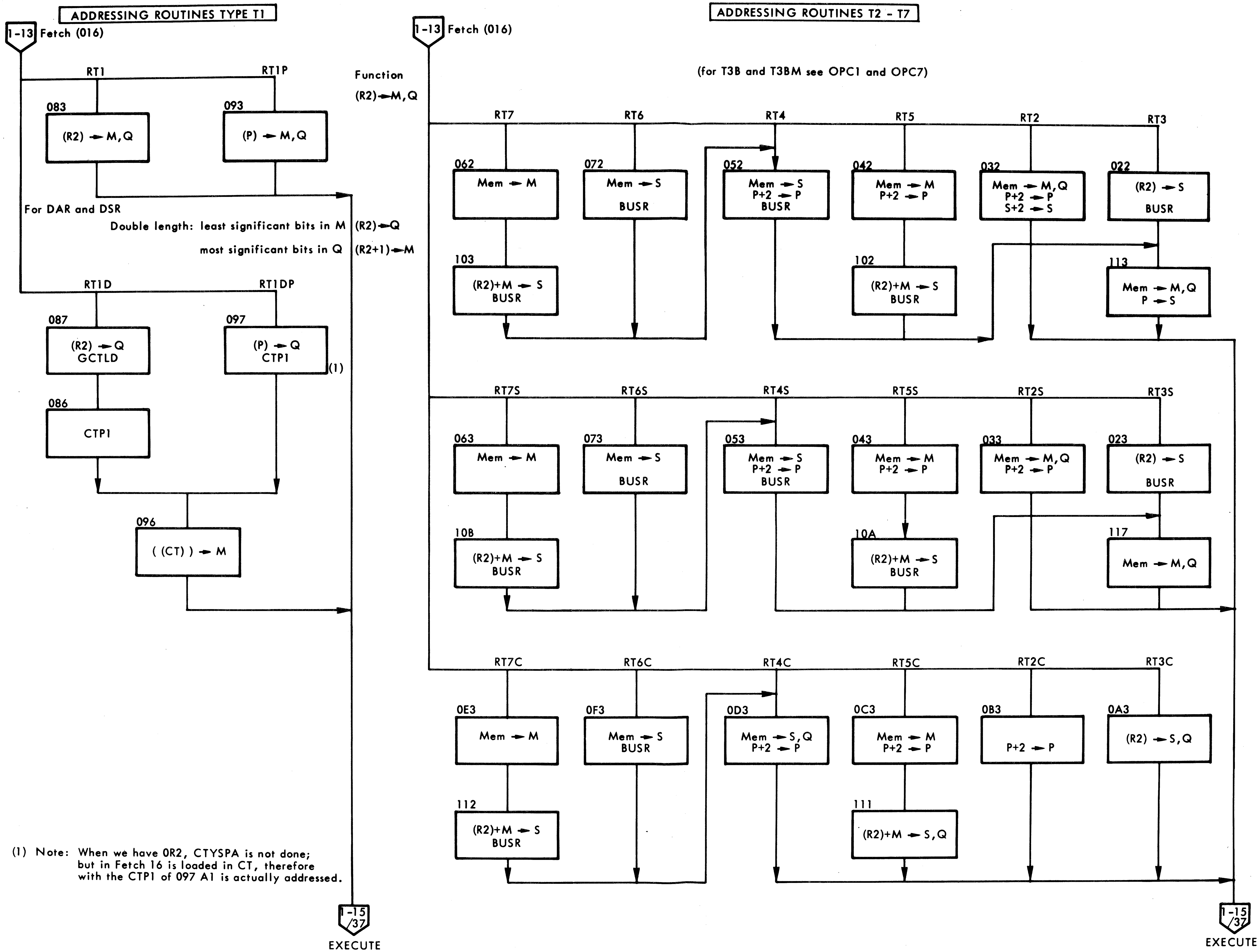


Figure 1-11 Addressing

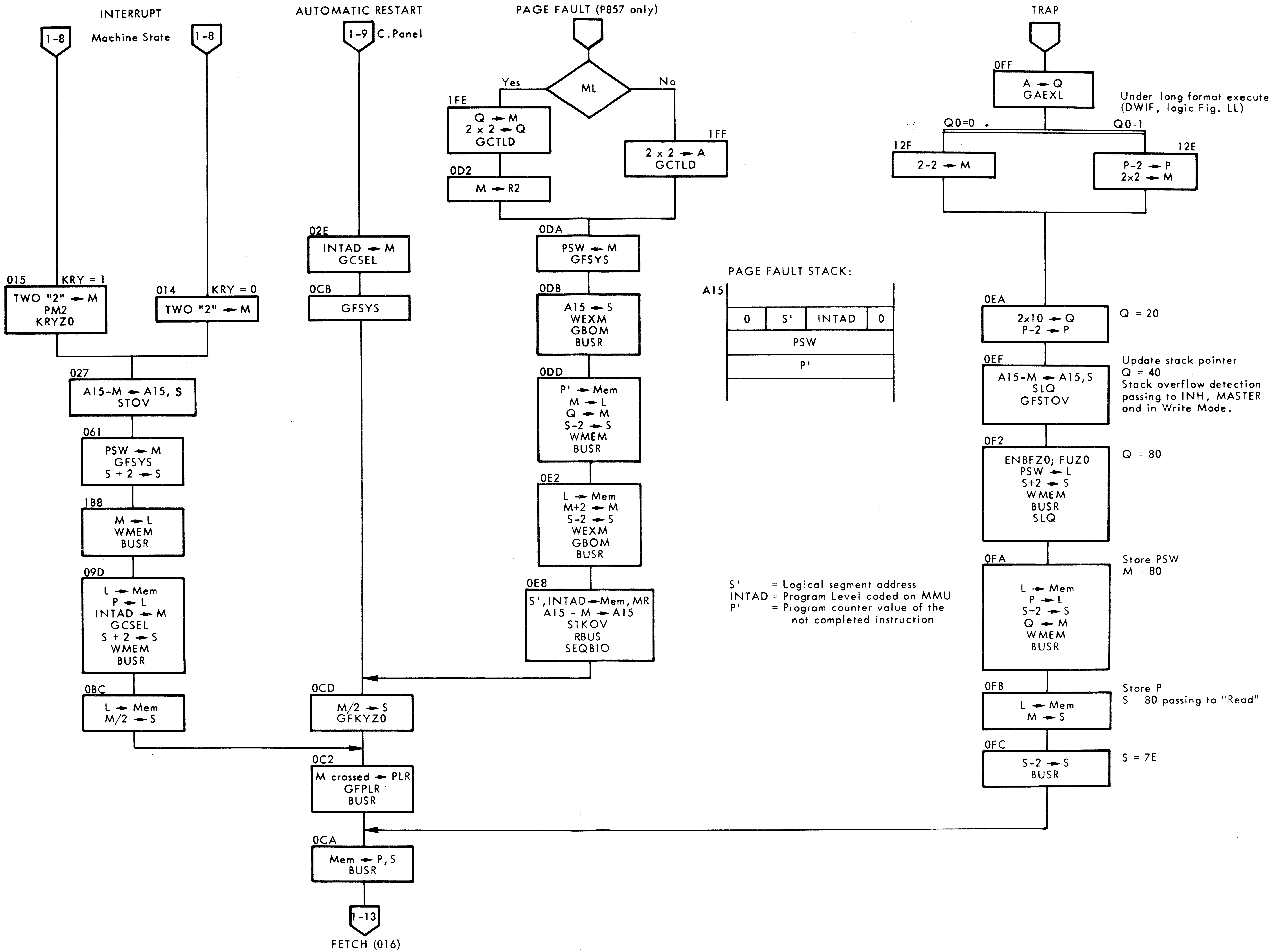


Figure 1-12 Interrupt, Restart, Fault, Trap

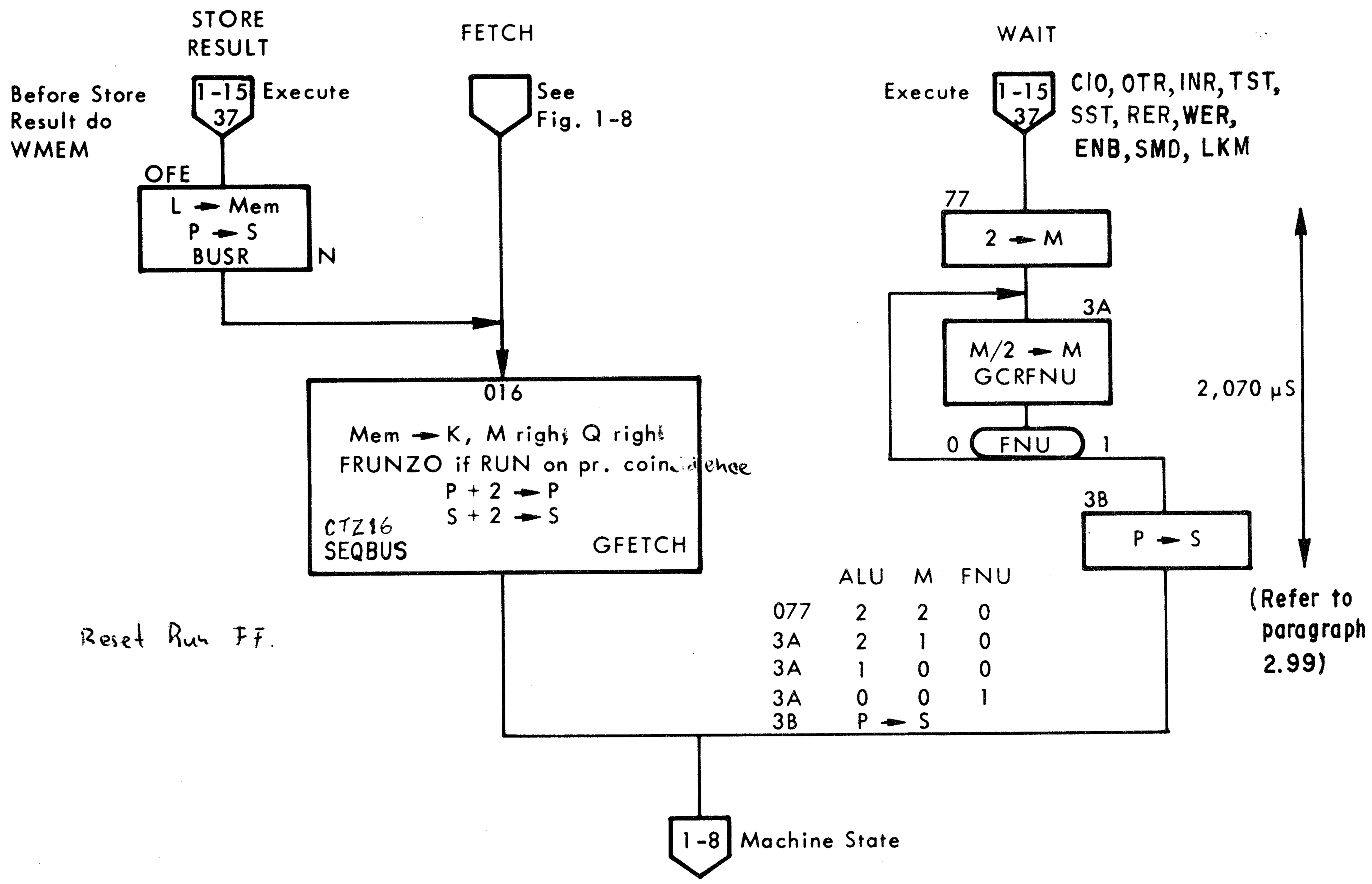


Figure 1-13 Store, Fetch, Wait

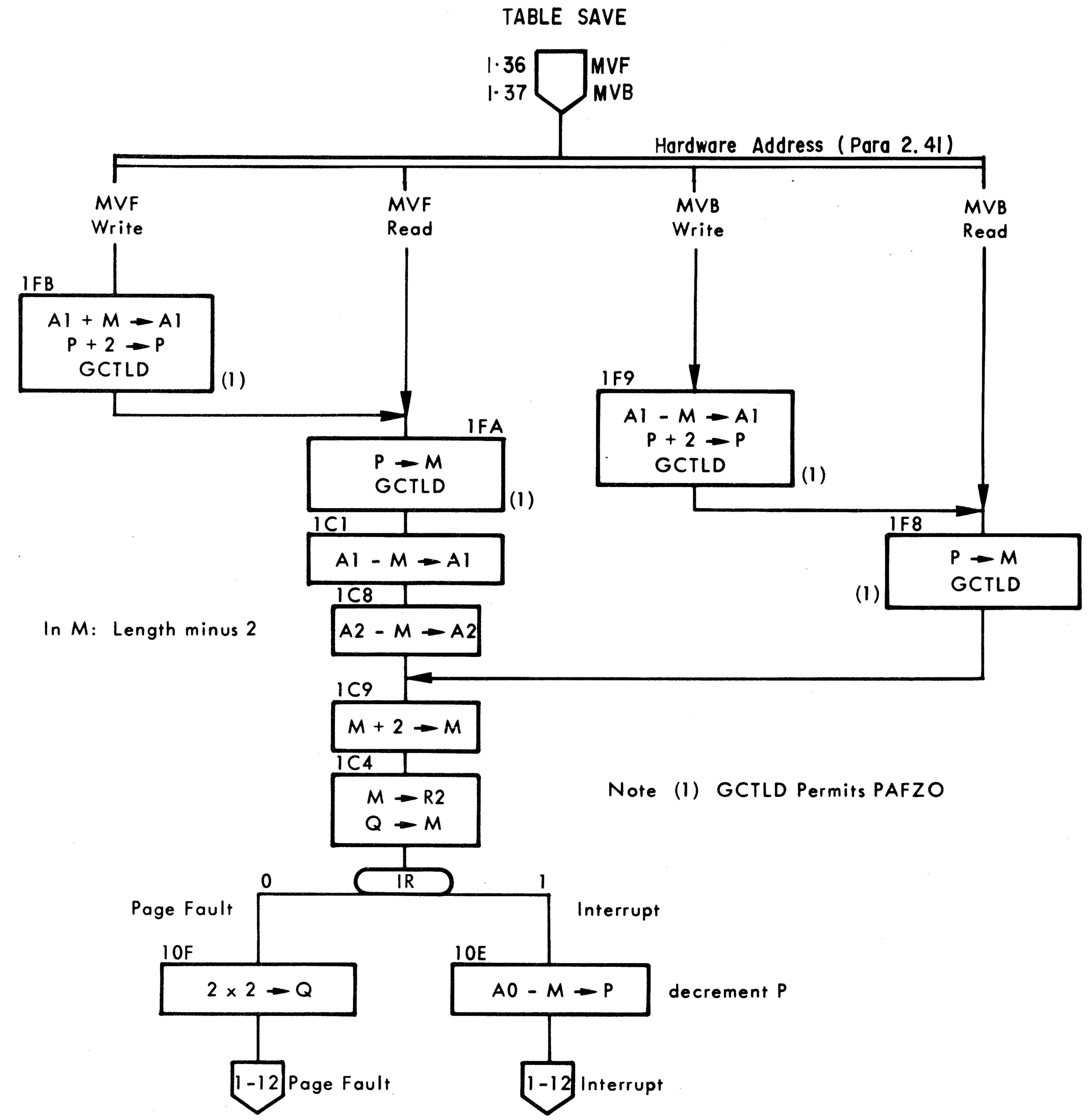


Figure 1-14 Table Save (P857 only)

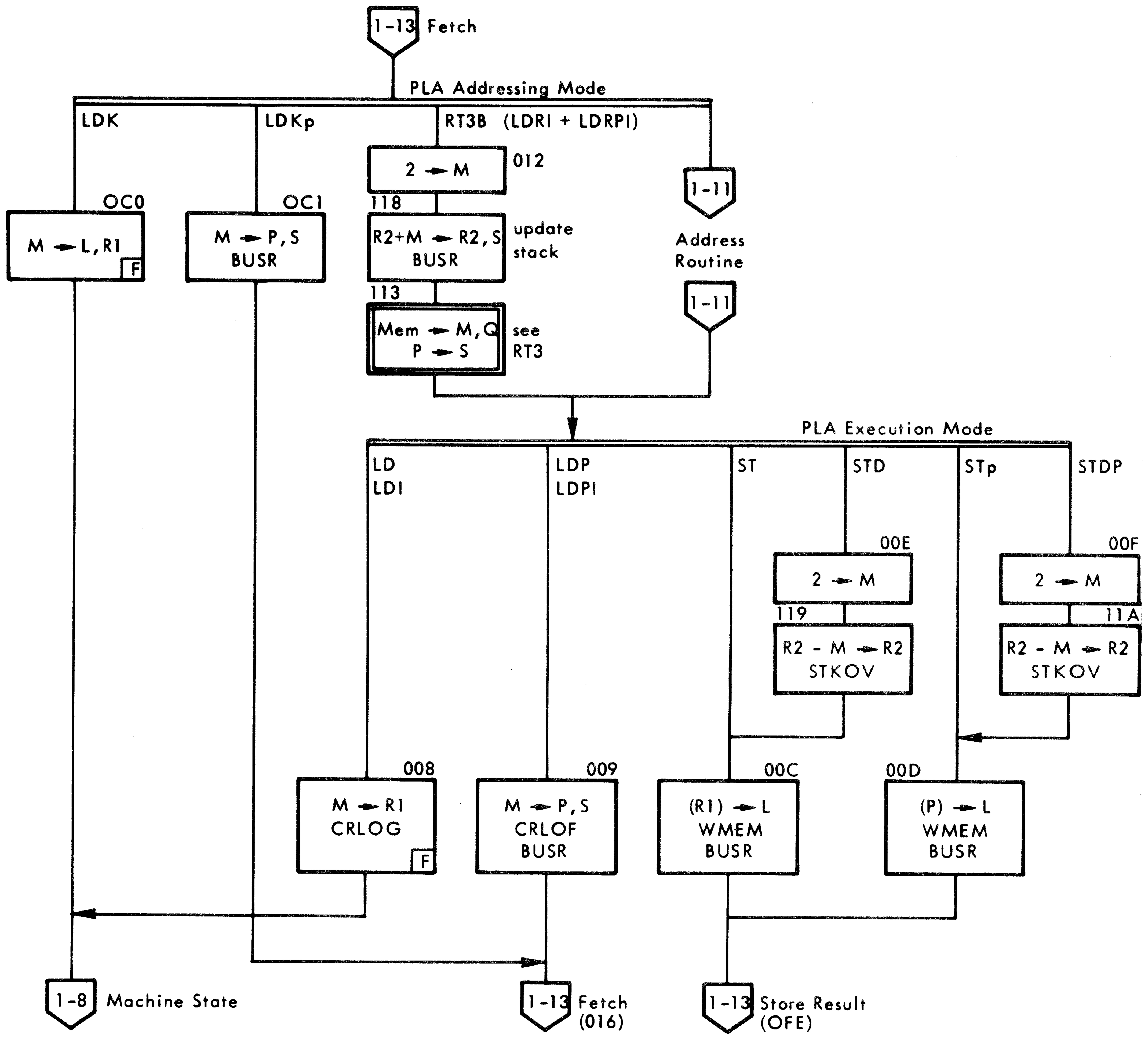


Figure 1-15 OPC 0 (LD, ST)

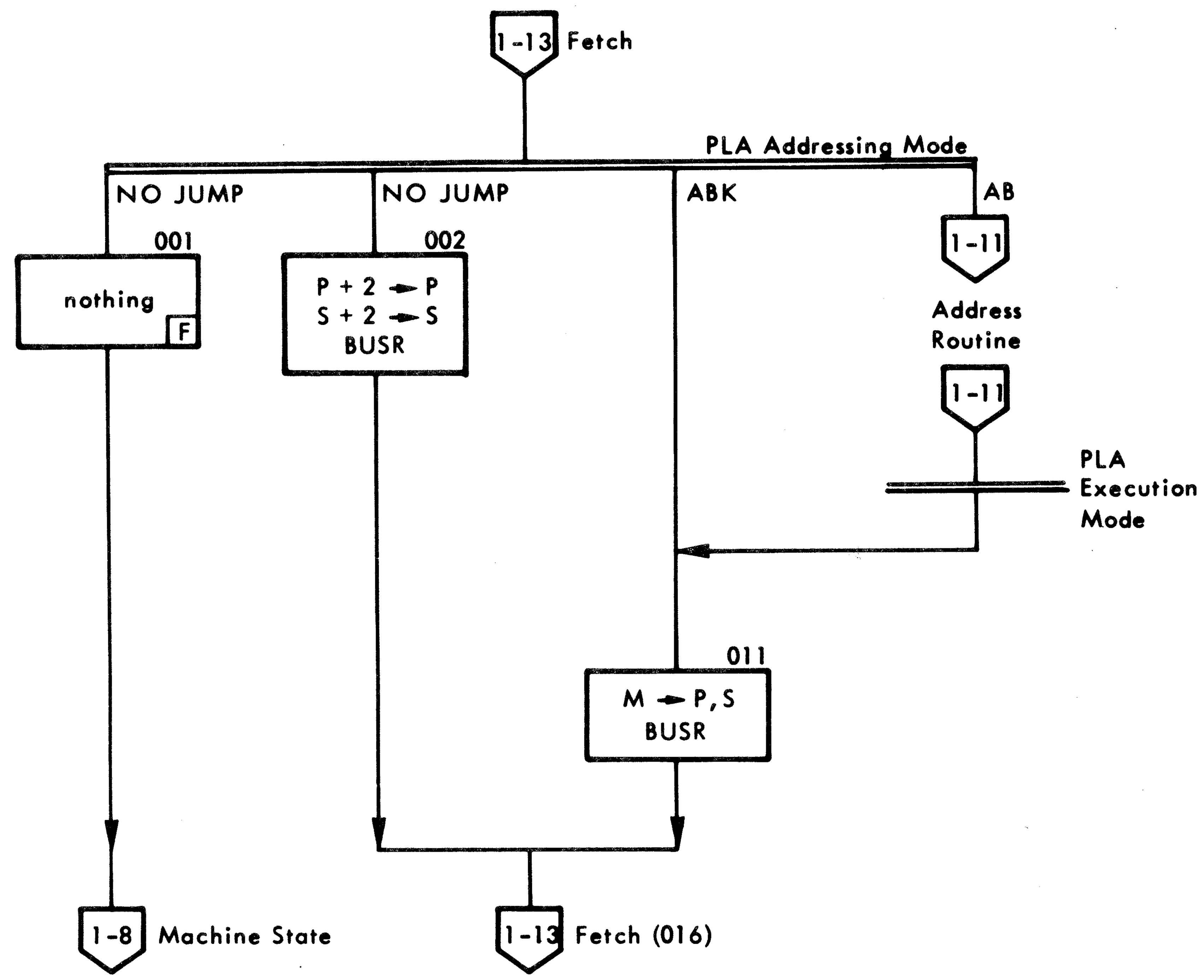


Figure 1-16 OPC 1 (AB)

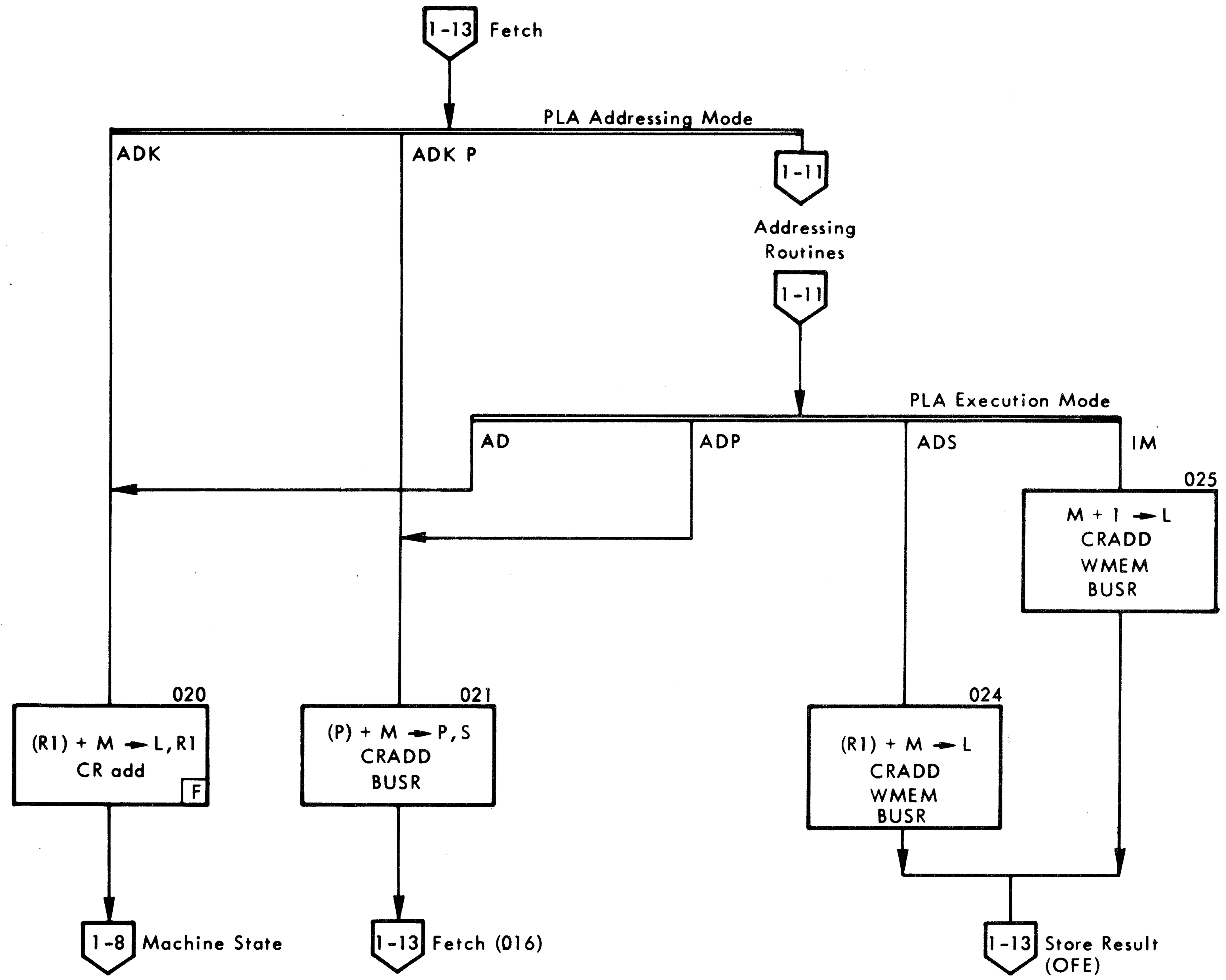
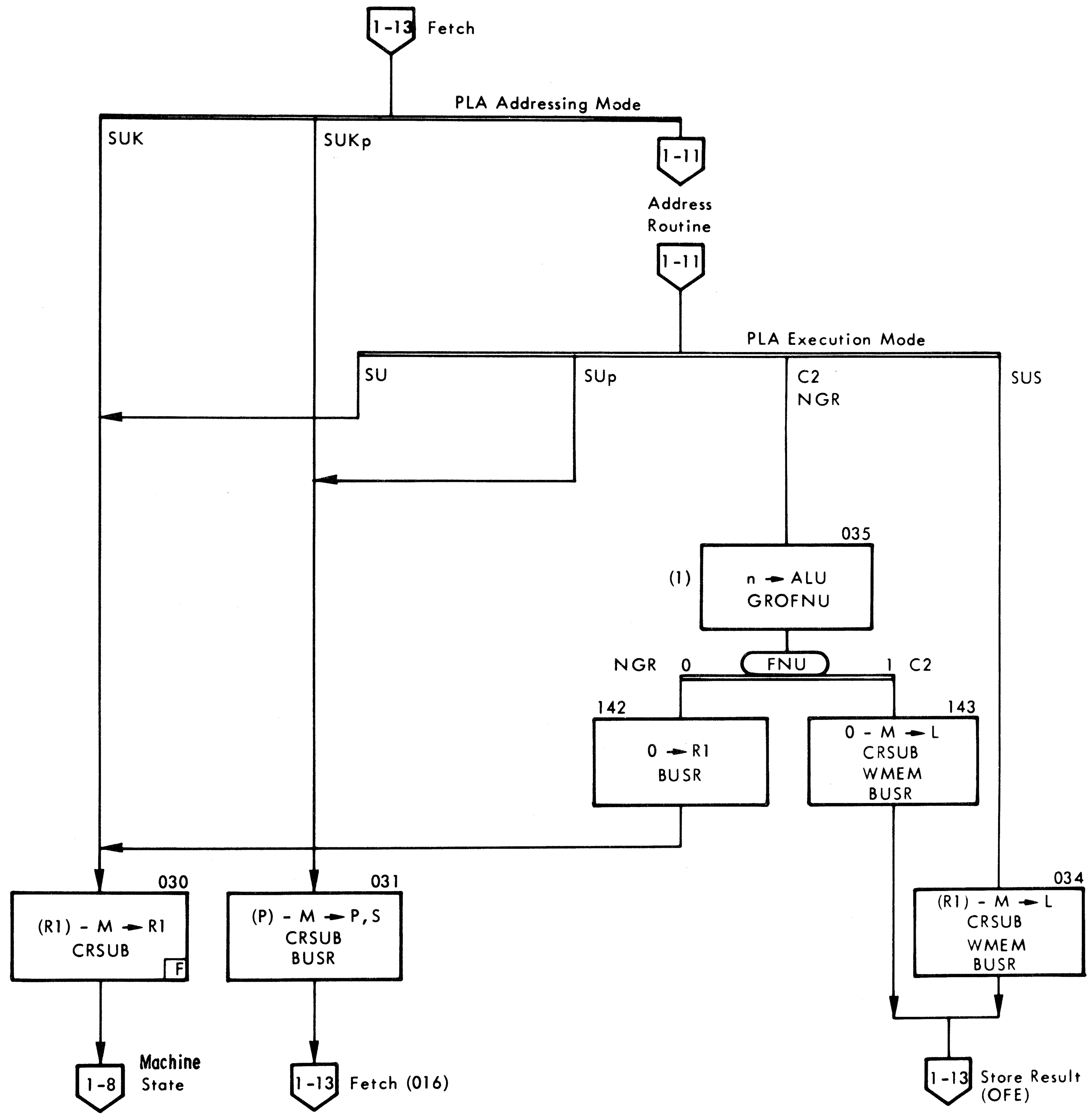


Figure 1-17 OPC 2 (AD, IM)



(1) n = K05, K06, K07, K08, 0 if n = 0 (OR1) ⇒ C2 if n ≠ 0 (OR1/) ⇒ NGR

Figure 1-18 OPC 3 (SU, NG, C2)

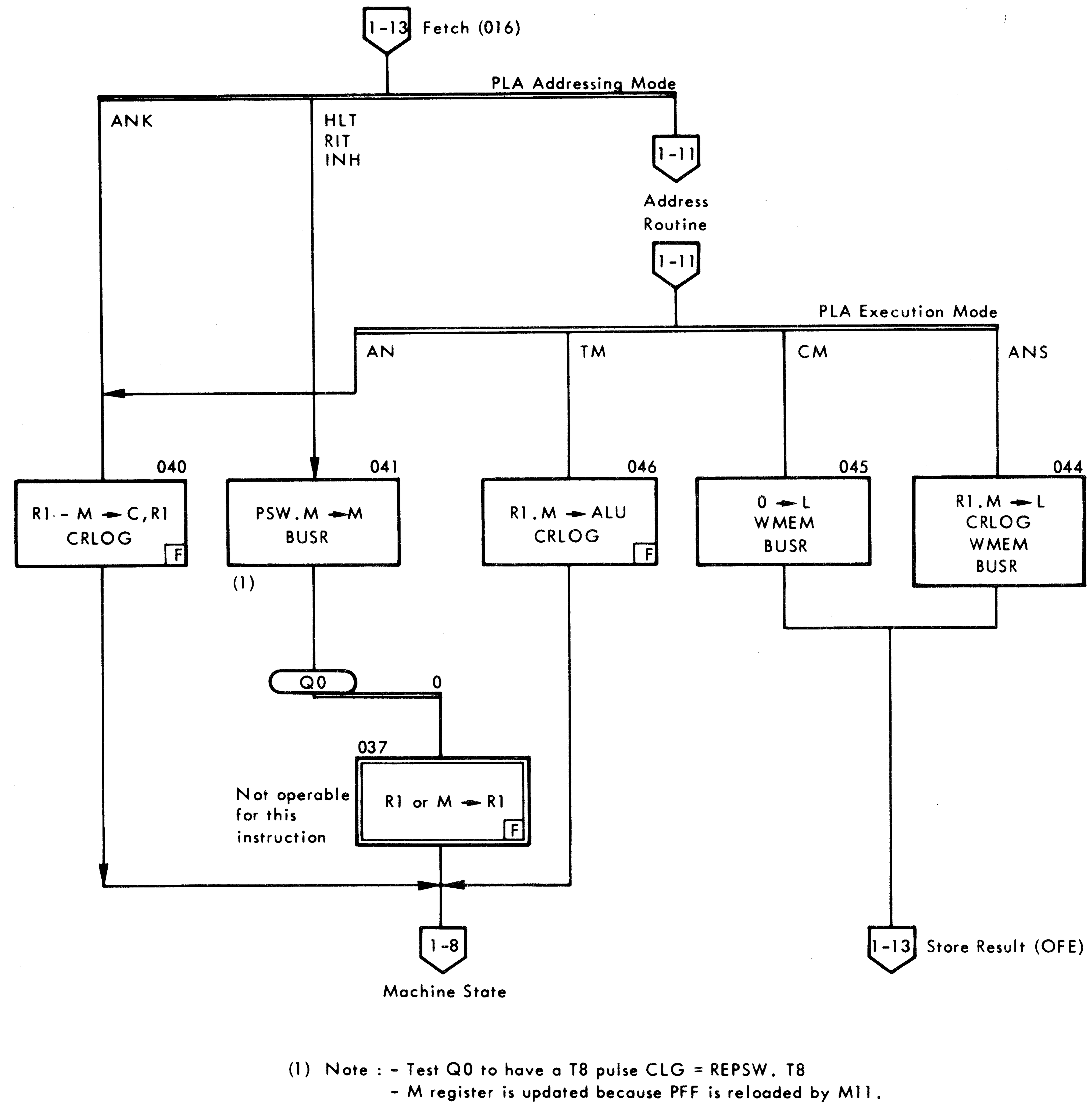


Figure 1-19 OPC 4 (AN, TM, CM, AC, HLT, INH, RIT)

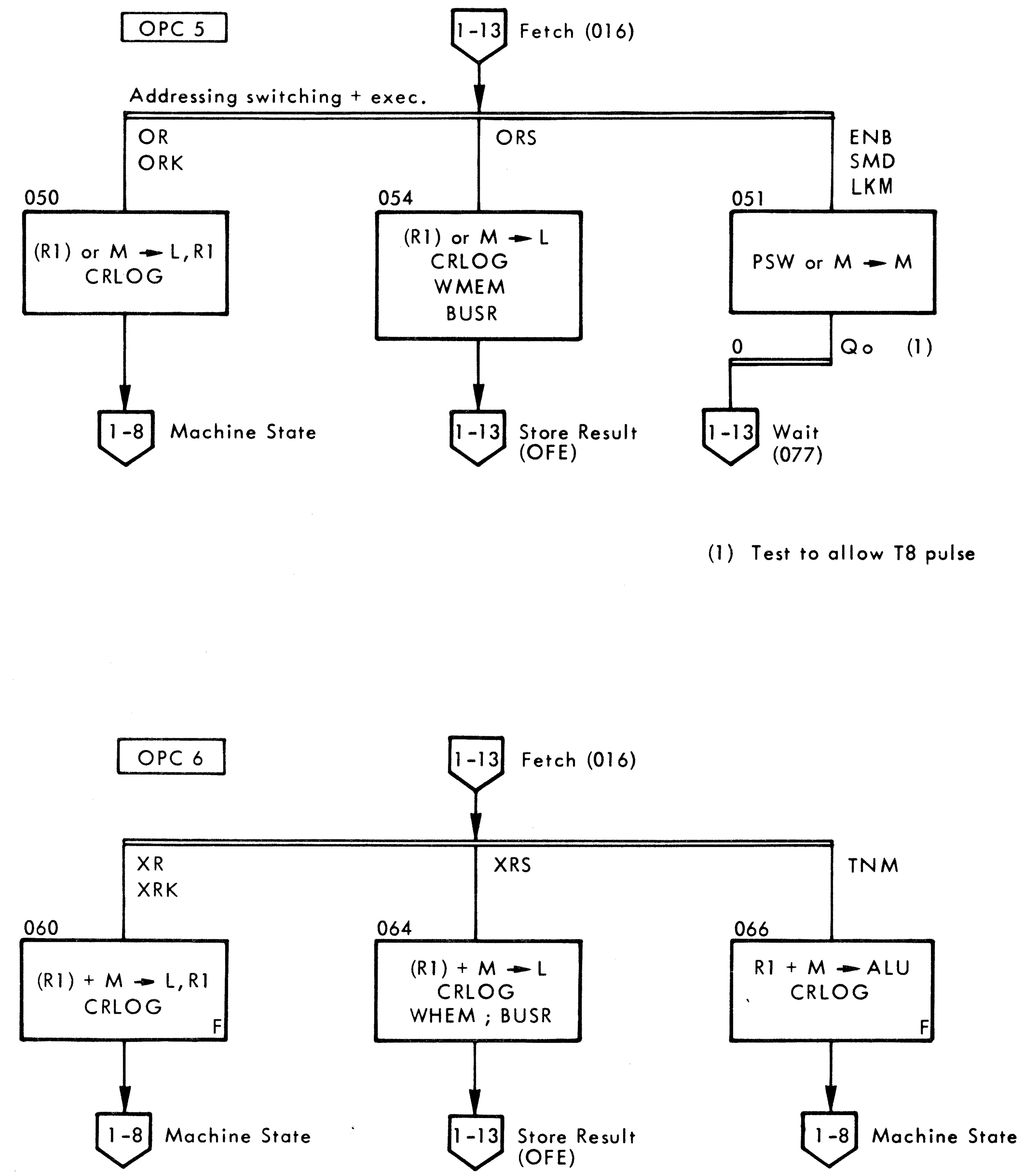


Figure 1-20 OPC 5, 6 (OR, ENB, LKM, SMD, XR, TNM)

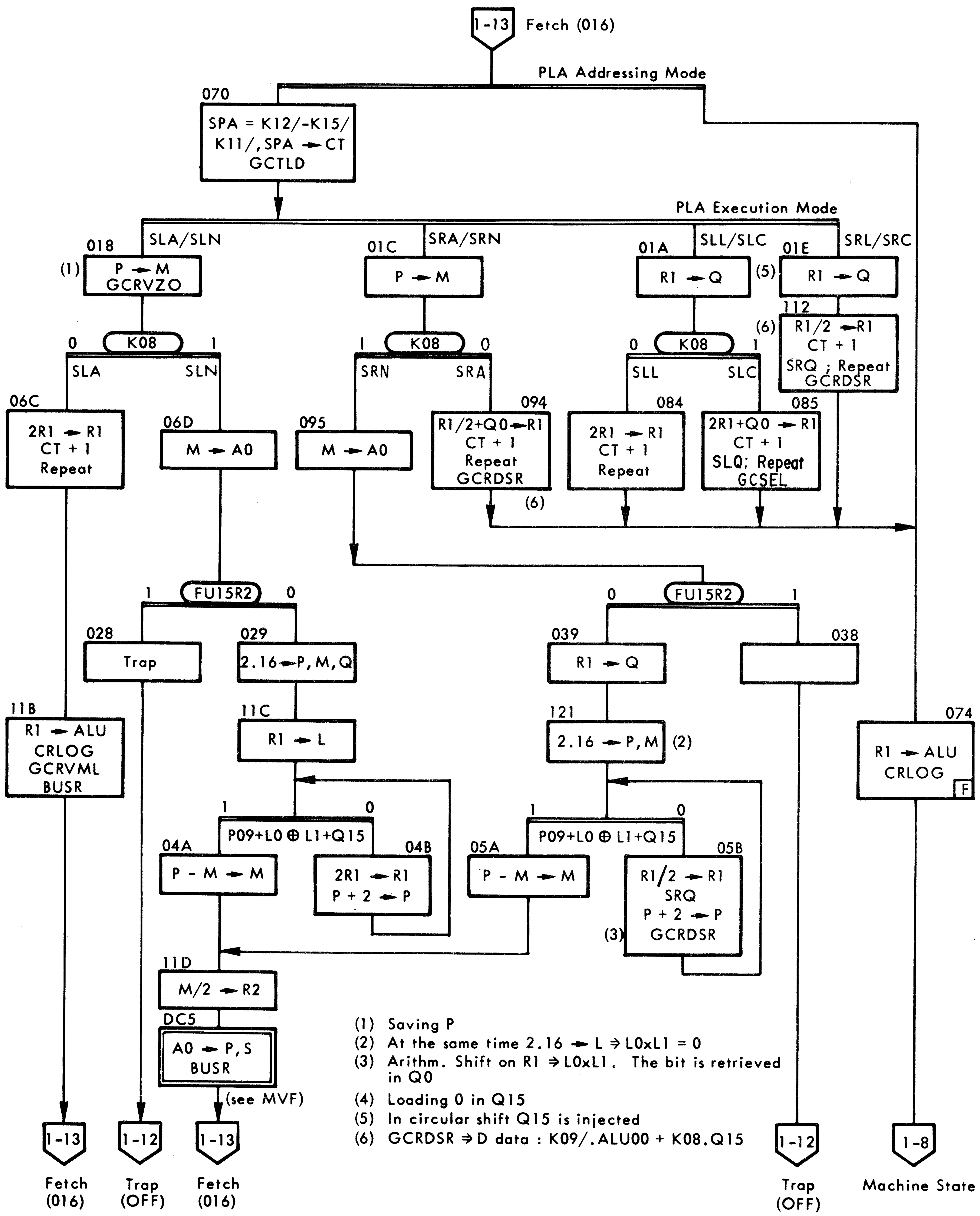


Figure 1-21 OPC 7 Single Shifts (SL, S R)

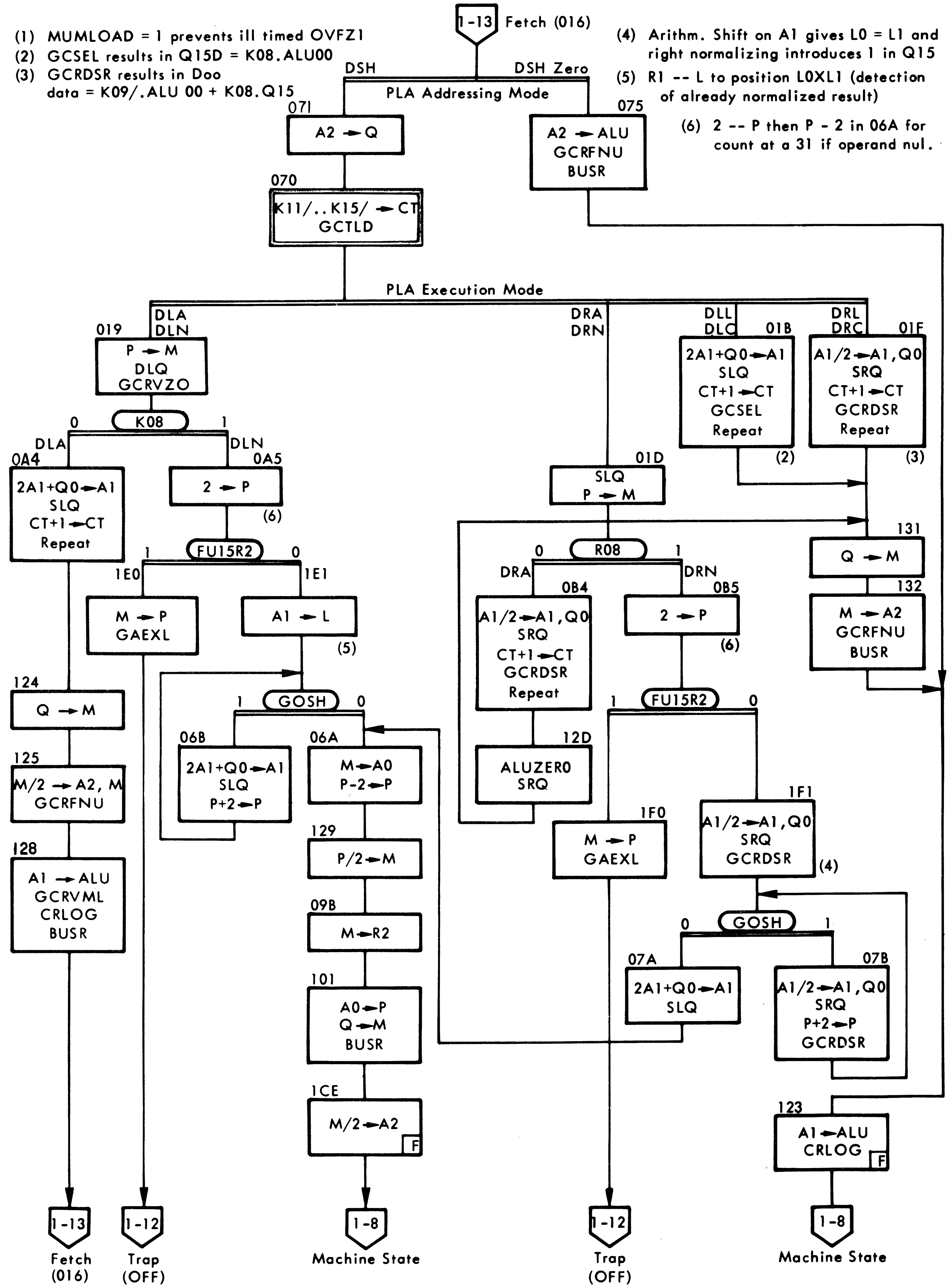
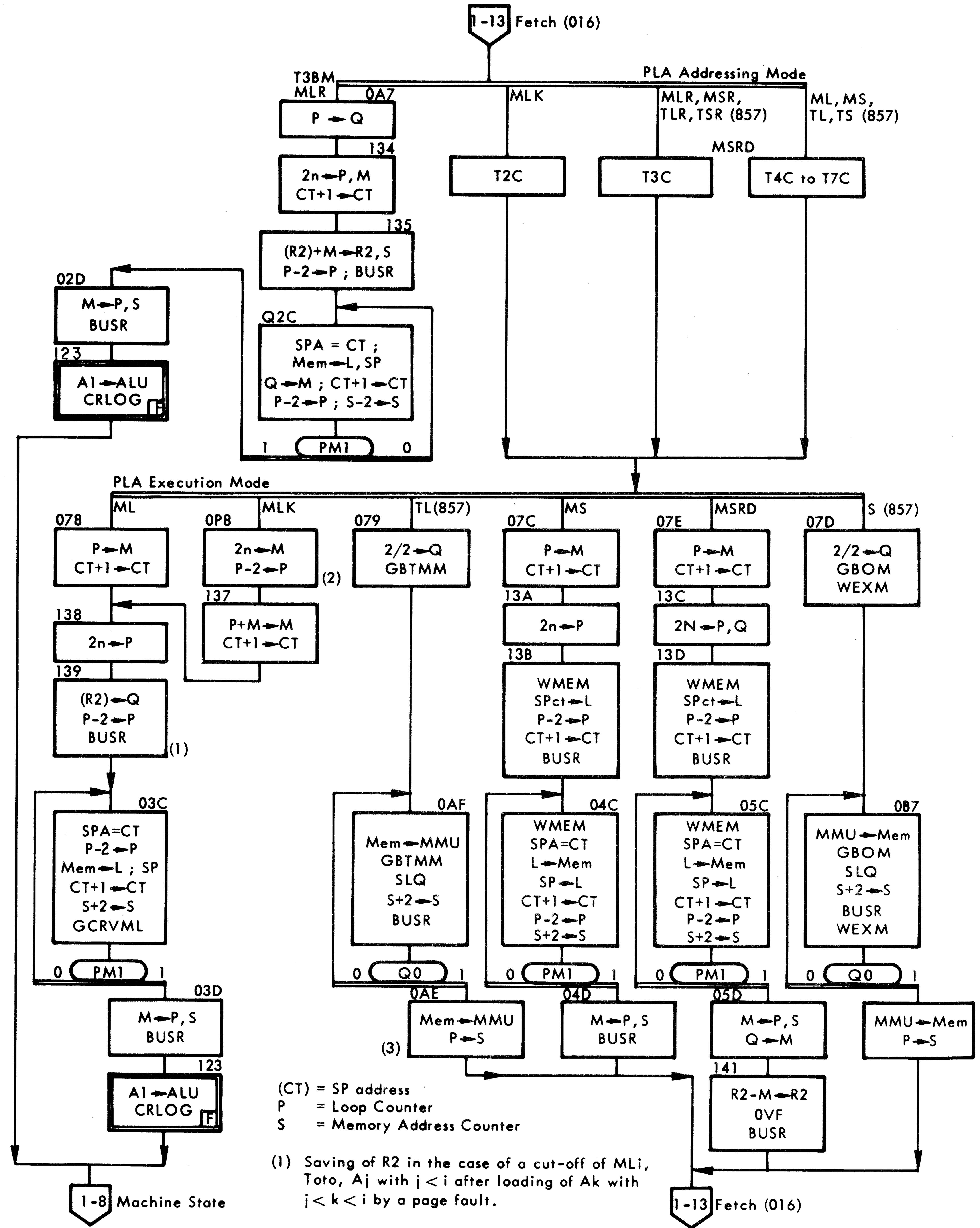


Figure 1-22 OPC 7 Double Shifts (DL, DR)

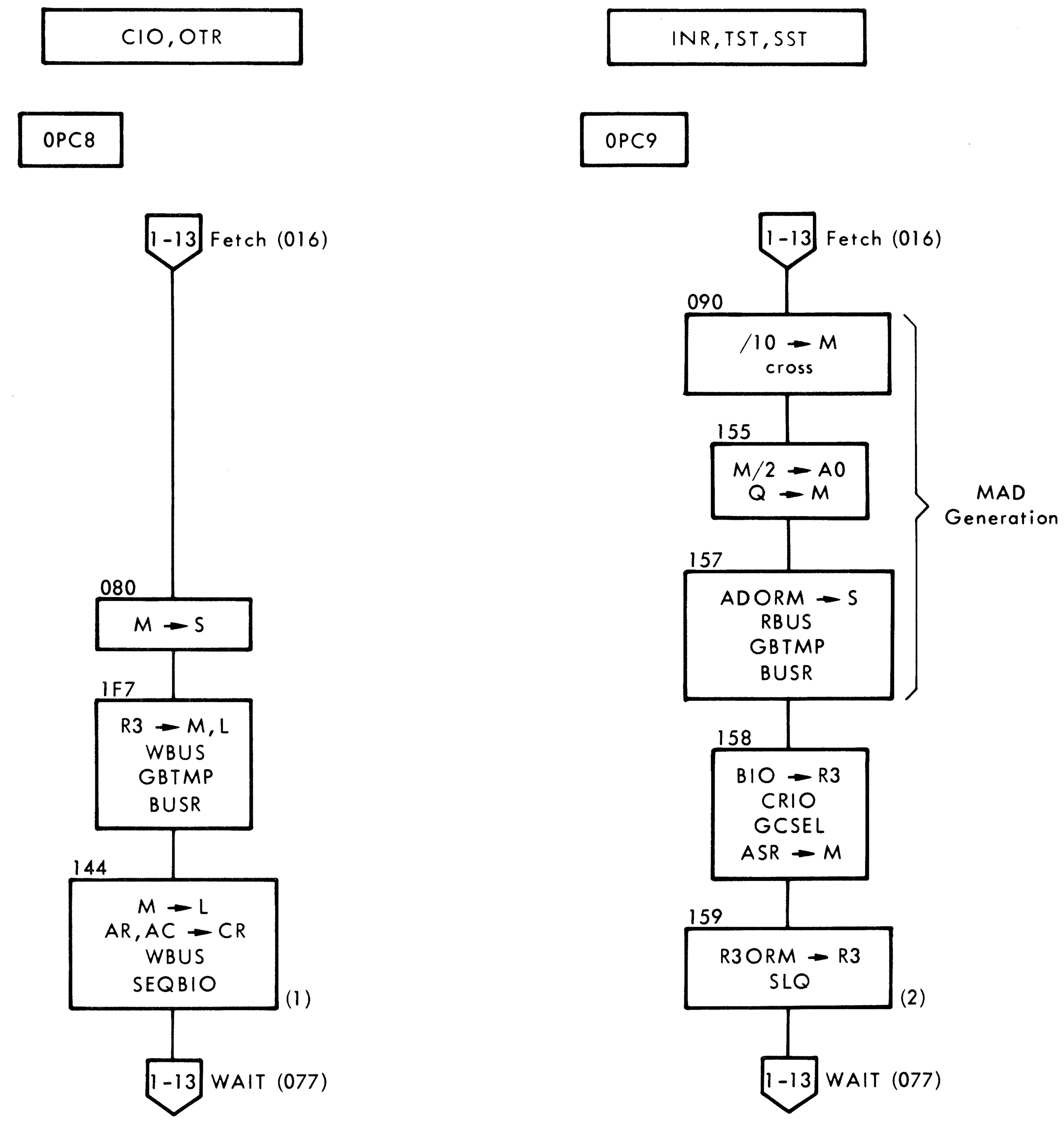


(1) Saving of R2 in the case of a cut-off of MLi, Toto, Aj with j < i after loading of Ak with j < k < i by a page fault.

(2) P has already been incremented in RT2C.

(3) GBMMU does MMU → BIO if K15 = 1 i.e. MMUYBIO if K15 = 0.

Figure 1-23 OPC 7 Multiple Load/Store, Table Load/Store (ML/MS, TL/TS)



(1) - M → L and WBUS maintain BIO for 120 ns after ending edge of TMR. In the same cycle SEQBIO inhibits the BSYCPU resetting on T6.
- WBUS sets WRITE to "one"; so in INR, SST, TST, WRITE = 0 and in CIO, OTR, WRITE = 1.

(2) SLQ for MUQ1 = 1 permits clock echo if FACINR.

Figure 1-24 OPC 8, 9 I/O Instructions

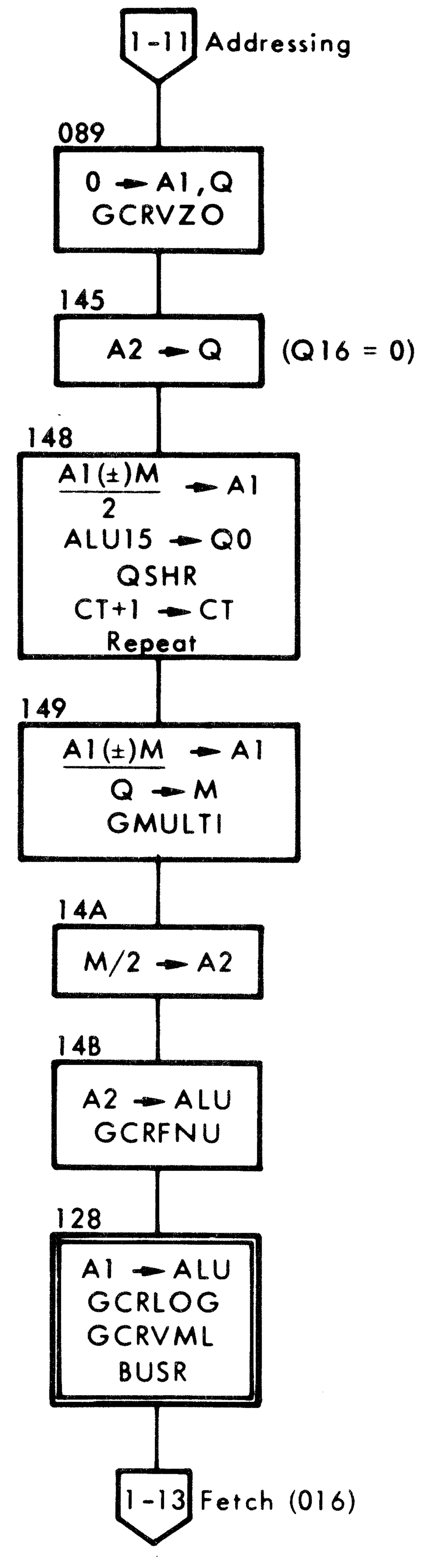
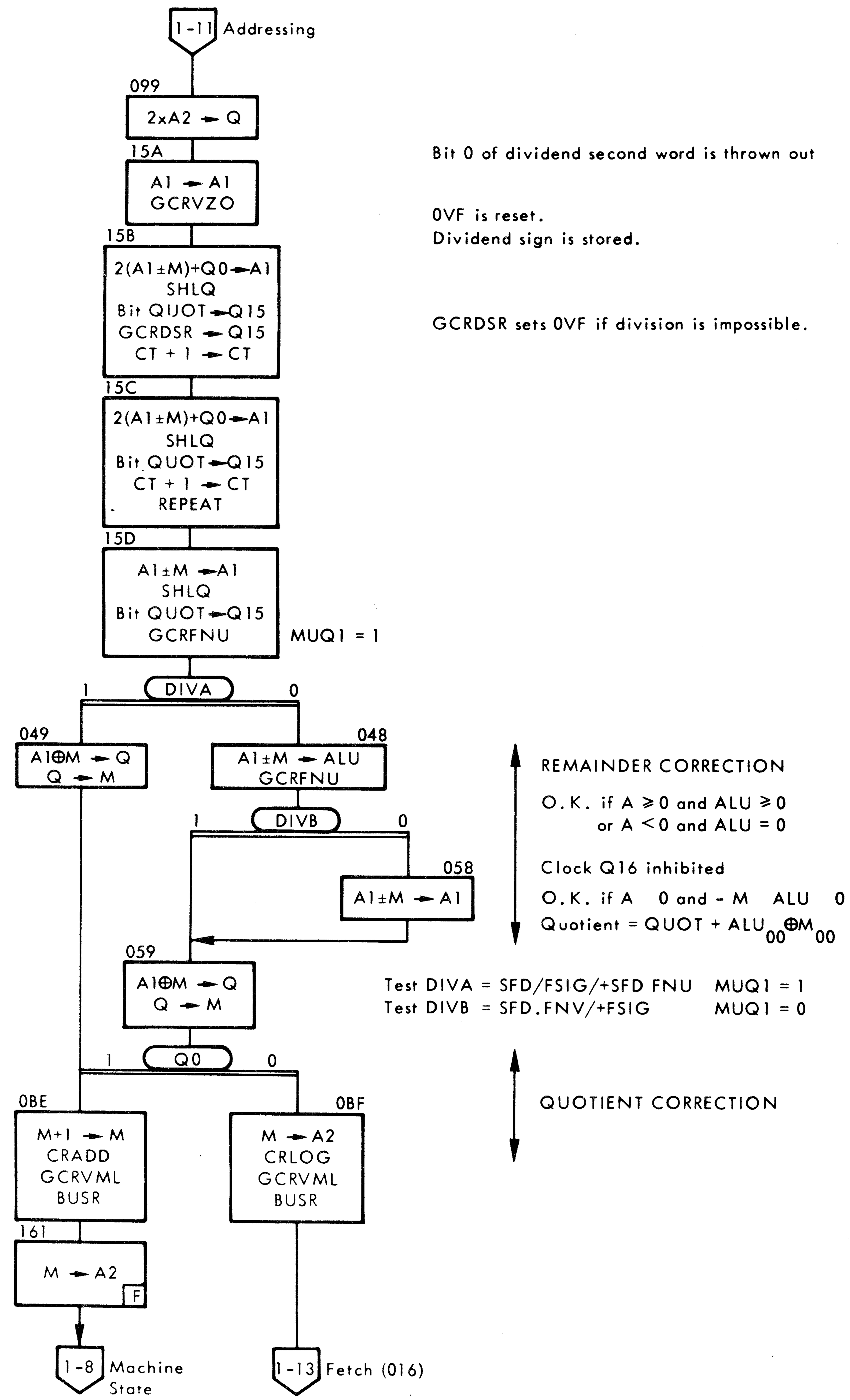
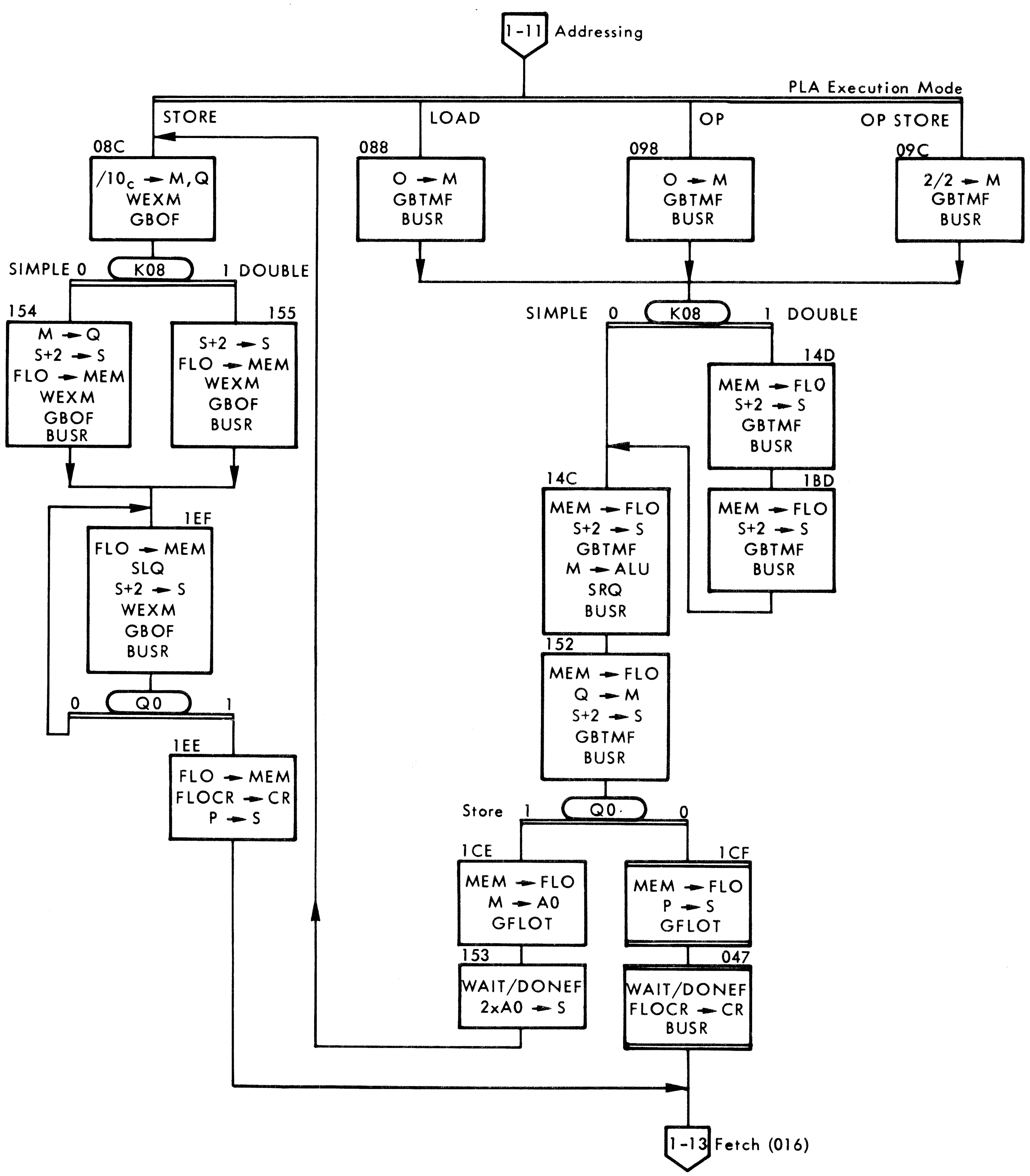


Figure 1-25 OPC 8 Multiplication (MU)



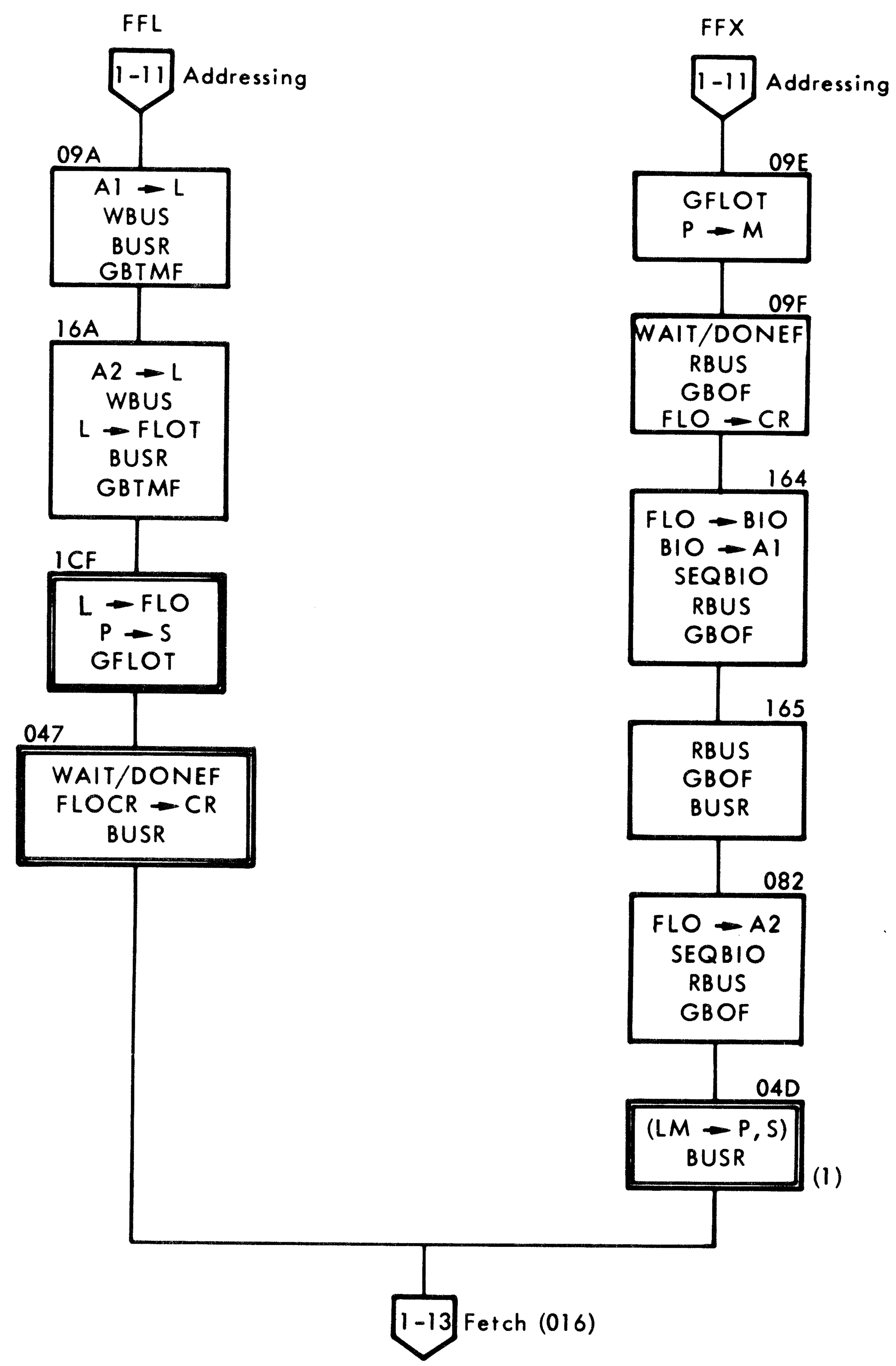
1-26 OPC 9 Division (DV)



After the addressing routines, the effective address of the operant is in Q register (see routines type "C")
(1) contents of Q are now: | S | AD |

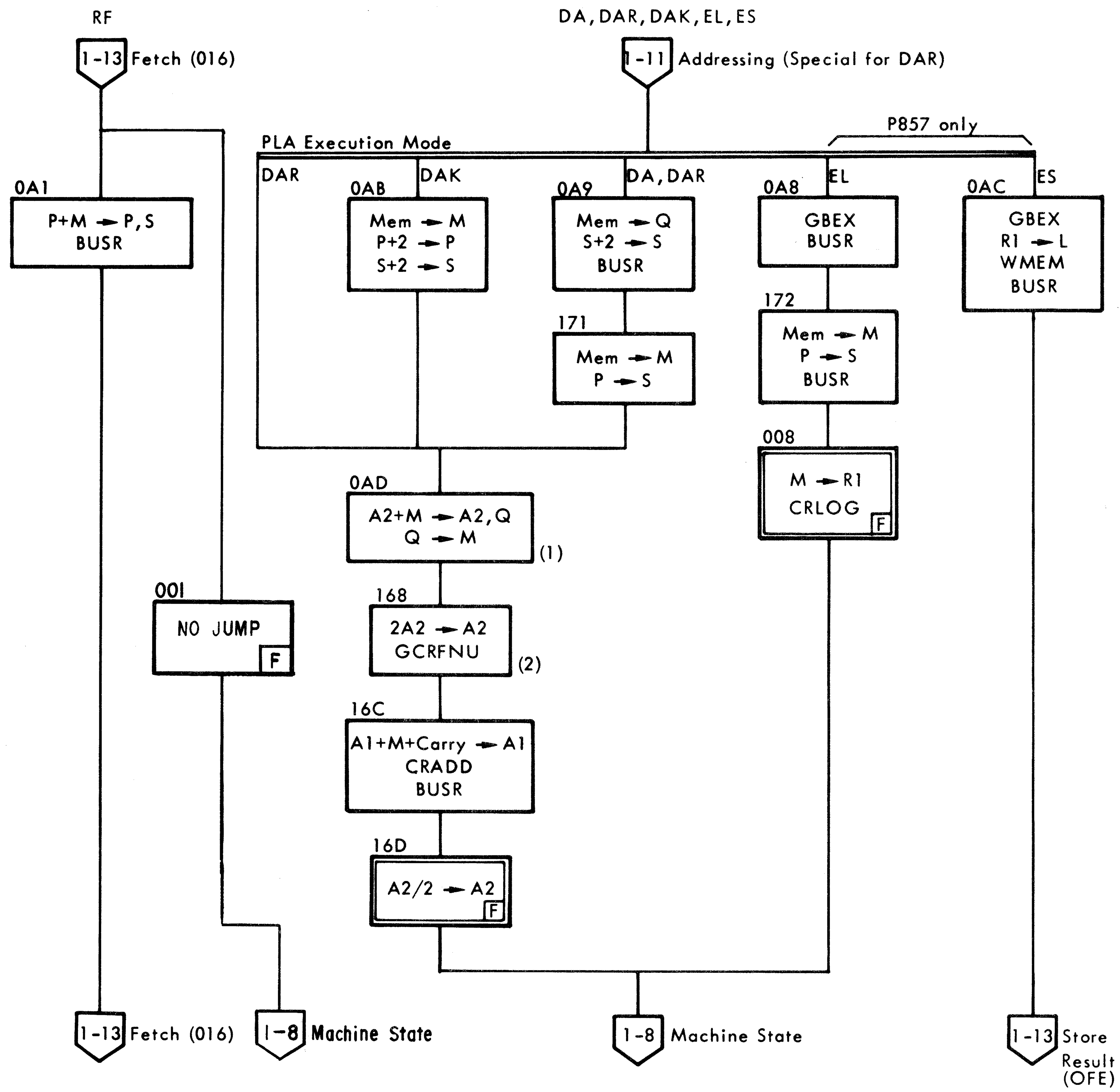
S = Store flag
AD = Address (right shifted)

Figure 1-27 OPC 8, 9 Floating Point:OP, OP Store, Load, Store (FA, FS, FM, FD, FLD, FST) P857 only



(1) This location is necessary to reset BSYCPU - M → P, S needs the transfer P → M in 09E

Figure 1-28 OPC 9 Floating Point Convertions (FFL, FFX), P857 only



(1) Carry in Q00

(2) FNU correctly positioned on 15 bits of the second word.

Figure 1-29 OPC10 (RF, DA, DAR, DAK, EL, ES)

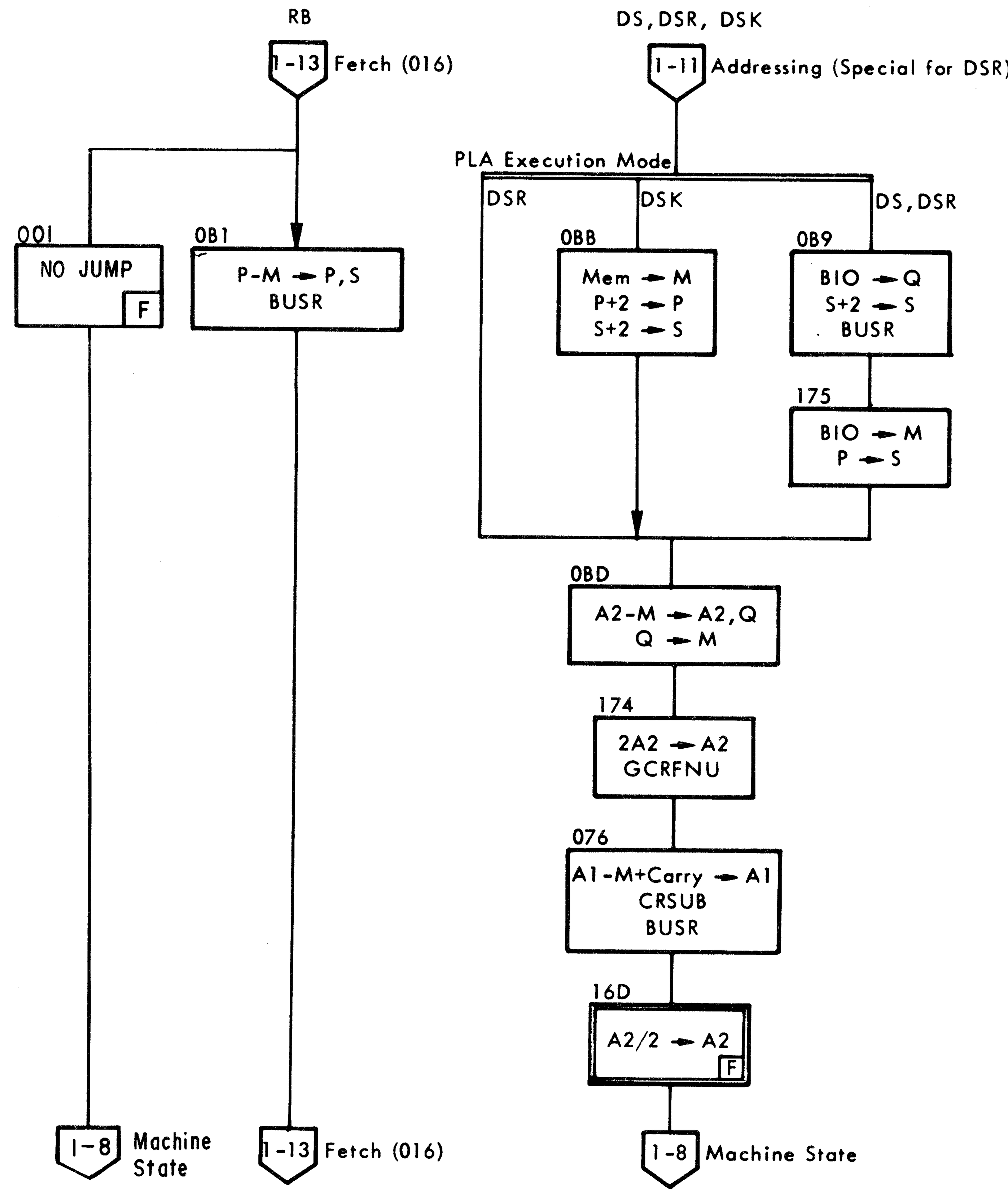


Figure 1-30 OPC 11 (RB, DS)

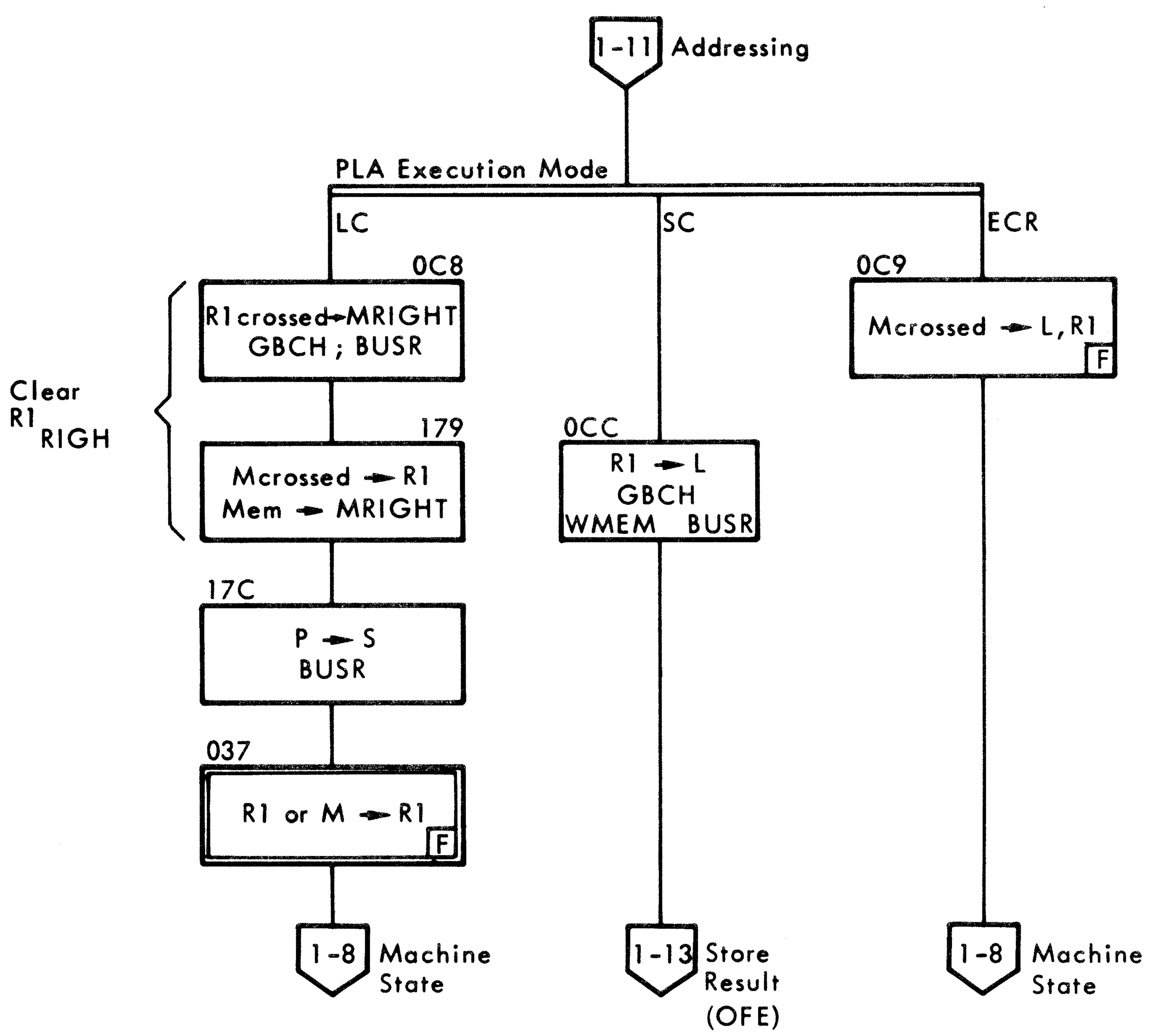


Figure 1-31 OPC 12 (LC, SC, FCR)

1/9/75

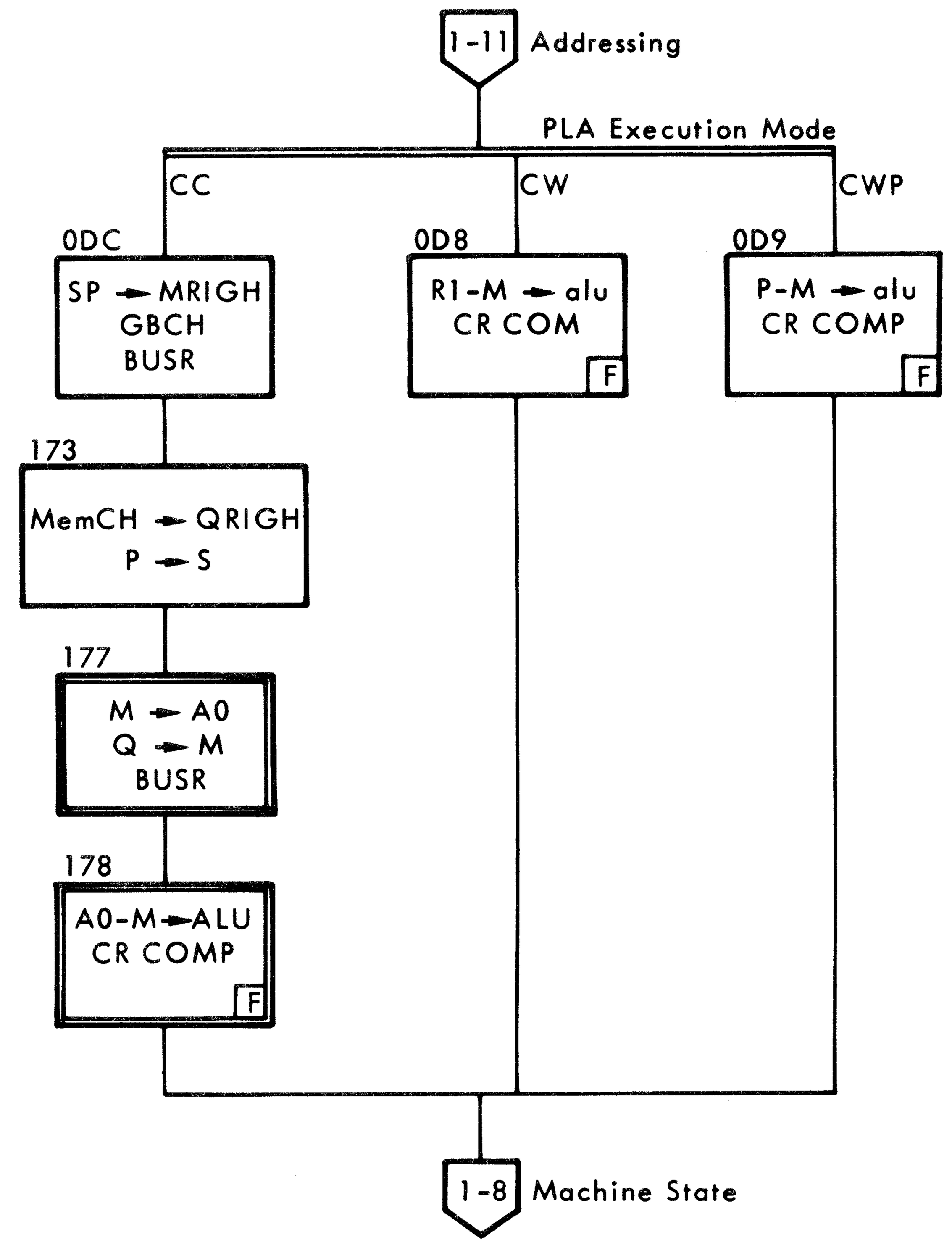


Figure 1-32 OPC 13 (CC, CW)

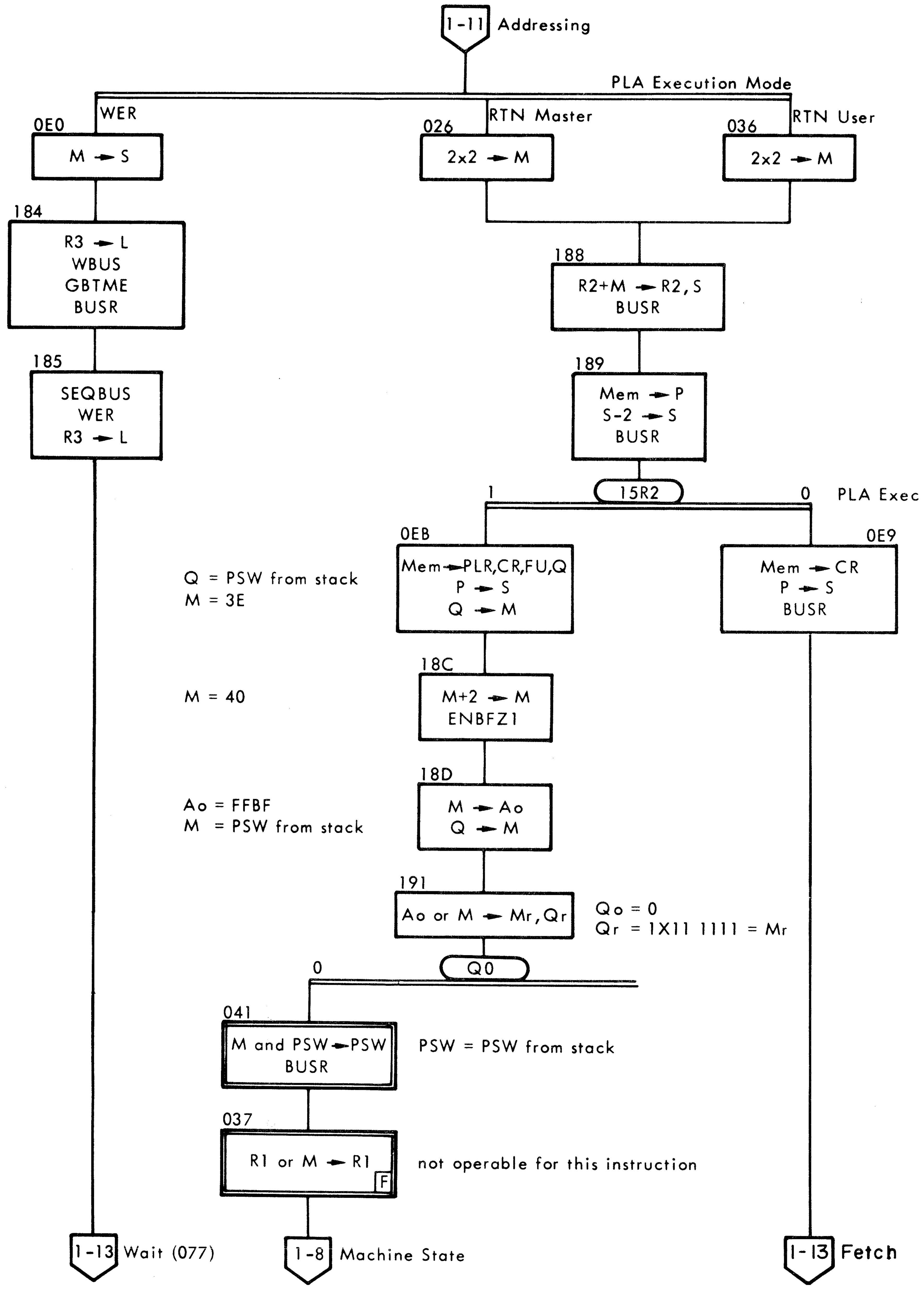
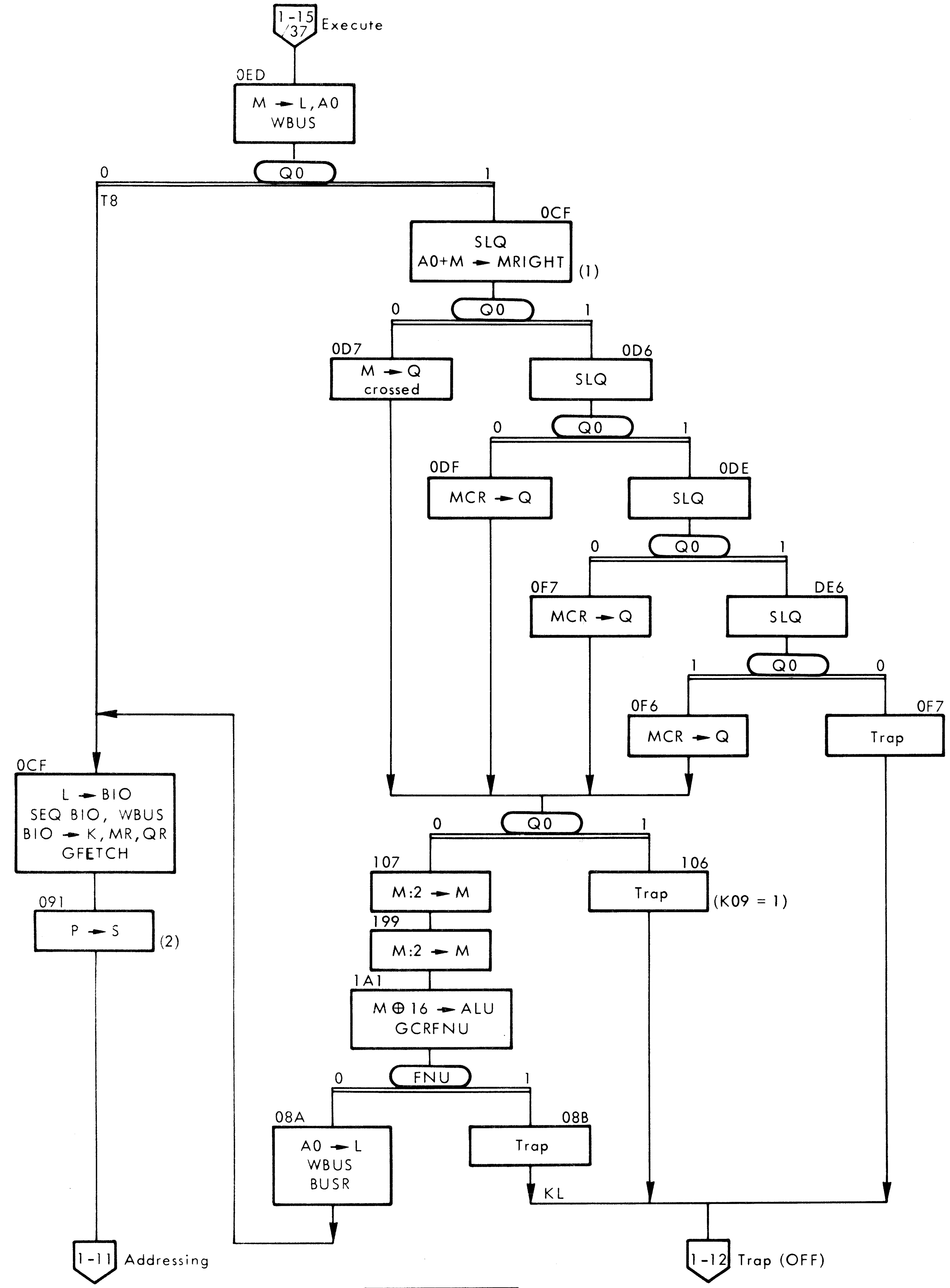


Figure 1-33 OPC 14 WER and RTN



(1) Contents of M are

0	k9	k15	0

(2) This position permits the BSYZO which had been inhibited in 0CF by SERBIO so that the BIO's are not destroyed on T6. S is loaded because addressing routines type "S" (without P→S) were used.

Figure 1-34 OPC 14 Execute (EX)

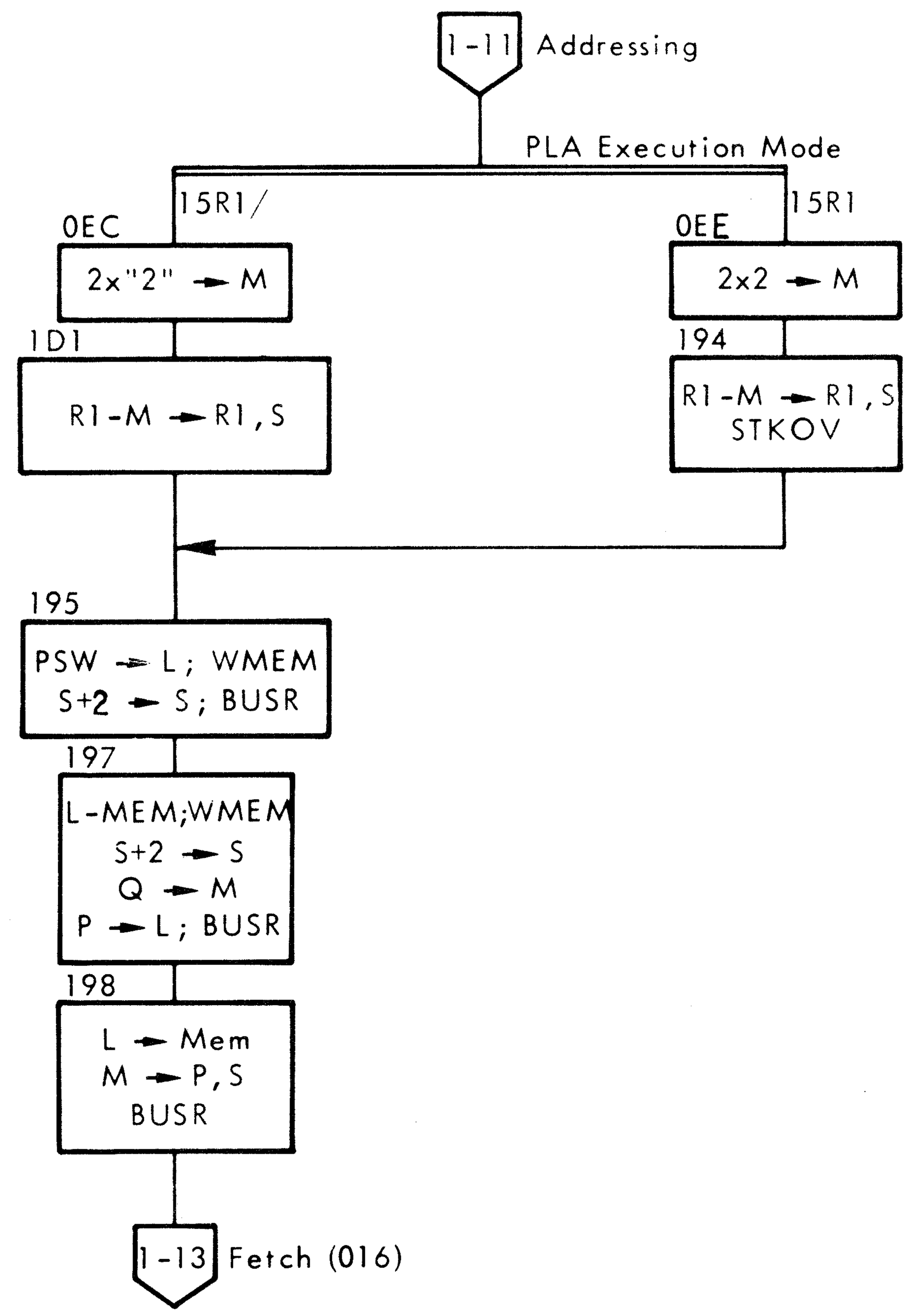


Figure 1-35 OPC 14 Call Function (CF)

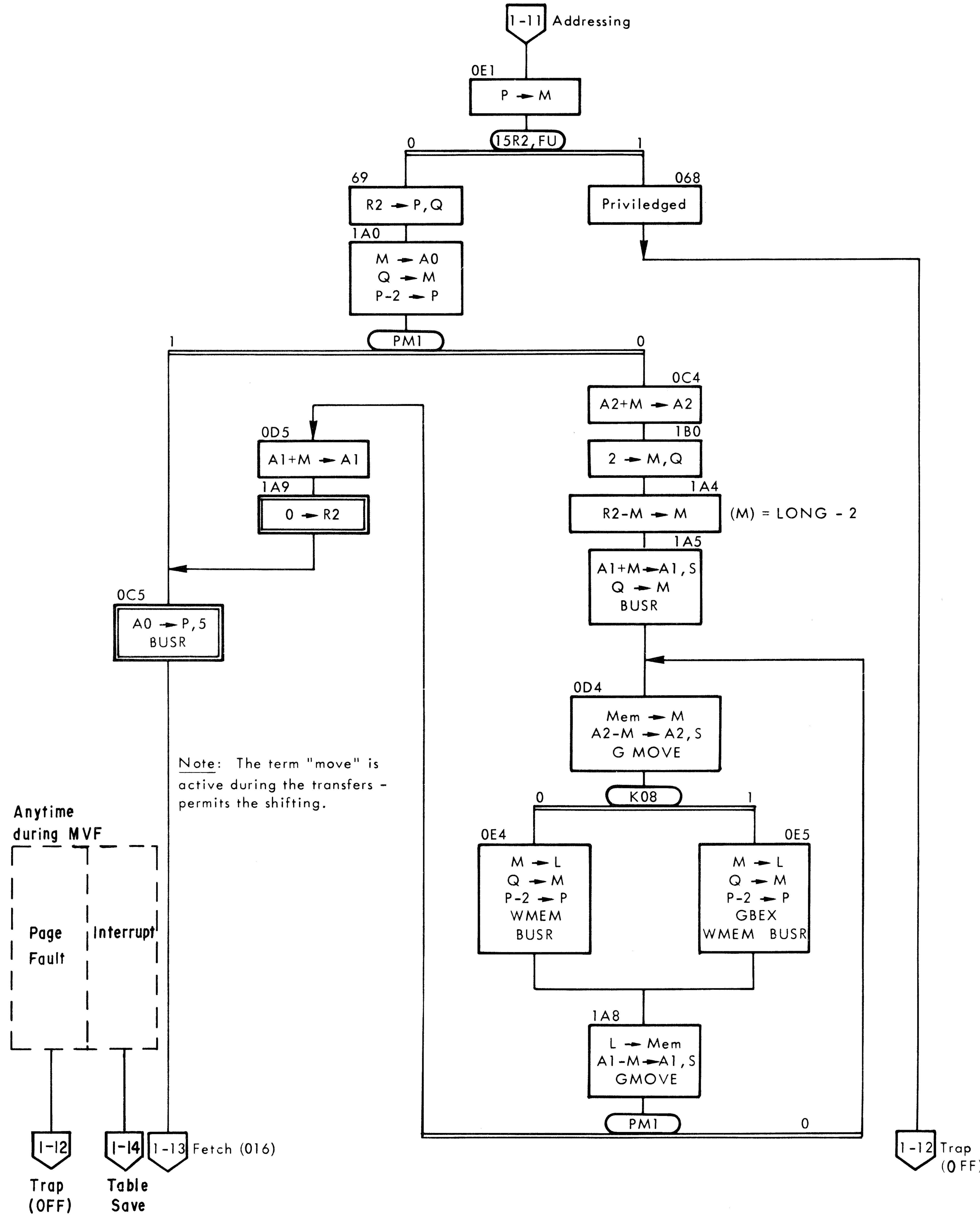


Figure 1-36 OPC 14 MVF and MVSU (P857 only)

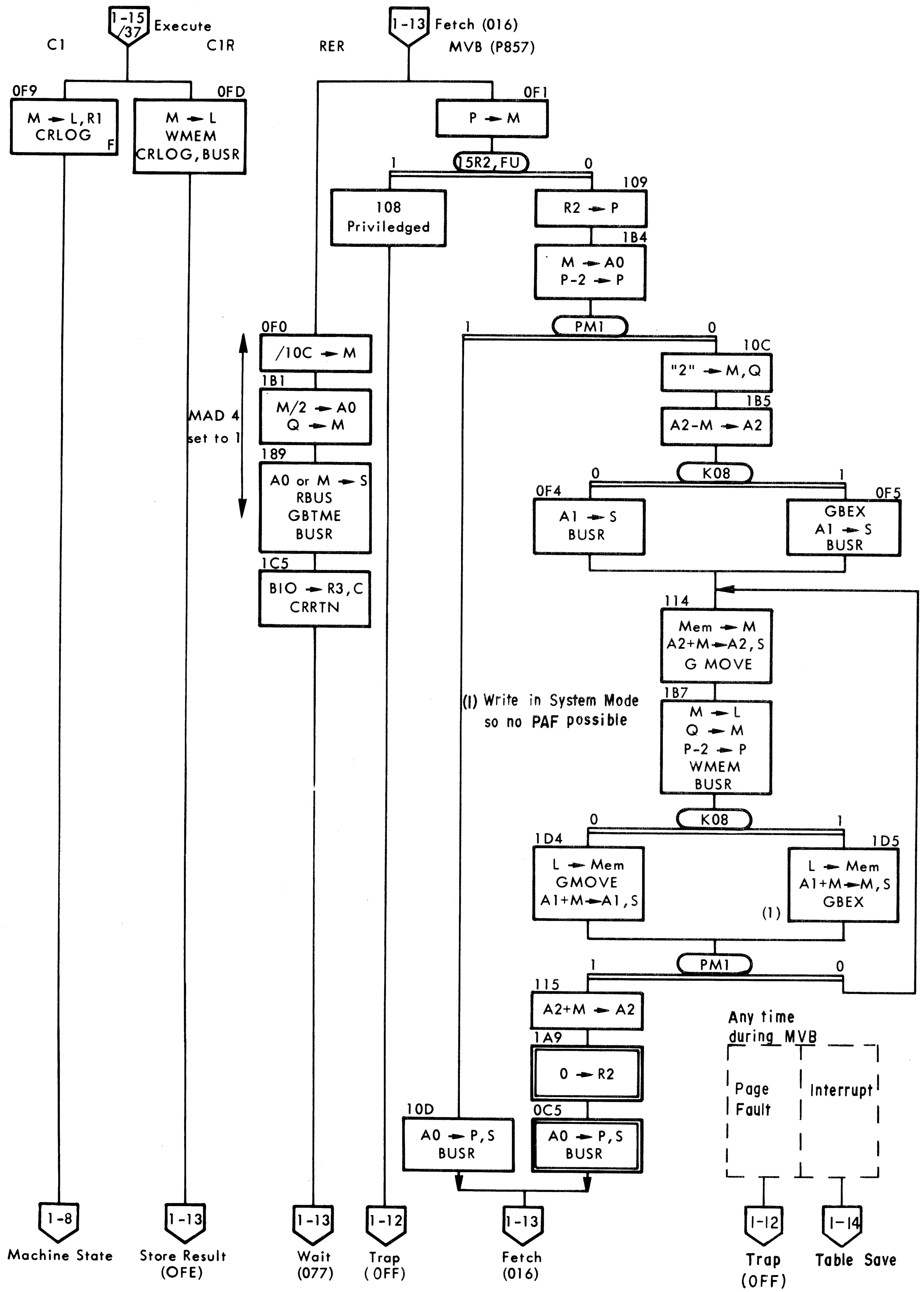


Figure 1-37 OPC 15 (CI, RER, MVB, MVUS)

SECTION II

CPU LOGIC

2.1 GENERAL

The CPU logic is divided into functional sections shown in the block diagram, Figure 2-1. Detailed logic diagrams and a list of all CPU signals are located at the end of this section. A guide to the integrated circuits is provided in Section IV. The CPU logic description is given in the following paragraphs:

2.4	GP Bus Lines	2.53	Data Handling Logic
2.10	Bus Controller	2.97	Interrupt Logic
2.24	Microprogram Control	2.104	Sequensor
2.48	Instruction Word Logic	2.116	Power-Fail/Restart/Resets

2.2 Signal Mnemonics

The signal names used are mnemonics for the function of the signals. The following letters have a special significance when used with the mnemonics:

- F indicates a flip-flop output.
- N suffix indicates an active-low signal (0V=1,5V=0).
- Y indicates the copying of information.
- Z0,Z1 is used after a flip-flop mnemonic to indicate (respectively) setting to 0 or setting to 1 of the flip-flop.

A complete list of signals is provided at the end of this section (Table 2-14).

2.3 Logic Conventions

In the following logic descriptions, the suffix N indicates only the electrical level of the named signal (NAMEN is 0v when active and NAME is +5v when active). The logic state of the signal is indicated either by saying "active NAME," "inactive NAME," "reset NAME," etc., or by the bar (NAME, $\overline{\text{NAME}}$).

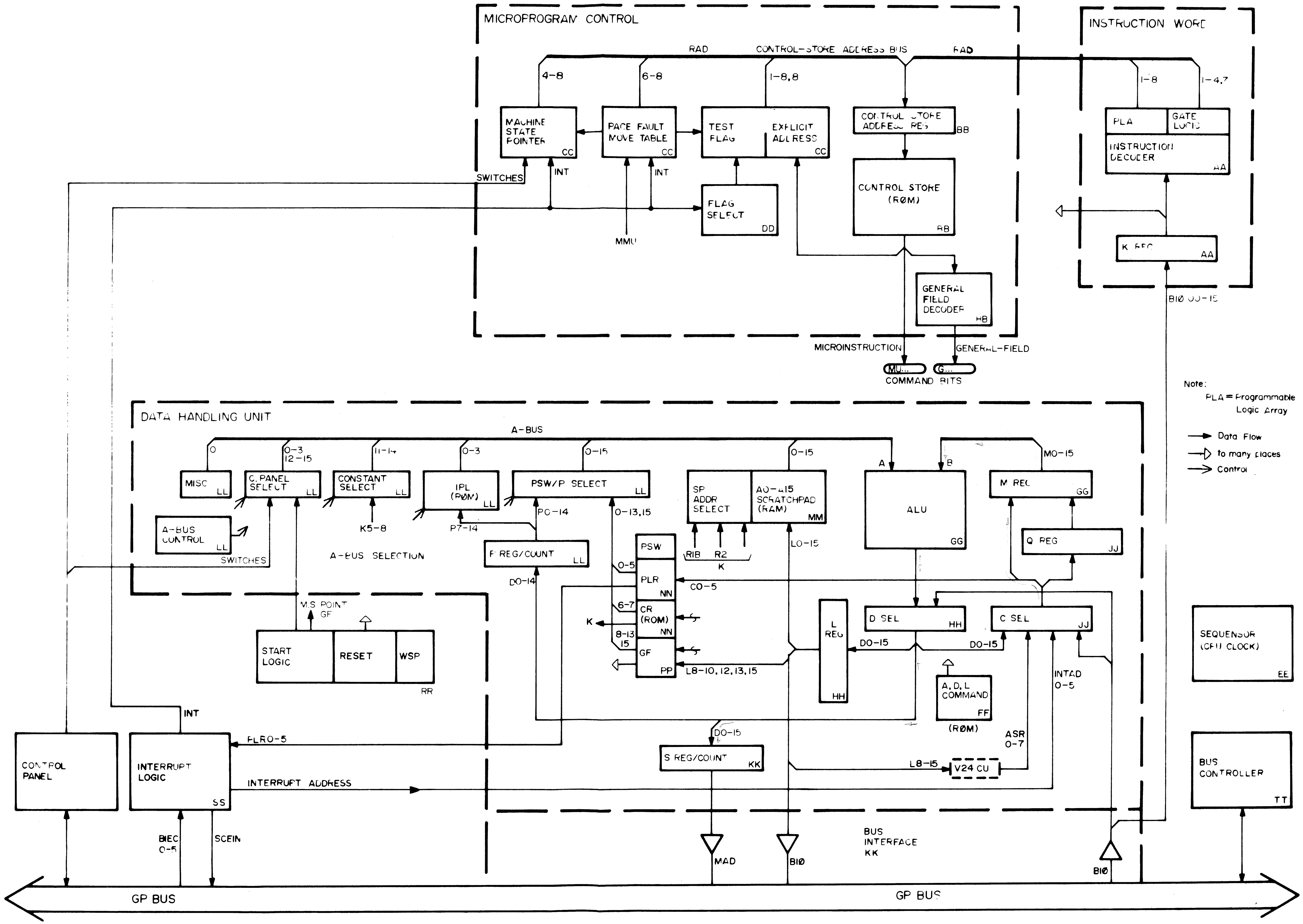
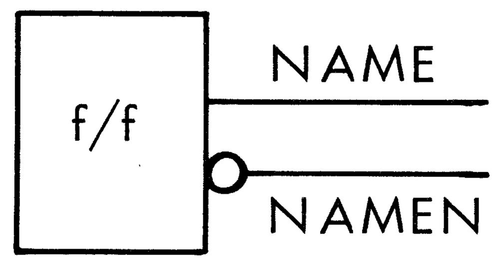


Figure 2-1 P856/857 CPU Block Diagram

Where command bits are used to form a control code, their state may be shown by the equation sign: μBIT0,1,2 = 001. A logic signal is in its True logic state when at the level specified by its name: NAME is True when high (+5v) and NAMEN is True when low (0v). Some examples are:



Active high (+5v) when f/f is set)

Active low (0v when f/f is set)

Signals in True logic state	Signals in False logic state
NAME, NAMEN	$\overline{\text{NAME}}$, $\overline{\text{NAMEN}}$
"active NAME," "active NAMEN"	"inactive NAME," "inactive NAMEN"
μBIT2 = 1, μBIT2N = 1	μBIT2 = 0, μBIT2N = 0

2.4 GP BUS

The 57-line General Purpose Bus comprises the following signals:

Control	BUSRN SPYC OKO/OKI MSN BSYN
Timing	TMRN TMPN TMEN TRMN TPMN
Data	BIO00-15N
Address	MAD00-15,64,128
Misc	ACN BIEC0-5 CHA CLEARN PWFN RSLN SCEIN WRITE

2.5 Control Lines

- BUSRN - Bus Request, from master to CPU (bus controller); remains active (low) while master is requesting control of the Bus.
- SPYC - Scan Priority Chain, CPU (bus controller) response to BUSRN; it warns the masters to prepare for the selection of a new master of the bus. SPYC is active low.
- OKO - from CPU (bus controller) to the master with the highest bus priority. If this master does not require the bus, it passes OKO on to the next lower-priority master. If a master requires the bus, it blocks OKO and generates MSN. (The order of priority of the masters is determined by hard wiring at installation time.)
- OKI - is the signal name of OKO at the input of each master.
- MSN - is the Master Selected, generated by the master which accepted OKO to take control of the Bus.
- BSYN - Bus Busy, from the CPU or any other master which has control of the Bus, when an exchange is in progress.

2.6 Timing Signals

- TMRN - from master to ~~register or~~ memory; validates the BIO and MAD lines and controls the exchange timing.
- TMPN - from master to peripheral CU; initializes the CU exchange and validates the CU address on the Bus.
- TMEN - from master to external register; validates the register addresses and the data, and controls the exchange timing.
- TRMN - from register or memory to master, in response to TMEN or TMRN when the slave is ready for the transfer; also terminates the exchange.
- TPMN - from peripheral CU to master, in response to TMPN to validate the response; also terminates the exchange.

2.7 Data Lines

- BIO00-15 handle both input and output data between all system elements on the GP Bus.

2.8 Address Lines

- MAD128,64,00-15 are the Bus address lines; 128 is the most-significant bit and 15 is the least-significant bit. The meaning of the MAD lines depends on the type of exchange, as follows:

Master←→Memory (TMRN time)

MAD:	128,64,00-14	15

Address from master to memory

0 = left character
1 = right character

Master←→CU (TMPN time)

MAD:	128,64,00-02	03	04	05-07	08	09	10-15

1 = last exchange

0 = to CU
1 = to master

0 = data transfer
1 = command or status transfer

Special Functions

Master←→External Register (TMEN time)

MAD:	128,64,00-03	04	05-07	08 - 15

0 = read
1 = write

register address

Note: Shaded parts not sent on Bus.

2.9 Miscellaneous Lines

- ACN - Accept, from the addressed CU. The CU accepts the command from the master.
- SCEIN — Scan External Interrupts from the CPU: A 2µsec signal sent at the end of every instruction, if the previous SCEIN is finished. After each SCEIN, the CPU compares the BIEC0-5 code with the level of the running program to determine if a program interrupt is required.
- BIEC0-5 - Bus Interrupt Encoded is the priority level of the highest-priority interrupt request pending from an external rack, at SCEIN time.
- CHA - Character, from master to memory indicates exchange is by character (CHA=1) or by word (CHA=0).
- CLEARN - General clear (reset) signal from CPU to all system elements, for initialisation.
- PWFN - Power Failure from the CPU sequences power off at power-failure and switching-off time without losing data, and controls power-on/auto-restart.
- RSLN - Reset Line from the CPU is the power restoration/validation signal.
- WRITE - From master to memory indicates exchange is write (line active) or read (line inactive).

2.10 BUS CONTROLLER

The Bus Controller logic (Figure 2-8TT) regulates access of all system masters to the GP Bus. When the Bus is free, the Controller scans the masters for a Bus-access request.

2.11 Bus Access

Any master requiring Bus access (Figure 2-2) may generate the Bus Request signal BUSRN if MSN is inactive, indicating no other master is being selected. The CPU Bus Controller logic receives BUSRN and, in response, generates Scan Priority Chain (SPYC), which is active low. If the Power Failure signal PWFN is inactive, BUSR sets the OKVAL flip-flop, and OKO is put onto the GP Bus along with SPYC. If PWFN is active, the CPU takes control of the Bus and does not generate OKO.

2.12 The highest-priority master (first master in the series) with a Bus Request blocks OKI and generates Master Selected (MSN). This highest-priority master which has just been selected may not have been the first to generate BUSRN. Active MSN blocks further generation of BUSRN by any master.

2.13 The Bus may be busy with an exchange (BSYN) while a new master is being selected. Once BSYN drops, the newly-selected master takes control of the Bus by generating BSYN and dropping MSN. With MSN inactive, any other master requiring Bus access may generate BUSRN and initiate a new selection sequence.

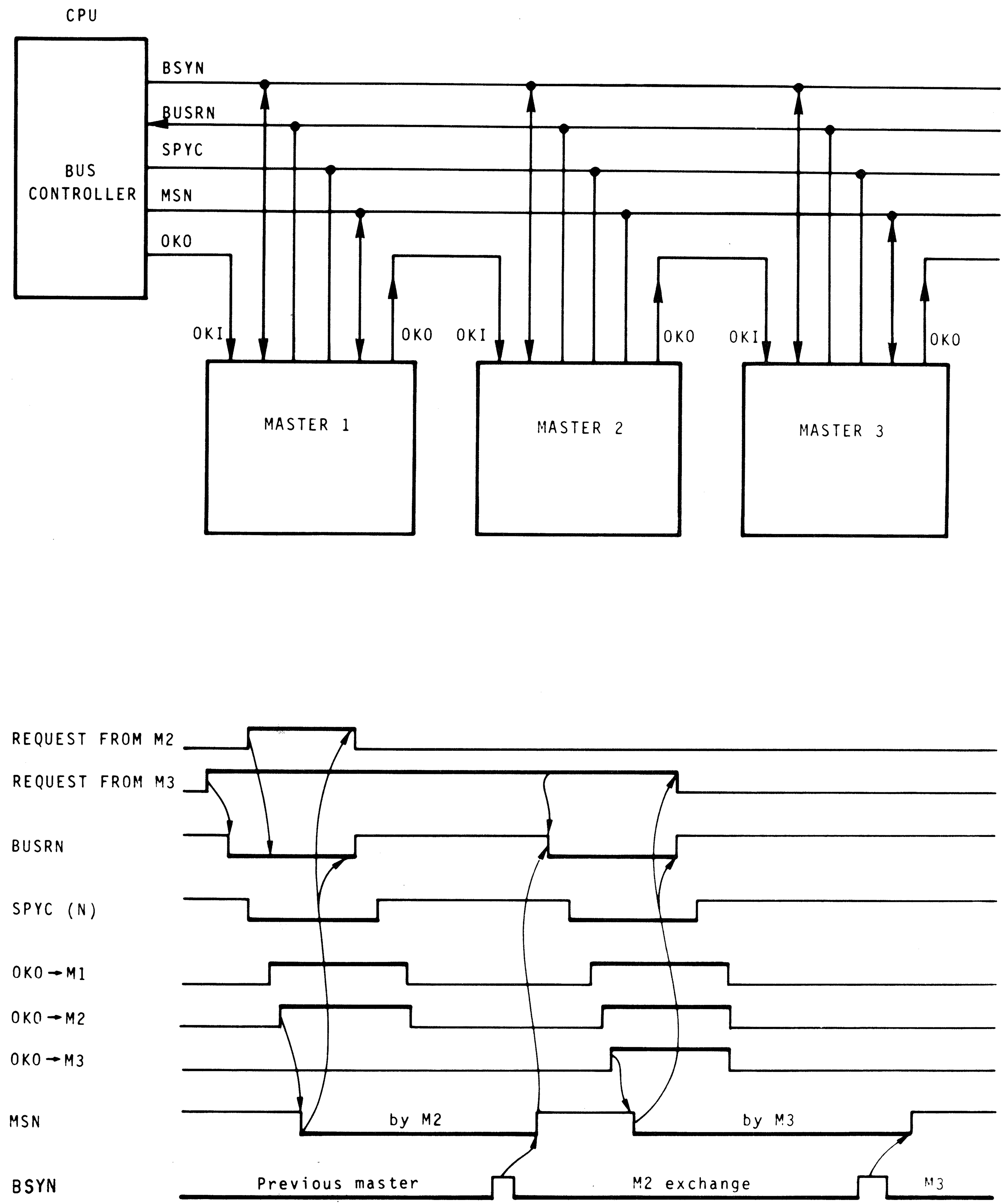


Figure 2-2 Bus Control Block/Timing Diagram

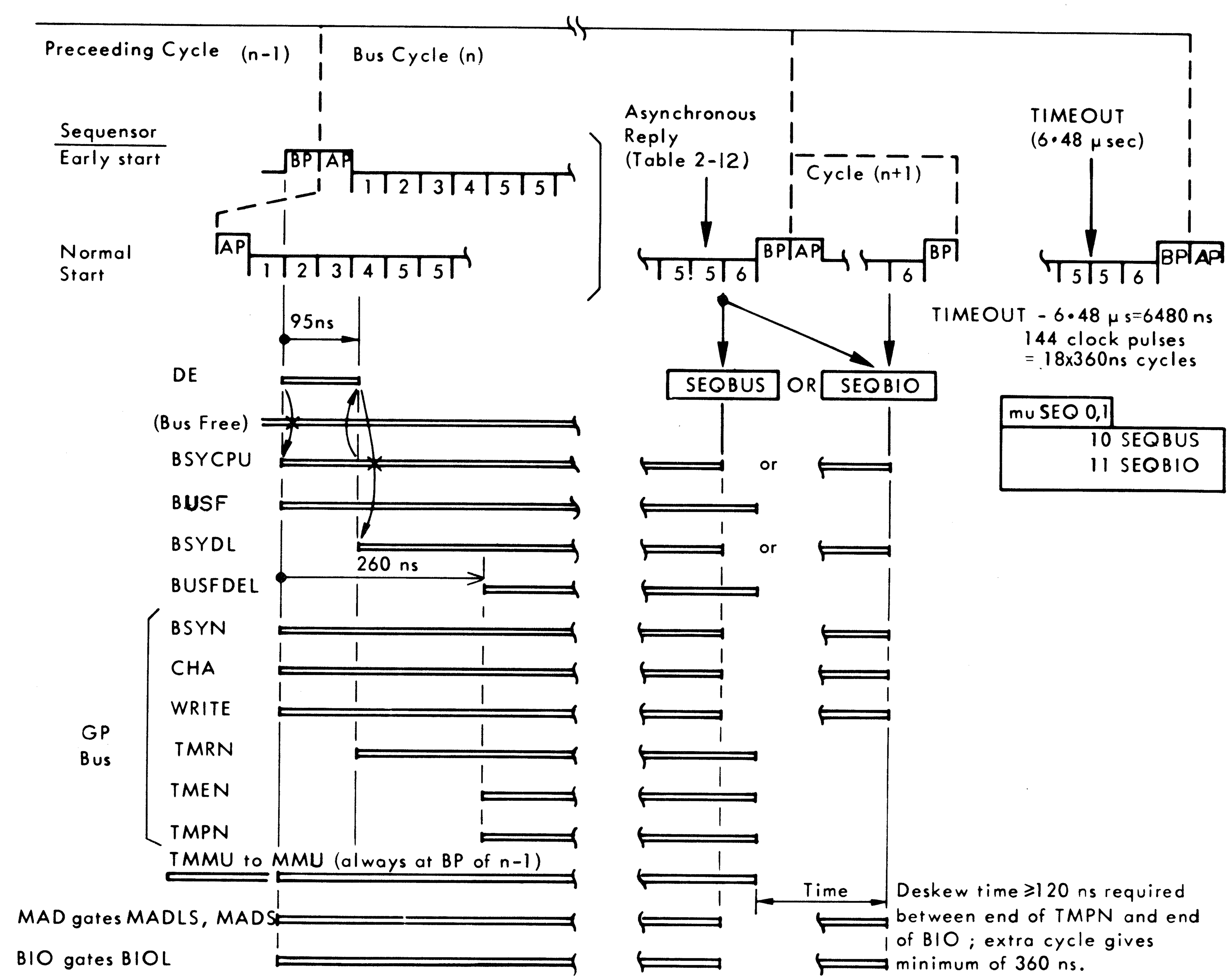


Figure 2-3 CPU Bus Control Timing

2.14 CPU Bus Control

The CPU has lowest priority for Bus access (except for power failure). The Bus-free condition, no other master requesting or using the Bus, is $\overline{BUSRN}$.$\overline{MSN}$.$\overline{BSYN}$.$\overline{TRMN}$.$\overline{TPMN}$. Any of these signals will inhibit BSYZ1N and prevent setting BSYCPU. The CPU takes control of the Bus under microprogram control when μSEQ0,1 = 1x (Sequensor cycles SEQBUS [10] or SEQBIO [11]).

2.15 (Figures 2-3, 2-8TT). If a specific microcycle (n-1) is always followed by a Bus microcycle (n), the Bus access is started early,at BP of the preceeding microcycle (n-1). This reduces the Bus cycle by 135ns if the preceeding cycle is a Sequensor logic cycle; additional time is saved following other sequensor cycles. The early Bus access is initiated by microinstruction-bit μBSR and BP setting the DE flip-flop (if there is no MMU page fault, MFAULTN/PAFN). When not started early, the Bus access is initiated at T2 of the Sequensor Bus cycle: T2 and μSEQ0 set the DE flip-flop if the early-start signal MUBURSRFN is inactive.

2.16 The DE output sets BSYCPU with signal BSYZ1N, if the Bus is free. BSYCPU in turn sets the BUSF flip-flop. If BSYCPU is set, DE resets itself after a 95ns delay. When DE drops, the BSYDL signal is activated; BSYDL and μSEQ0 = 1 select the Sequensor operating cycles for Bus transfers (Table 2-12). BSYCPU also provides the following Bus gating:

- BYSN activated onto the Bus.
- Exchange-parameters CHA and WRITE gated onto the Bus: the CHA and WRITE data are always loaded into the Bus Controller from the microprogram control store at BP of the preceeding cycle (n-1).
- Bus timing signal TMRN, TMEN, or TMPN gated onto the Bus: the conditions for these signals (Table 2-1) are set into the Bus Controller at BP of the preceeding cycle.
- MADS, MADLS gated to the Bus interface logic for gating the MAD lines onto the Bus.
- BIOL gated to the Bus interface logic for gating the BIO lines onto the Bus.

Table 2-1 Bus Timing Signals

Operation	Microinst	Stored at BP	GP Bus Signal	Reply
CPU Exchange Ext. Reg/CU/Memory				
External Register	GBTMEN	TMEF	TMEN	TRMN
CU	GBTMPN	TMPF	TMPN	TPMN
Memory	μTMRN	TMRF	TMRN	TRMN
	GBCP + ($\overline{FU}$ · $\overline{GBEX}$) + MMUABS			
Mem. via MMU translation	μTMRN	TMMU	TMRN from MMU	TRMN or MFAULTN
	$\overline{GBCP}$·(FU + GBEX)·$\overline{MMUABS}$			
MMU Operations				
Table Load	GBTMFN	TMMN		DONEMN
Table Store	GBOMN .GCRFNUN	BOMFN		DONEMN
Load P'	GFETCH			
FPP Operations				
Load K'	GFETCH			
Store Operand	GBOFN. GCRFNUN	BOFFN		
Load Operand	GBTMMN	TMFN		DONEFN
Process	GFLOT	FLOACT		DONEFN

P857 only

2.17 Bus Addressing (MAD Lines)

The control of the Bus MAD lines is determined by the MADLCPUN and MADCPUN signals which control MADLS and MADS to the MAD output gates. The gating signal to both MADLS and MADS is BSYCPU (preceeding paragraph). The conditioning logic for setting MADLCPUN and MADCPUN produces the Boolean equations:

for MADLS: $(\overline{GBCP}+CPBABS).\underbrace{(GBCP+(\overline{GBEX}.\overline{FU})+MMUABS)}_{\text{P857 only}}$ = within page

for MADS : $(\overline{GBCP}+CPBABS)$ = page address

- If GBCP: the control panel validates the MAD lines; the CPU has control only if control-panel B-half is absent (CPBABS).

- If $\overline{GBCP}$ (P856): the CPU validates MAD128,64,00-15 with MADLS and MADS.
- If $\overline{GBCP}$ (P857): the CPU validates MAD128,64,00-03 with MADS; the MMU validates the MAD04-15 lines if GBEX or FU, and the MMU option is present; the CPU has control only if the MMU is absent or if GBEX and FU.

The following diagram is a resume of the MAD line validation control for memory addressing:

MAD	128,64,00-03	04 15	
BSYCPU.$\overline{GBCP}$.$\overline{GBEX}$.$\overline{FU}$	CPU		
BSYCPU.GBCP.$\overline{CPBABS}$	CONTROL PANEL		
BSYCPU.GBCP.CPBABS	CPU		
BSYCPU.$\overline{GBCP}$.(GBEX+FU).$\overline{MMUABS}$	CPU	MMU	P857 only
BSYCPU.$\overline{GBCP}$.(GBEX+FU).MMUABS	CPU		

2.18 For memory addressing under CPU control, the S register provides the memory address to the Bus MAD lines, according to the preceeding diagram. For CU or External Register addressing, the CPU logic provides the address via the D-selector and the S-register to the MAD lines.

2.19 Bus Data (BIO Lines)

The Bus data from the CPU are gated onto the BIO lines by the BIOL signal from the Bus Controller. BIOL is generated by microinstruction bit μBIOL, which is loaded at BP of the preceeding microcycle (n-1) as BIOELN.

2.20 Bus Timing Signals

The Bus timing signals (Table 2-1) TMRN, TMEN, TMPN are generated by the master that has control of the Bus. For CPU Bus control, one of these signals is generated by the Bus Controller during the Sequensor Bus cycle (SEQBUS or SEQBIO). For MMU or FPP operation (P857), the Bus Controller sends a special timing signal directly to the MMU or FPP, and this external unit generates the Bus timing signal. The CPU waits at Sequensor clock-pulse T5 until the necessary reply is received to end the Bus cycle. The reply (Table 2-5) is either the Bus response TRMN, TPMN to the CPU or the special reply signal directly from the MMU or FPP to the CPU.

2.21 When the required ending condition is satisfied, including reply received, the Sequensor steps to clock T6. For SEQBUS (μSEQ0,$\overline{1}$) operations, BSYCPU is reset by the T6 and μ SEQ1 = 0. The inactive BSYCPU ends the Bus control and data lines BSYN, CHA, WRITE, MAD, and BIO. The timing signals end 45ns later, being reset directly by the BP clock. For SEQBIO (μSEQ0,1) operations, the BIO lines must be prolonged at least 120ns to satisfy CIO timing requirements. In this case, μSEQ1 = 1 and BSYCPU is not reset. The Bus control and data lines, including BIO, are continued into the next Sequensor cycle (n+1) while the timing signal TMPN is reset normally by BP at the end of the BUS cycle. The microcycle after the Bus cycle (n+1) will have μSEQ1 = 0 so that BSYCPU is reset at T6 of that cycle.

2.22 Timeout

A timeout circuit (Figure 2-8TT) is used to release the Bus and set the condition register to 3 (CR = 11_2) if the addressed slave doesn't respond within 6.48usec. The basic clock for timeout is derived from the 45ns OSC signal (Figure 2-8EE) divided-by-2 by the OSC90 (90ns), and then divided-by-9 by the TC810 counter. The counter initially (following RSLBN -- 0) counts from 0 to 15; each TC810 output pulse (at count 15) then resets the counter to 7 so that the counting cycles continuously from 7 through 15. The resultant TC810 pulse (810ns period) drives the Timeout counter (Figure 2-8TT).

2.23 Timeout is held reset by the four inactive Bus-Control signals to the MR input. At the beginning of a bus cycle, (Figure 2-3), BSYCPUN or any master command goes active and the counter begins counting the TC810 input pulses. During the bus cycle, one of the TM... timing signals will also be activated. When the addressed slave responds, the bus cycle is terminated and the timing signal TM... and signal BSYCPUN are deactivated, and the Timeout counter is again reset. If there is no response by the time eight TC810 pulses have been counted

(6480ns), the counter produces TIMEOUT. TIMEOUT activates the required response signal on the Bus (TPMN or TRMN) to stop the exchange and unblock the system, generates sequensor pulse T6 to end the bus cycle, and sets CR bit 0.

2.24 MICROPROGRAM CONTROL

Each CPU instruction or operating sequence (paragraph 1.67) is controlled by a selection of microinstruction control words stored in a read only memory (Control ROM). The different microinstruction control words are accessed by the instructions or sequences (at each AP clock pulse time). The words are addressed partly by the instruction content (K register) and partly by the currently-accessed control word. Various CPU or system conditions may also modify or change the selection of the next microinstruction control word.

2.25 The microinstruction control words are used directly by the CPU logic as command bits (μ...). Each 48-bit microinstruction is divided into 14 command fields (Table 2-2). The five-bit general-field section of the control word is decoded to produce 28 general field command bits for further microinstruction control. Both the micro-command bits (μ...) and the general-field command bits (G...), listed in Table 2-3, are used by the CPU logic as direct command bits. These command bits are shown on the logic diagram (Figure 2-8BB) with their destinations listed; they are shown throughout the logic as command inputs, and identified by being enclosed in boxes.

Table 2-2 Microinstruction Command Bits

	Field	Bit	CPU section controlled by the command field
Next Word Control	μSNA 0-1	0-1	Select Source of Next Address
	μNA 0-8N	2-10	Next Address, Explicit
Data Path Control	μA 0-4	11-15	A-Bus Selector
	μADL 0-4	16-20	ALU,D,L Command Decoder
	μC 0-1	21-22	C Multiplexer
	μMLOAD,MSEL	23-24	M Register
	μQ 0-1	25-26	Q Register
	μS 0-1	27-28	S Register
	μP 0-1	29-30	P Register
	μCT	31	CT Counter: loops, shifts
	μSEQ 0-1	35-36	Sequensor (CPU clock). Bus Controller
	μTMRN,BIOL, WRITE,BUSR	32-34 37	Bus Controller
	μCR 0-2	43-45	Condition Register
	(not used)	46-47	
	μGP 0-4	38-42	GENERAL FIELD SELECTION

2.26 Microinstruction Addressing

The Next-Address mode for the microinstruction word determines which of four types of microprogram addressing is to be used. The mode is selected by bits μSNAO and μSNA1 from the current microinstruction word, as follows:

μ SNA 0 1	Mode	Function
0 0	Explicit	Explicit address is μNA0-8.
0 1	Flag	Flag from execution-pointer tests (selected by μNA6-8) sets RAD8; μNA0-7 to RAD0-7.
1 0	Instruction Word (PLA)	Instruction word (K-reg, decoder) provides address RAD1-8; μNA7 modifies the decoder for fetch or execution.
1 1	Machine-State	The machine-state pointer provides address RAD4-8; μNA0-3 to RAD0-3.
x x	Move Table/Fault: an interrupt or a page fault during a Move Table instruction force a special microinstruction address with bits RAD6-8.	

The Next-Address mode selectes a set of open-collector gates onto the control-store address bus (Figures 2-8AA,CC). This control-store address is loaded into the RA register by the AP clock pulse.

2.27 Explicit Address. The next address is explicit in the current micro-instruction μNA0-8 field.

μSNA0,1	RAD 0	8
0 0	μNA0-8	

The explicit-address gates (Figure 2-8CC) invert the μNA0N-8N onto the RAD0-8 lines. All eight gates (and the flag gate) are blocked (forced high) by NARDT during a Move Table/fault. Other gates are blocked selectively by next-address modes which use some of the μ NA bits.

2.28 Flag. The next address is explicit in the current microinstruction μ NA0-7 field, with the least significant bit being selected by the Flag test. Any one of eight different execution-pointer tests can be performed during Flag mode, as selected by the bits μNA6-8.

μSNA0,1	RAD 0	7	8
0 1	μNA0-7		

μNA6,7,8N	Flag	Use
0 0 0	Q00N	Display, C.Panel, IPL, Trap, End of TL-TS (P857), Execute, DIV correction.
0 0 1	IRN	Move Jump.
0 1 0	PM1	End of ML, MS, Move
0 1 1	K08	Shifts, Move
1 0 0	FNU	C.Panel test, IPL, Wait-C2, Trap in EX-T2.
1 0 1	NORM	End of normalized shifts.
1 1 0	DIV	Remainder correction.
1 1 1	FU15R2	Privileged test; Trap for SLN, SRN; Move.

The Flag mode uses explicit-address gates 0 to 7 (Figure 2-8CC) onto the RAD0-7 lines. The eighth explicit gate is blocked by NAEXPL while the flag gate is enabled by NAFLG. The signal FLAGN sets bit RAD8. FLAGN is selected by a type 74151A eight-input multiplexer (Figure 2-8DD) which is controlled by μNA6N,7N,8N. The eight tests selected for the FLAGN bit are described in the following paragraphs.

2.29 Flag Q00N is used as a general test condition for various commands and instruction.

2.30 Flag IRN is the interrupt request. During a page fault or an interruption of a Move-instruction/execution, flag IR enables the separation of the exit subroutine. If the Move instruction was interrupted, the last exchange was executed. These subroutines save the parameters necessary for resuming the Move execution.

2.31 Flag PM1 is used to detect the last word transfer during a Multiple Load or Store (ML,MS) or Move instruction. PM1 is taken from the un-used (most-significant) bit position of the P counter when it is used as an auxiliary counter. The P counter is normally used as a 14-bit counter for program addresses. When used as an auxiliary counter, PM1 is set by the first decrementing clock after P0-14 has counted down to zero.

2.32 Flag K08 is taken directly from the instruction word to differentiate between different types of shift and Move instructions, as follows:

Shifts: K08-0 = SLA, SLL, SRA, DLA, DRA
K08-1 = SLN, SLC, SRN, DLN, DRN

Move: K08-0 = MVF, MVB
K08-1 = MVSU, MVUS

2.33 Flag FNU stores the zero contents of the ALU. FNU is stored in one bit of a four-bit register (74175). The other three bits of this register have no relationship with FNU.

2.34 Flag NORM defines the end of a shift normalize execution. The end-of-shift signal is GOSH (Figure 2-8DD) which is controlled by a 74151A chip. The inputs to the multiplexer chip are held at 1 or 0 (+5v or 0v) so that the selection and enable inputs provide direct control of GOSH as follows:

$Q15+(L00 \oplus L01)+P09 = \overline{GOSH} = NORM$

- Shift Right Normalize ends when Q15 sets. The active Q15N selects the low 74151A inputs 10-14 to generate a low GOSH. The P counter is used as an auxiliary counter to count 32 shifts (the program address is saved). If P09 sets before Q15, it means the constant to shift is equal to zero and GOSH goes low. L0⊕L1 never sets.
- Shift Left Normalize ends when L00⊕L01 is true. Since Q15N is not used (held high), L00,01 select 74151A inputs 14-17 as an EXclusive-OR, and GOSH is low when L00⊕L01 is true. The P counter is used in the same manner as in Shift Right Normalize.
- Shift without normalizing. The number of shifts to be executed is contained in K11 - K15. This number is loaded in complement into the Scratchpad CT counter. The counter is incremented to 31 in a single microinstruction (CT+1, Repeat), and the shifting is ended when the microinstruction is finished.

2.35 Flag DIV enables the remainder correction according to test DIVA or DIVB, selected by µQ1. Test DIVA (µQ1) is true if the remainder sign is opposite to the dividend sign. Test DIVB (µQ1N) is true if both the remainder and dividend are <0, but the absolute remainder value is greater than the absolute divisor value. The select inputs and data inputs of a 74151A multiplexer chip are used together to generate the FLAGDIV signal when either test is true.

- DIVA: µQ1 FSIGDIV FSIG (input 14)
  FSIGDIV FNU (input 16,17)
- DIVB: µQ1 FSIGDIV FSIG FNUN (input 13)

2.36 Flag FU15R2 tests for System-Mode reserved instructions which can modify the stack pointer contents. When the stack pointer is indicated (R2E15) and User Mode is set (FU), the active FU15R2N signal is inverted to a high FLAGN, and the bit RAD8 = 0 to cause a trap.

2.37 <u>Instruction Word.</u> The next address is decoded directly from the instruction word.

µSNA0,1	RAD 0		8
1 0	0	K-reg decoding	

This is used as the first address of an instruction microprogram. The instruction-word addressing mode operates in one of four sub-modes, selected by the signals PLA0, PLA1 :

PLA0,1	Sub-Mode	Purpose
0 0	Inhibit	Instruction-word outputs held inactive (high).
1 0	Add Master	Fetch operand for types T1-T7; execute for type T8; Trap if illegal.
1 1	Add User	Fetch operand for types T1-T7; execute for type T8; Trap if privileged.
0 1	Execution	Used for first microinstruction word after the fetch-operand routine.

The PLA selection logic is shown on Figure 2-8AA. The sub-mode inhibit is set by the inactive validate signal (PLAVALN high). PLAVALN is held high by μSNA0 = 0 (Explicit mode or Flag mode) or an active PAFN. PLAVALN is activated by μSNA0 = 1 (Instruction-Word or Machine-State modes). The conditioning for PLA is as follows:

	μSNA 0,1	PLAVALN	FU μNA7	PLA 0,1	Sub-Mode
	0 0, 0 1	h		0 0	Inhibit
Explicit Implicit	1 0	L	0 0	1 0	Add Master
	1 1		1 0	1 1	Add User
	.RADETPLN		- 1	0 1	Execution

Microprogram bit μNA7 selects a fetch sub-mode (Add Master or User) or the execution sub-mode. If fetch is selected, the FU signal indicates either User mode (FU = 1) or Master mode (FU = 0). For all three of the active sub-modes, the instruction-word addressing may be explicit (μSNA1 = 0) or implicit (μSNA1 = 1). The explicit addressing is used for instruction execution and the execution pointers. The implicit addressing is part of the machine-state-pointer control of the microprogram.

2.38 Instruction-word addressing of the microinstruction store is controlled by the instruction-word decoder (Figure 2-8AA). The decoding is done mainly by a programmable logic array (PLA) which provides address codes for 96 different input combinations. In many cases, the output address codes are the same for various different input combinations. The PLA decoding is shown in Table 2-6, Instruction Decoder. Some instruction-word decoding is done by logic gates operating in parallel with the PLA.

2.39 The external instruction-word logic gates are used for an ineffective branch (NOJUMP) or, with P857, a floating-point instruction without an attached FPP (NOFLO). For both instructions, PLA0 is high, and the external gates for RAD1-4 are enabled; also, either NOFLON or NOJUMPN generates NACND which activates the four gates to force RAD1 through 4 low. For No Jump, active NOJUMPN generates NACNDAD; RAD7 is selected by DWIN (double-word) and address /001 or /002 is selected. For No FPP, inactive NOJUMPN blocks NACNDAD; RAD7 is forced high and address /007 is selected. Bit 8 is PLA selected.

		0	1	2	3	4	5	6	7	8	
No Jump	RAD	0	**0**	**0**	**0**	**0**	0	0	0	1	/001
									1	0	/002
No FFP	RAD	0	**0**	**0**	**0**	**0**	0	0	**1**	1	/003

2.40 Machine-State Pointer. The Next Address is determined by machine-state pointers, such as RUNF and control-panel switches. (See also Figure 1-9 for general flow). The machine-state pointer logic is shown on Figure 2-8CC. The instruction-word decoder (Figure 2-8AA) is also used for one sub-mode of the machine-state pointer microprogram control. There are four sub-modes of machine-state pointer address selection, as follows:

				0	1	2	3	4	5	6	7	8	
Instruction			RAD	0	1							8	
				0	PLA Decoder								
Interrupt	RUNF.IR.$\overline{\text{PUP}}$.		RAD	0	1	2	3	4	5	6	7	8	
(RUNF.PUP		KRY		0	0	0	0	1	0	1	0	1	(/015)
=RADET6)		$\overline{\text{KRY}}$		0	0	0	0	1	0	1	0	0	(/014)
C.Panel	RUNF.PUP		RAD	0	1	2	3	4	5	6	7	8	
				0	0	0	0	1	0	0	0	0	(/010)
Program End	$\overline{\text{RUNF}}$.$\overline{\text{IR}}$.$\overline{\text{PUP}}$.		RAD	0	1	2	3	4	5	6	7	8	
(RUNF.μNA8		KRY		0	0	0	0	0	0	1	0	1	(/005)
=RADET6)		$\overline{\text{KRY}}$		0	0	0	0	0	0	1	0	0	(/004)

In all cases, the control-panel TEST key will set RAD bit 0 to 1, via ETATEST and VALNA0 on Figure 2-8CC. The Instruction sub-mode generates the signal RADETPLN (Figure 2-8CC) which activates the instruction-word decoder PLA via signal PLAVALN on sheet AA. Refer to paragraph 2.37. The Interrupt sub-mode selects address /014 or /015 for the interrupt routines. The Control-Panel sub-mode selects address /010 if any of the control-panel command signals are active (PUP signal active). The control-panel command switches operate through the A-bus of the Data Handling logic. The Program-End sub-mode selects a branch to address /004 or /005.

2.41 Move Table/Fault (P857). An interrupt or a page fault during a Move Table instruction force the RAD bits to a special control-store address, as follows:

	Address	RAD	0	1	2	3	4	5	6	7	8
Interrupt. MVB read	1F8		1	1	1	1	1	1	0	0	0
MVB write	1F9		1	1	1	1	1	1	0	0	1
MVF read	1FA		1	1	1	1	1	1	0	1	0
MVF write	1FB		1	1	1	1	1	1	0	1	1
Page Fault	1FF		1	1	1	1	1	1	1	1	1

The move table/fault logic is shown on Figure 2-8CC. Either the page fault (PAFN) or the interrupt during Move Table (MOVEIRN) cause the signal NARDT. NARDT blocks all the control-store addressing gates on the machine-state-pointer, explicit-address, and test-flag logic, which puts the address bits high. The move table/fault logic then forces bits RAD6-8 low according to its control inputs to obtain the address code shown in the above table.

2.42 RA -- Microinstruction Address Register

The 9-bit RA register holds the address of a single 48-bit microinstruction word. This is actually a 10-bit buffer register comprising one 6-bit 74s174 and one 4-bit 74s175. The tenth bit position, however, is used to store the command-bit-derived signal FETCH, which is re-generated as FTDEL.

2.43 The 9-bit address code RAD0-8 is loaded from the control-store address bus at the leading-edge of clock pulse AP. The address code is produced by the Instruction Word logic or the Microprogram Control logic; the source is selected by gating (from the source selector) within that logic. The reset signal RSLBN resets the RA register to all zeros so that control-store address 000 is selected by the RA output. The signal RSLBN is derived from the GP-Bus reset-line signal RSLN, via RSLF.

2.44 Microinstruction Store (Control ROM)

This section is composed of six type 8205 read only memory (ROM) ICs. Each IC contains 512 eight-bit words which are accessed by nine addressing inputs. Addressing from the RA register is applied in parallel to the six ICs to obtain 512 48-bit words of store.

2.45 The enabling inputs of the control ROM are tied high; the outputs thus display continuously the content of whatever word is addressed by the RA register. When RA is reset by RSLN, the control-ROM address zero is selected (Idle state). A listing of all control-ROM microinstructions is provided in Table 2-4B. The binary contents of the control ROM are provided in Table 2-5.

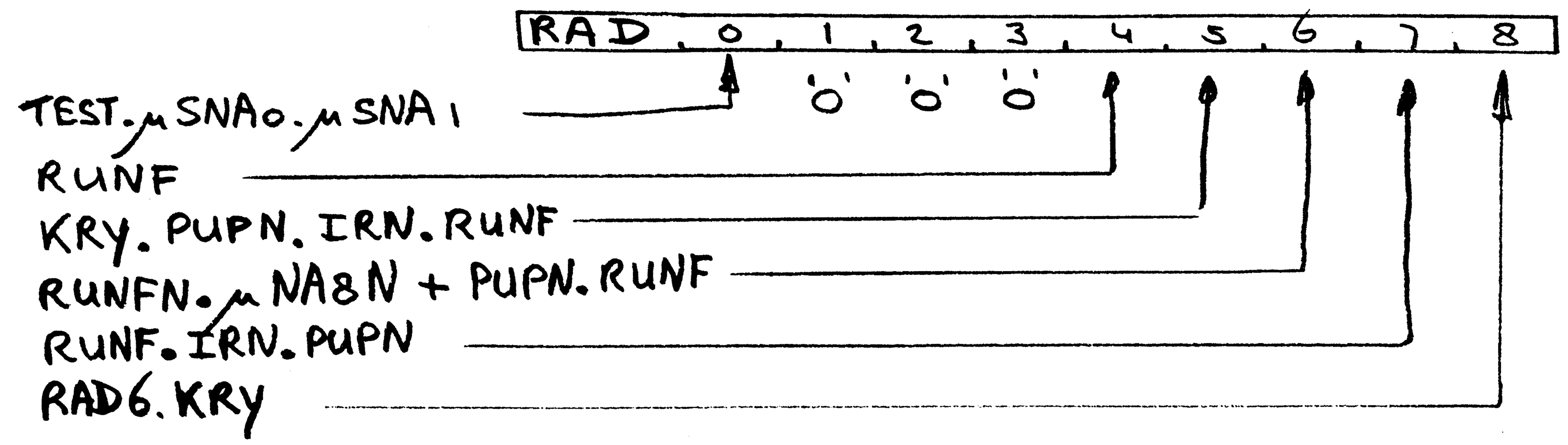
2.46 Microinstruction Decoding

Two groups of command bits (Table 2-2) are used by the CPU logic as direct command bits. The micro-command bits (μ...) are the actual bit contents of the currently-addressed micro-instruction word. The general-field section of these bits (μGP0-4) are decoded to produce a further 28 command bits (G...).

2.47 The general-field command bits (Figure 2-8BB) are decoded by four type 74s138 ICs. The signals μGP4,3,2 are connected in parallel to the A, B, C inputs. The gating inputs G1, G2A, G2B are connected as follows:

Bits:	1-7	9-15	16-23	24-29
G1	+5V	μGP1	μGP0	μGP0
G2A/	μGP0	μGP0	0V	$\overline{\mu GP1}$
G2B/	μGP1	BPN	μGP1	0V

The three gate inputs must all be active (G1-high; G2A, B-low) to enable the three A, B, C inputs. The resultant output for the different input combinations are listed in Table 2-3. Seven of the general-field commands are valid during BP for use as set/reset commands. Ten other general-field commands are stored at BP time and updated on the following BP; these are used as parameters for next-cycle exchange.



(machine state pointer addressing)

Table 2-3 General Field Command Bit Codes

μGP 01234	Command Bit	Clocked by BP	Stored on BP	Function	Application
00000	---				
00001	GIDLEN	--	--	Resynchro. Sequensor	Display, Tests.
00010	GCTLDN	--	--	Load Counter	Shift, DAR, DSR.
00011	GCRDSRN	--	--	Set OVF if Divide Error and DOOD	Shift right, Test Divide.
00100	GCRFNUN	--	--	Store OALU in FNU	IPL, PUP, Arith. Oper.
00101	GCRVMLN	--	--	Set CR if OVF; Branch for ML.	ML, Divide correction, Shift LA.
00110	GCSELN	--	--	ASR or INTAD on sel.C; data Q15	DLL, DLC, INR, Interrupt.
00111	GMOVEN	--	--	Branch during Move	Move read/write cycles.
01000	---				
01001	GFSYSN	Ys	--	Reset ENBF, ARF, PAF, FU.	System operations: INT, IPL, PAF, etc.
01010	GFENBN	Ys	--	Set ENBF	RTN with R2=15.
01011	GFSTOV	Ys	--	Allows set PIF if Stack OVF.	Update A15.
01100	GFPLR/ CLPLR	Ys	--	Loads PLR,FU by C	INT, AR, RTN Master
01101	GFRZO	Ys	--	Reset RUNF	Control-panel operations
01110	GFKYZON	Ys	--	Reset K Ready	INT, PAF, AR, LR, RR
01111	GCRVZON	Ys	--	Reset OVF; Divided Sign Store	Shift LA, DIV.
10000	GBCHN	--	Ys	Character Mode	Character instructions.
10001	GBCPN	--	Ys	C.Panel-B memory address	LM,RM.
10010	GBEXN	--	--	MMU translation to System Mode	EL,ES,MVUS,MVSU.
10011	GBOKN	--	Ys	(keys) on BIO	IPL, C.Panel, Tests.
10100	GBOFN	--	Ys	(FPP) on BIO	FPP store, FIX.
10101	GBOMN	--	Ys	(MMU) on BIO	TS, PAF.
10110	GBTMEN	--	Ys	Ext. Register Cmnd.	RER, WER.
10111	GBTMPN	--	Ys	Prog. Channel Cmnd.	I/O instructions.
11000	GBTMMN	--	Ys	MMU command	TL
11001	GBTMFN	--	Ys	FPP command	FL, FO, FOS.
11010	GFLOTN	--	Ys	FPP activation	Wait for FPP execution.
11011	GAEXLN	--	--	Execute double word	Trap for EX,MV,15R2, OPC=15.
11100	GFETCHN	--	--	Load K; set CT to 16	Fetch,EX,LR,RR,Tests.
11101	GMULTIN	--	--	D selection; OVF set for Multi	Last cycle of Multi.
11110	---				
11111	---				

Ys = Yes

Table 2-4A Contról-ROM Microinstruction Listing P857

ADD.		
000	ROM ,/1E8.AR15.ALUB.CALU.MYC.QYC,,,WBUS,,,O	IDLE
001	FTA ,,O	NO JUMP
002	ROM ,/1E9,,,,,SP2.PP2,,,BUSR,,O	INCR P
003	ROM FLAG,/ODO.AZ.ALUA.CALU,,QYC,,,,,,GAEXL,O	NO FLOT PROC
004	ROM FLAG./1F8.ASYS.TWOA.CALU,,QYC,,,,,,,,O	TEST DISPLAY
005	ROM ,/1FB,,,,,SM2.PM2,,,,GFKYZO.O	DECR BEFORE VISU
006	ROM ,/1FF.AEP.ALUA.CALU.MYC,,,,,,,,,O	VISUA
007	ROM ,/1FF,,DBIO.CBIO.MYC,,,,,SEQBUS,,,O	VISUB
008	FTA AWR1.ALUB.CRLOG	LD
009	ROM ,/1E9,,ALUB,,,SYD.PYD,,,BUSR,,CRLOG	LD P
00A	ROM ,/16D.ASYS.ALUA,,,,,,WBUS,,,O	RCP IN L REG. LR OR RR
00B	ROM ,/1FF.ARAO.ALUA,,MYC,,,,,,,,O	ERRONEOUS START
00C	ROM ,/101.ARR1.ALUA,,,,,,WMEM,,BUSR,,O	ST
00D	ROM ,/101.AEP.ALUA,,,,,,WMEM,,BUSR,,O	ST P
00E	ROM ,/OE6.ATWO.ALUA.CALU.MYC,,,,,,,,O	STD
00F	ROM ,/OE5.ATWO.ALUA.CALU.MYC,,,,,,,,O	STD P
010	ROM FLAG,/1DO.ASYS.ALUA.CALU,,QYC,,,,,,,,O	TEST AUTOREST
011	ROM ,/1E9,,ALUB,,,SYD.PYD,,,BUSR,,O	AB
012	ROM FLAG,/OEO.AWAO.ALUB,,,,,,RBUS,,GBOK,O	SAVE P TEST IF L/R R
013	ROM ,/OEO.ARAO.ALUA,,,SYD.PYD,,,,,O	LOAD P AND S LR
014	ROM ,/1D8.ATWO.TWOA.CALU.MYC,,,,,,,,O	INT
015	ROM ,/1D8.ATWO.TWOA.CALU.MYC,,,PM2,,,,GFKYZO.O	INT DECR
016	FTA ,,O	FETCH
017	ROM ETAT,/1FE,,,,,,,,,,SEQBUS,,GIDLE,O	VISU IDLE
018	ROM FLAG,/193.AEP.ALUA.CALU.MYC,,,,,,,,GCRVZO,O	SLASLN SAVE P
019	ROM FLAG,/158.AEP.ALUA.CALU.MYC.SLQ,,,,,,GCRVZO,O	DLADLN SAVE P
01A	ROM FLAG,/17B.ARR1.ALUA.CALU,,QYC,,,,,,,O	SLL SLC
01B	ROM ,/OCE,AWA1,ASHL,,,SLQ,,,CTP1,,REPEAT,,GCSEL,O	DLL DLC
01C	ROM FLAG,/16B.AEP.ALUA.CALU.MYC,,,,,,,,,O	SRA SRN
01D	ROM FLAG,/14B.AEP.ALUA.CALU.MYC.SLQ,,,,,,,,O	DRA DRN
01E	ROM ,/ODD.ARR1.ALUA.CALU,,GYC,,,,,,,O	SRL SRC
01F	ROM ,/OCE.AWA1.ASHR,,,SRQ,,,CTP1,,REPEAT,,GCRDSR,O	DRL DRC
020	FTA AWR1.APB.CRADD	AD
021	ROM ,/1E9.AEP.APB,,,SYD.PYD,,,BUSR,,CRADD	AD P
022	ROM ,/OEC.ARR2.ALUA,,,SYD,,,,BUSR,,O	RT3
023	ROM ,/OE8.ARR2.ALUA,,,SYD,,,,BUSR,,O	RT3S
024	ROM ,/101.ARR1.APB,,,,,,WMEM,,BUSR,,CRADD	AD S
025	ROM ,/101.AZ.APLB1,,,,,,WMEM,,BUSR,,CRADD	IM
026	ROM ,/077.ATWO.TWOA.CALU.MYC,,,,,,,,O	RTN MASTER
027	ROM ,/19E.AW15.AMB,,,SYD,,,,,GFSTOV,O	INT UPDATE A15
028	ROM FLAG,/ODO.AZ.ALUA.CALU,,QYC,,,,,,GAEXL,O	TRAP SLN
029	ROM ,/OE3.ATEN.TWOA.CALU.MYC.QYC,,PYD,,,,,O	SLN
02A	ROM ,/159.ATEN.ACR.CALU.MYC.QYC,,,,,,,O	IPL SET
02B	ROM ,/145.AEP.ALUA,,,SYD,,,RBUS,,BUSR.GBOK,O	IPL END
02C	ROM FLAG,/1D2.AWCT.DBIO,,MYQ,,SM2.PM2.CTP1,,SEQBUS,,,O	MLRI
02D	ROM ,/ODC,,ALUB,,,SYD.PYD,,,BUSR,,O	MLRI
02E	ROM ,/134,,,CLUR.MYC,,,,,,,GCSEL.O	AUTO RESTART
02F	ROM FLAG,/1CO.APUP.ALUA.CALU,,QYC,,,,,,,,O	TEST IPL
030	FTA AWR1.AMB.CRSUB	SU
031	ROM ,/1E9.AEP.AMB,,,SYD.PYD,,,BUSR,,CRSUB	SU P
032	ROM SNPLA./100,,,CBIO.MYC.QYC.SP2.PP2,,,SEQBUS,,,O	RT2
033	ROM SNPLA./100,,,CBIO.MYC.QYC,,PP2,,,SEQBUS,,,O	RT2S
034	ROM ,/101.ARR1.AMB,,,,,,WMEM,,BUSR,,CRSUB	SU S
035	ROM FLAG,/OBC.AEN.ALUA,,,,,,,,,,GCRFNU,O	C2 OR NGR
036	ROM ,/077.ATWO.TWOA.CALU.MYC,,,,,,,,,O	RTN USER
037	FTA AWR1.AOB.O	INH.RTN A15 AND LC END
038	ROM FLAG,/ODO.AZ.ALUA.CALU,,QYC,,,,,,,GAEXL,O	TRAP SRN
039	ROM ,/ODE.ARR1.ALUA.CALU,,QYC,,,,,,,,O	SRN
03A	ROM FLAG,/1C4,,BSHR.CALU.MYC,,,,,,,,GCRFNU,O	WAIT IO
03B	ROM ETAT,/1FF.AEP.ALUA,,,SYD,,,,,,O	WAIT IO
03C	ROM FLAG,/1C2.AWCT.DBIO,,,,SP2.PM2.CTP1,,SEQBUS,,GCRVML,O	ML LOOP
03D	ROM ,/ODC,,ALUB,,,SYD.PYD,,,BUSR,,O	ML END
03E	ROM ,/15F.ATWO.ACR,,,,PYD,,,,,O	IPL
03F	ROM FLAG,/1BO,,,,SLQ,,,RBUS,,,GBOK,O	TEST LOADM
040	FTA AWR1.AANDB.CRLOG	AN
041	ROM FLAG./1C8.APSW.AANDB.CALU.MYC,,,,,,,BUSR,,O	HLT,RIT,INH,RTA15
042	ROM ,/OFD,,,CBIO.MYC,,,PP2,,,SEQBUS,,,O	RT5
043	ROM ,/OF5,,,CBIO.MYC,,,PP2,,,SEQBUS,,,O	RT5S
044	ROM ,/101.ARR1.AANDB,,,,,,WMEM,,BUSR,,CRLOG	AN S
045	ROM ,/101,,ZERO,,,,,,WMEM,,BUSR,,O	CM
046	FTA ARR1.AANDB.CRLOG	TM
047	ROM ,/1E9,,,,,,,,,,,BUSR,,CRFLO	FPP WAIT EXEC.
048	ROM FLAG,/1A6.ARA1.DIVALU,,,,,,,,,GCRFNU,O	DIV TEST COR.REM.
049	ROM FLAG,/140.ARA1.AXB.CALU.MYQ.QYC,,,,,,,,O	DIV TEST QUOT.
04A	ROM ,/OE2.AEP.AMB.CALU.MYC,,,,,,,,,O	SLN
04B	ROM FLAG,/1B5.AWR1.TWOA,,,,,PP2,,,,,,O	SLN LOOP
04C	ROM FLAG,/1B2.ARCT.ALUA,,,SP2.PM2.CTP1.WMEM.SEQBUS,,,O	MS LOOP
04D	ROM ,/1E9,,ALUB,,,SYD.PYD,,,BUSR,,O	MS END
04E	ROM FLAG,/1A8,,,,,SLQ,,,,,,,GFRZO.O	TEST READM
04F	ROM ,/172,,DBIO.CBIO.MYC,,,,,WMEM.SEQBUS,,GBCP,O	LOAD M
050	FTA AWR1.AOB.CRLOG	OR
051	ROM FLAG,/188.APSW.AOB.CALU.MYC,,,,,,,,,O	ENB SMD LKM
052	ROM ,/OEC,,DBIO,,,SYD.PP2,,,SEQBUS.BUSR,,O	RT4
053	ROM ,/OE8,,DBIO,,,SYD.PP2,,,SEQBUS.BUSR,,O	RT4S
054	ROM ,/101.ARR1.AOB,,,,,,WMEM,,BUSR,,CRLOG	OR S
055	ROM ,/OA8.AWAO.BSHR,,MYQ,,,,,,,,,O	INR
056	ROM FLAG./1AO,,,,SLQ,,,,,,,,O	TEST READST
057	ROM FLAG,/190.ASYS.TWOA.CALU,,QYC,,,,,,GBCP,O	READ M
058	ROM ,/1A6.AWA1.DIVALU,,,,,,,,,,,O	DIV CORR.REM.
059	ROM FLAG,/140.ARA1.AXB.CALU.MYQ.QYC,,,,,,,,O	DIV TEST QUOT.
05A	ROM ,/OE2.AEP.AMB.CALU.MYC,,,,,,,,,O	SRN
05B	ROM FLAG,/1A5.AWR1.ASHR,,,SRQ,,PP2,,,,,GCRDSR,O	SRN LOOP
05C	ROM FLAG,/1A2.ARCT.ALUA,,,SM2.PM2.CTP1.WMEM.SEQBUS,,,O	MSRD LOOP
05D	ROM ,/OBE,,ALUB,,MYQ,,SYD.PYD,,,,,,O	MSRD END
05E	ROM ,/198,,,,,SLQ,,,,,,,,O	PUP
05F	ROM ,/1FF.APSW.ALUA.CALU.MYC,,,,,,,,,O	READ STATUS
060	FTA AWR1.AXB.CRLOG	XR
061	ROM ,/047.APSW.ALUA.CALU.MYC,,SP2,,,,,,GFSYS.O	INT
062	ROM ,/OFC,,,CBIO.MYC,,,,,,SEQBUS,,,O	RT7
063	ROM ,/OF4,,,CBIO.MYC,,,,,,SEQBUS,,,O	RT7S
064	ROM ,/101.ARR1.AXB,,,,,,WMEM,,BUSR,,CRLOG	XR S
065	FTA AWA2.BSHR.O	DLN END
066	FTA ARR1.AXB.CRLOG	TNM
067	ROM ,/180.AWAO.ALUB,,MYQ,,,,,,,,,O	PUP
068	ROM FLAG,/ODO.AZ.ALUA.CALU,,QYC,,,,,,,GAEXL.O	MVF 15R2 TRAP
069	ROM ,/05F.ARR2.ALUA.CALU,,QYC,,PYD,,,,,,O	MVF 15R2N
06A	ROM ,/OD6.AWAO.ALUB,,,,,PM2,,,,,,O	DLNDRN END
06B	ROM FLAG,/195.AWA1.ASHL,,,SLQ,,PP2,,,,,,O	DLN LOOP
06C	ROM ,/OE4.AWR1.TWOA,,,,,,CTP1,,REPEAT,,,O	SLA
06D	ROM FLAG,/107.AWAO.ALUB,,,,,,,,,,,O	SLN
06E	ROM ,/1FF,,DBIO.CBIO.MYC,,SP2.PP2,,,SEQBUS,,,O	READ M PUP A
06F	ROM ,/1FF,,DBIO.CBIO.MYC,,,,,SEQBUS,,,O	READM PUP B
070	ROM SNPLA./100.AR1215,,,,,,,,,,,GCTLD.O	SH DSH
071	ROM ,/18F.ARA2.ALUA.CALU,,QYC,,,,,,,GFKYZO.O	DSH
072	ROM ,/1AD,,DBIO,,,,SYD,,,,SEQBUS.BUSR,,O	RT6
073	ROM ,/1AC,,DBIO,,,,SYD,,,,SEQBUS.BUSR,,O	RT6S
074	FTA ARR1.ALUA.CRLOG	SHZERO
075	ROM ,/ODC.ARA2.ALUA,,,,,,,,,,BUSR.GCRFNU.O	DSHZERO
076	ROM ,/092.AWA1.DSUB,,,,,,,,,,BUSR,,CRSUB	DS
077	ROM ,/1C5.ATWO.ALUA.CALU.MYC,,,,,,,,,O	WAIT
078	ROM ,/OC7.AEP.ALUA.CALU.MYC,,,,CTP1,,,,,O	ML
079	ROM ,/150.ATWO.ASHR.CALU,,QYC,,,,,,,GBTMM.O	TL
07A	ROM ,/195.AWA1.ASHL,,,SLQ,,,,,,,,,O	DRN
07B	ROM FLAG,/185.AWA1.ASHR,,,SRQ,,PP2,,,,,GCRDSR,O	DRN LOOP
07C	ROM ,/OC5.AEP.ALUA.CALU.MYC,,,,CTP1,,,,,O	MS
07D	ROM ,/148.ATWO.ASHR.CALU,,QYC,,,,WEXM,,,GBOM.O	TS
07E	ROM ,/OC3.AEP.ALUA.CALU.MYC,,,,CTP1,,,,,O	MSRD
07F	ROM FLAG,/1F4,,BCR.CALU,,QYC,,,,,,,GCRFNU	PUP
080	ROM ,/008,,ALUB,,,,SYD,,,,,,,O	CIO OTR
081	ROM ,/100,,,,,,,,,,,,,,,O	GO TO TRAP (WMP)
082	ROM ,/1B2.AWA2.DBIO,,,,,,,RBUS.SEQBIO,,,O	FFX LD A2
083	ROM SNPLA./100.ARR2.ALUA.CALU.MYC.QYC,,,,,,,,O	RT1
084	ROM ,/18B.AWR1.TWOA,,,,,,CTP1,,REPEAT,,,O	SLL
085	ROM ,/18B.AWR1.ASHL,,,SLQ,,,CTP1,,REPEAT,,GCSEL.O	SLC
086	ROM ,/169,,,,,,,,CTP1,,,,,O	RT1D
087	ROM ,/179.ARR2.ALUA.CALU,,QYC,,,,,,,GCTLD.O	RT1D
088	ROM FLAG,/OB3,,ZERO.CALU.MYC,,,,,,,BUSR.GBTMF,O	FPP LOAD
089	ROM ,/OBA.AWA1.ZERO.CALU,,QYC,,,,,,,GCRVZO.O	MUL-RESET PART.PROD
08A	ROM ,/130.ARAO.ALUA,,,,,,,WBUS,,BUSR,,O	EX T1 OR T3
08B	ROM FLAG,/ODO.AZ.ALUA.CALU,,QYC,,,,,,,GAEXL,O	EX T2 TRAP
08C	ROM FLAG,/OAB.ATEN.ACR.CALU.MYC.QYC,,,,WEXM,,,GBOF,O	FPP STORE
08D	ROM ,/1A8,,,,,,,,,,,SEQBUS,,GFRZO.O	LOAD M
08E	ROM FLAG,/1D4.AEP.ALUA,,,,SP2,,,,SEQBUS,,GCRFNU,O	IPL LOOP
08F	ROM ,/159,,BSHR.CALU.MYC.SLQ,,,,,,,,O	IPL LOOP
090	ROM ,/1AA.ATEN.ACR.CALU.MYC,,,,,,,,,O	INR
091	ROM SNPLA./103.AEP.ALUA,,,,SYD,,,,,,,O	EX BSYZO,RESTORE S
092	ROM ,/1ED.AEP.ALUA.CALU.MYC,,,,,RBUS.SEQBIO,,GFETCH,O	K-RCP L/RR
093	ROM SNPLA./100.AEP.ALUA.CALU.MYC.QYC,,,,,,,,O	RT1P
094	ROM ,/18B.AWR1.ASHR,,,,,,CTP1,,REPEAT,,GCRDSR,O	SRA
095	ROM FLAG,/1C7.AWAO.ALUB,,,,,,,,,,,O	SRN
096	ROM SNPLA./100.ARCT.ALUA.CALU.MYC,,,,,,,,,O	RT1D END
097	ROM ,/169.AEP.ALUA.CALU,,QYC,,,CTP1,,,,,O	RT1DP
098	ROM FLAG,/OB3,,ZERO.CALU.MYC,,,,,,,BUSR.GBTMF,O	FPP OPER
099	ROM ,/OA5.ARA2.TWOA.CALU,,QYC,,,,,,,,O	DIV
09A	ROM ,/095.ARA1.ALUA,,,,,,,WBUS,,BUSR.GBTMF,O	FFL
09B	ROM ,/OFE.AWR2.ALUB,,,,,,,,,,,O	DLN,DRN END
09C	ROM FLAG,/OB3.ATWO.ASHR.CALU.MYC,,,,,,,BUSR.GBTMF,O	FPP OP/S
09D	ROM ,/143.AEP.ALUA.CLUR.MYC,,SP2,,,WMEM.SEQBUS.BUSR.GCSEL,O	INT
09E	ROM ,/160.AEP.ALUA.CALU.MYC,,,,,,,,GFLOT.O	FFX
09F	ROM ,/09B,,,,,,,,,,RBUS,,,GBOF.CRFLO	FFX UPD CR
0A0	ROM ,/1D5,,ZERO,,,,SYD,,,,,,GFSYS.O	IPL
0A1	ROM ,/1E9.AEP.APB,,,SYD.PYD,,,BUSR,,O	RF
0A2	ROM ,/OE7.ATWO.ALUA.CALU.MYC,,,,,,,,,O	RT3B
0A3	ROM SNPLA./100.ARR2.ALUA.CALU,,QYC.SYD,,,,,,,O	RT3C
0A4	ROM ,/ODB.AWA1.ASHL,,,SLQ,,,CTP1,,REPEAT,,,O	DLA
0A5	ROM FLAG,/01F.ATWO.ALUA,,,,,PYD,,,,,,O	DLN
0A6	ROM ,/155,,BSHR.CALU.MYC,,,PM2,,,,,,O	IPL LOOP
0A7	ROM ,/OCB.AEP.ALUA.CALU,,QYC,,,,,,,,O	RT3BM
0A8	ROM ,/08D,,,,,,,,,,,,BUSR.GBEX.O	EL
0A9	ROM ,/08E,,,CBIO,,QYC.SP2,,,,SEQBUS.BUSR,,O	DA
0AA	ROM ,/14F,,BSHR.CALU.MYC,,,,,,,,,O	IPL LOOP
0AB	ROM ,/152,,,CBIO.MYC,,SP2.PP2,,,SEQBUS,,,O	DAK
0AC	ROM ,/101.ARR1.ALUA,,,,,,WMEM,,BUSR.GBEX.O	ES
0AD	ROM ,/094.AWA2.APB.CALU.MYQ.QYC,,,,,,,,O	DA
0AE	ROM ,/1E9.AEP.ALUA,,,SYD,,,,SEQBUS,,,O	TL END
0AF	ROM FLAG,/150,,,,,SLQ.SP2,,,,SEQBUS.BUSR.GBTMM.O	TL LOOP
0B0	ROM ,/14D,,BSHR.CALU.MYC,,,,,,,,,O	IPL LOOP
0B1	ROM ,/1E9.AEP.AMB,,,SYD.PYD,,,BUSR,,O	RB
0B2	ROM FLAG,/170.AIPL.APB.CALU.MYC,,,,,WMEM,,,,O	IPL NEW QUART
0B3	ROM SNPLA./100,,,,,,,,PP2,,,,,,O	RT2C
0B4	ROM ,/OD2.AWA1.ASHR,,,SRQ,,,CTP1,,REPEAT,,GCRDSR.O	DRA
0B5	ROM FLAG,/OOF.ATWO.ALUA,,,,,PYD,,,,,,O	DRN
0B6	ROM ,/1E9.AEP.ALUA,,,SYD,,,,SEQBUS,,O	TS END
0B7	ROM FLAG,/148,,,,,SLQ.SP2,,,WEXM.SEQBUS.BUSR.GBOM.O	TS LOOP
0B8	ROM FLAG,/ODO.AZ.ALUA.CALU,,QYC,,,,,,,GAEXL,O	TRAP
0B9	ROM ,/08A,,,CBIO,,QYC.SP2,,,,SEQBUS.BUSR,,O	DS
0BA	ROM ,/1E9.AW15.DBIO,,,,,,,,SEQBUS,,,O	IPL END
0BB	ROM ,/142,,,CBIO.MYC,,SP2.PP2,,,SEQBUS,,,O	DSK
0BC	ROM ,/13D,,BSHR,,,,SYD,,,,SEQBUS,,,O	INT
0BD	ROM ,/08B.AWA2.AMB.CALU.MYQ.QYC,,,,,,,,O	DS
0BE	ROM ,/09E.AZ.APLB1.CALU.MYC,,,,,,,BUSR.GCRVML.CRADD	DIV COR.QUOT
0BF	ROM ,/1E9.AWA2.ALUB,,,,,,,,,,BUSR.GCRVML.CRLOG	DIV QUOT OK

1/9/75

Table 2-3a General Field Command Bit Codes

μGP 01234	Command Bit	Clocked by BP	Stored on BP	Function	Application
00000	---				
00001	GIDLEN	--	--	Resynchro. Sequensor	Display, Tests.
00010	GCTLDN	--	--	Load Counter	Shift, DAR, DSR.
00011	GCRDSRN	--	--	Set OVF if Divide Error and DOOD	Shift right, Test Divide.
00100	GCRFNUN	--	--	Store OALU in FNU	IPL, PUP, Arith. Oper.
00101	GCRVMLN	--	--	Set CR if OVF; Branch for ML.	ML, Divide correction, Shift LA.
00110	GCSELN	--	--	ASR or INTAD on sel.C; data Q15	DLL, DLC, INR, Interrupt.
00111	GMOVEN	--	--	Branch during Move	Move read/write cycles.
01000	---				
01001	GFSYSN	Ys	--	Reset ENBF, ARF, PAF, FU.	System operations: INT, IPL, PAF, etc.
01010	GFENBN	Ys	--	Set ENBF	RTN with R2=15.
01011	GFSTOV	Ys	--	Allows set PIF if Stack OVF.	Update A15.
01100	GFPLR/ CLPLR	Ys	--	Loads PLR,FU by C	INT, AR, RTN Master
01101	GFRZO	Ys	--	Reset RUNF	Control-panel operations
01110	GFKYZON	Ys	--	Reset K Ready	INT, PAF, AR, LR, RR
01111	GCRVZON	Ys	--	Reset OVF; Divided Sign Store	Shift LA, DIV.
10000	GBCHN	--	Ys	Character Mode	Character instructions.
10001	GBCPN	--	Ys	C.Panel-B memory address	LM,RM.
10010	GBEXN	--	--	MMU translation to System Mode	EL,ES,MVUS,MVSU.
10011	GBOKN	--	Ys	(keys) on BIO	IPL, C.Panel, Tests.
10100	GBOFN	--	Ys	(FPP) on BIO	FPP store, FIX.
10101	GBOMN	--	Ys	(MMU) on BIO	TS, PAF.
10110	GBTMEN	--	Ys	Ext. Register Cmnd.	RER, WER.
10111	GBTMPN	--	Ys	Prog. Channel Cmnd.	I/O instructions.
11000	GBTMMN	--	Ys	MMU command	TL
11001	GBTMFN	--	Ys	FPP command	FL, FO, FOS.
11010	GFLOTN	--	Ys	FPP activation	Wait for FPP execution.
11011	GAEXLN	--	--	Execute double word	Trap for EX,MV,15R2, OPC=15.
11100	GFETCHN	--	--	Load K; set CT to 16	Fetch,EX,LR,RR,Tests.
11101	GMULTIN	--	--	D selection; OVF set for Multi	Last cycle of Multi.
11110	---				
11111	---				

Ys = Yes

Table 2-3b Control-ROM Microinstruction Field Descriptions (P856/7)

```
ROM      FORM     2,9,5,5,2,2,2,2,2,1,3,2,1,5,5,16*0
*
*
*        FTA REALISE LE FETCH ANTICIPE
*
*        MICROCOMMANDES UTILISES
*        ETAT
*        /1FF
*        CIOR
*        MYC
*        QYC
*        SP2
*        PP2
*        CTINH
*        RMEM
*        SEQBUS
*        PAS DE BUS REQUEST
*        GFETCH
*
*        SEULS PARAMETRES A PRECISER :
*        VOIE A
*        ALU,D,L
*        CR
*
FTA FORM 2*3,9*/1FF,5,5,2*3,2*2,2*3,2*2,2*3,1*0,3*0,2*2,1*0,5*28,5,16*0
         EJECT
*
* LISTE DES MICROCOMMANDES
*
* SELECTION NEXT ADDRESS
*
EXPL     EQU      0
FLAG     EQU      1
SNPLA    EQU      2
ETAT     EQU      3
*
* NEXT ADDRESS  (COMPLEMENT)
*
R512     EQU      0              PAF  FIN D INSTRUCTION
IDLE     EQU      /1FF           PLA MODE D ADRESSAGE
NOEND    EQU      1              FIN D OPERATION PUPITRE
EXEC     EQU      /1FD           PLA MODE EXECUTION
         EJECT
*
* SELECTION EMETTEUR VOIE A
*
ARA0     EQU      0
ARA1     EQU      1
ARA2     EQU      2
AR15     EQU      3
ARR1     EQU      4
ARR2     EQU      5
AR1215   EQU      6
ARCT     EQU      7
AWA0     EQU      8
AWA1     EQU      9
AWA2     EQU      10
AW15     EQU      11
AWR1     EQU      12
AWR2     EQU      13
AW1215   EQU      14
AWCT     EQU      15
AZ       EQU      16
AIPL     EQU      18
AEP      EQU      20
APSW     EQU      22
AEN      EQU      24
ATEN     EQU      26
ATWO     EQU      27
APUP     EQU      28
ASYS     EQU      30
         EJECT
*
* ALU, SELECTEYR D, REGISTRE L
*
AMB      EQU      0
ALUB     EQU      1
AOB      EQU      2
ALUA     EQU      3
APLB1    EQU      4
BINV     EQU      5
APB      EQU      6
AXB      EQU      7
AANDB    EQU      8
BCR      EQU      9
ZERO     EQU      10
ACR      EQU      11
BSHR     EQU      13
DBIO     EQU      14
ASHR     EQU      15
FORA     EQU      16
ASHL     EQU      17
TWOA     EQU      18
MULTI    EQU      20
DSUB     EQU      24
DADD     EQU      26
DIVALU   EQU      28
DIVSH    EQU      30
         EJECT
*
* SELECTEUR C
*
CALU     EQU      0
CBIO     EQU      1
CLUR     EQU      2
CIOR     EQU      3
*
```

```
* REGISTRE M
*
MYC      EQU     2
MYQ      EQU     3
*
* REGISTRE Q
*
SRQ      EQU     2
SLQ      EQU     1
QYC      EQU     3
*
* COMPTEUR S
*
SYD      EQU     1
SM2      EQU     3
SP2      EQU     2
*
* COMPTEUR P
*
PYD      EQU     1
PM2      EQU     2
PP2      EQU     3
*
*COMPTEUR CT
*
CTP1     EQU     1
         EJECT
*
* ECHANGE SUR LE BUS
*
RMEM     EQU     Ø         TMR
WMEM     EQU     3         TMR  BIO=L
RBUS     EQU     4         PAS DE TMR
WBUS     EQU     7         NØ TMR   BIØ=L   WRITE=1
WEXM     EQU     1         EXTRN WRITE MEMORY
*
* SEQUENSEUR
*
REPEAT   EQU     1
SEQBUS   EQU     2
SEQBIO   EQU     3         BSYZØ  IN T6 OF NEXT CYCLE
*
* DEMANDE DE BUS ANTICIPEE
*
BUSR     EQU     1
         EJECT
*
* DECODAGES DIVERS
*

BIDLE    EQU     1         LONG CYCLE IF TEST
GCTLD    EQU     2         CTYSPA PAFZO
GCRDSR   EQU     3         OVFZ1 SI ERREUR DIVI; ENTREE GAUCHE DE D
GCRFNU   EQU     4         FNU Y QALU
```

```
GCRVML   EQU     5         CRZ3 SI OVF; DEROUTEMENT ML
GCSEL    EQU     6         C=ASR OU INTAD;Q15D=ALU00,K08
GMOVE    EQU     7         DEROUTEMENT EN MOVE TABLE
GFSYS    EQU     9         ENBFZO ARFZO FUZO
GFENB    EQU     10        ENBFZ1
GFSTOV   EQU     11        PIFZ1 SI STKOV
GFPLR    EQU     12        PLR Y C; FU Y C15
GFRZØ    EQU     13        RUNFZØ
GFKYZØ   EQU     14        KRYZØ
GCRVZØ   EQU     15        OVFZØ ; MEMO SIGNE DU DIVIDENDE
GBCH     EQU     16        MODE CARACTERE
GBCP     EQU     17        MAD=REG D ADRESSE PUPITRE B
GBEX     EQU     18        PASSER PAR MMU MEME EN MODE SYSTEME
GBOK     EQU     19        BIO = KEY
GBOF     EQU     20        BIO = FLOTTANT
GBOM     EQU     21        BIO = MMU
GBTME    EQU     22        TME
GBTMP    EQU     23        TMP
GBTMM    EQU     24        TMM
GBTMF    EQU     25        TMF
GFLOT    EQU     26        ACTIVATION DU PROCESSEUR FLOTTANT
GAEXL    EQU     27        A= SOUS EXEXUTE DOUBLE MOT
GFETCH   EQU     28        CTZ16, KYBIO
GMULTI   EQU     29        INHIBITS DSHR AT 16TH PASS;ALLOWS OVF MUL
*
* REGISTRE DE CONDITION
*
CRRTN    EQU     4
CRIO     EQU     8
CRFLO    EQU     12
CRLOG    EQU     16
CRADD    EQU     20
CRSUB    EQU     24
CRCMP    EQU     28
         EJECT
BUFFER   DATA    Ø
```

ADD		ADD.		ADD.	
0C0	FTA AWR1.ALUB.0 LDK	100	ROM ,/03F,,,,,,,,,,,,GFSYS,0 VERIF MAN. PUPIT	140	ROM ,/07E,,,,,,,,,RBUS,,GBOK,0 TEST MAN REG M
0C1	ROM ,/1E9,,ALUB,,,,SYD,PYD,,,,BUSR,,0 LDK P	101	ROM ,/19A,ARA0,ALUA,,MYQ,,,PYD,,,,BUSR,,0 DLN,DRN END	141	ROM ,/1E9,AWR2,AMB,,,,,,,,,BUSR,GFSTOV,0 UPD STK MSRD
0C2	ROM ,/135,,BCR,,,,,,,,,BUSR,GFPLR,0 AR PAF INT	102	ROM ,/0EC,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT5	142	ROM ,/1CF,AWR1,ZERO,,,,,,,,,BUSR,,0 NGR
0C3	ROM ,/0EE,,,CBIO,MYC,,,PP2,,,SEQBUS,,,0 RT5C	103	ROM ,/1AD,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT7	143	ROM ,/101,AZ,AMB,,,,,,,WMEM,,BUSR,,CRSUB C2
0C4	ROM ,/04F,AWA2,APB,,,,,,,,,,,0 MVF DEST IN A2	104	ROM ,/0D3,,,CBIO,MYC,,,,,RBUS,SEQBIO,,,0 TMP TEST RD KEYS	144	ROM ,/188,,ALUB,,,,,,,WBUS,SEQBIO,,,CRIO CIO OTR
0C5	ROM ,/1E9,ARA0,ALUA,,,,SYD,PYD,,,,BUSR,,0 MOVE END SLN REST.P	105	ROM FLAG,/0D4,ARA0,ALUA,,,,,,,,,,GCRFNU,0 TEST RETURN	145	ROM ,/0B7,ARA2,ALUA,CALU,,QYC,,,,,,,,0 MU- MUL.TOR IN Q REG.
0C6	ROM ,/088,ARR1,ALUA,CALU,,QYC,,,,,,,,0 CA	106	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 EX OPER T4T7 TRAP	146	ROM FLAG,/01C,ATWO,APB,,MYC,,,,,,,,,GCRFNU,0 TEST NO 1
0C7	FTA ARR1,AMB,CRCMP CA	107	ROM ,/066,,BSHR,CALU,MYC,,,,,,,,,,0 EX OPER T1,T2,T3	147	ROM ,/018,,,,,SLQ,,,,,,,,0 TEST NO 1
0C8	ROM ,/086,ARR1,ACR,CLUR,MYC,,,,,,,BUSR,GBCH,0 LC	108	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 MVB 15R2FU TRAP	148	ROM ,/0B6,AWA1,MULTI,,,SRQ,,,CTP1,,REPEAT,,,0 MU- ALGORITHM
0C9	FTA AWR1,BCR,0 ECR	109	ROM ,/04B,ARR2,ALUA,,,,,PYD,,,,,,0 MVB 15R2FUN	149	ROM ,/0B5,AWA1,MULTI,,MYQ,,,,,,,,GMULTI,0 MULTI 16TH PASS
0CA	ROM ,/1E9,,DBIO,,,,SYD,PYD,,,SEQBUS,BUSR,,0 AR PAF INT TRAP	10A	ROM ,/0E8,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT5S	14A	ROM ,/0B4,AWA2,BSHR,,,,,,,,,,,0 MULTI STORE LSB OF PROD.
0CB	ROM ,/132,,,,,,,,,,,,,GFSYS,0 AR END	10B	ROM ,/1AC,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT7S	14B	ROM ,/0D7,ARA2,ALUA,,,,,,,,,,GCRFNU,0 MULTI UPDATE FNU
0CC	ROM ,/101,ARR1,ALUA,,,,,,,WMEM,,BUSR,GBCH,0 SC	10C	ROM ,/04A,ATWO,ALUA,CALU,MYC,QYC,,,,,,,,0 MVB LENGTH>0	14C	ROM ,/0AD,,ALUB,,,SRQ,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP SINGLE RD MA
0CD	ROM ,/13D,,BSHR,,,,SYD,,,,,,GFKYZ0,0 PAGE FAULT AR	10D	ROM ,/1E9,ARA0,ALUA,,,,SYD,PYD,,,,BUSR,,0 MVB LENGTH=0	14D	ROM ,/042,,,,,,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP OP/S DOUBLE READ M1
0CE	ROM FLAG,/128,ARA0,APB,CLUR,MYC,SLQ,,,,,,,,0 EX T8N TEST K1	10E	ROM ,/1EB,ARA0,AMB,,,,,PYD,,,,,,0 INT MOVE	14E	ROM ,/053,ATWO,ASHR,,MYC,QYC,SM2,PM2,,WMEM,,,,0 T. MEMOIRE NO1
0CF	ROM ,/16E,,,CIOR,MYC,QYC,,,,WBUS,SEQBIO,,GFETCH,0 EX LOAD K	10F	ROM ,/125,ATWO,TWOA,CALU,,QYC,,,,,,,,0 PAF MOVE	14F	ROM ,/06C,AWA1,ALUB,,,,SM2,PM2,,,,,,0 TEST MEMOIRE NO1
0D0	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	110	ROM ,/0DF,,,,,,,,,,,,,GFRZ0,0 VERIF MAN. REG L	150	ROM ,/09F,,,,,,,,,,,,,GFRZ0,0 TEST Q,CHARG.NO
0D1	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	111	ROM SNPLA,/100,ARR2,APB,CALU,,QYC,SYD,,,,,,,0 RT5C	151	ROM ,/0A1,AWA1,BCR,,MYC,QYC,,,,,,,,0 TEST DLA
0D2	ROM ,/125,AWR2,ALUB,,,,,,,,,,,0 PAGE FAULT ML	112	ROM ,/12C,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT7C	152	ROM FLAG,/030,,,,MYQ,,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP READ LAST MA
0D3	ROM SNPLA,/100,,DBIO,CBIO,,QYC,SYD,PP2,,,SEQBUS,,,0 RT4C	113	ROM SNPLA,/100,AEP,ALUA,CBIO,MYC,QYC,SYD,,,,SEQBUS,,,0 RT3	153	ROM ,/173,ARA0,TWOA,,,,SYD,,,,,,,0 FPP OPS WAIT.LD S WITH 1ST AD.
0D4	ROM FLAG,/11B,AWA2,AMB,CBIO,MYC,,SYD,,,,SEQBUS,,GMOVE,0 MVF READ	114	ROM ,/048,AWA2,APB,CBIO,MYC,,SYD,,,,SEQBUS,,GMOVE,0 MVB READ	154	ROM ,/010,,BINV,CALU,,QYC,SP2,,,WEXM,SEQBUS,BUSR,GBOF,0 FPP/S SI.
0D5	ROM ,/056,AWA1,APB,,,,,,,,,,,0 MVF REST A1	115	ROM ,/056,AWA2,APB,,,,,,,,,,,0 MVB CORRECT A2	155	ROM ,/010,,,,,,SP2,,,WEXM,SEQBUS,BUSR,GBOF,0 FPP/S DOUBLE ST M1
0D6	ROM FLAG,/120,,,,,SLQ,,,,,,,,0 EX K1 TEST K2	116	ROM ,/0AE,ATEN,ALUA,,MYC,,,,,,,,GFRZ0,0 TEST DLA	156	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA
0D7	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,0 EX K1N TEST MD	117	ROM SNPLA,/100,,,CBIO,MYC,QYC,,,,,SEQBUS,,,0 RT3S	157	ROM ,/0A7,ARA0,A0B,,,,SYD,,,RBUS,,BUSR,GBTMP,0 INR
0D8	FTA ARR1,AMB,CRCMP CW	118	ROM ,/0EC,AWR2,APB,,,,SYD,,,,,BUSR,,0 RT3B UPDATE STACK	158	ROM ,/0A6,AWR1,DBIO,CIOR,MYC,,,,,,SEQBUS,,GCSEL,CRIO INR
0D9	FTA AEP,AMB,CRCMP CWP	119	ROM ,/1F3,AWR2,AMB,,,,,,,,,,GFSTOV,0 STD	159	ROM ,/188,AWR1,A0B,,,SLQ,,,,,,,,0 INR TST SST
0DA	ROM ,/124,APSW,ALUA,CALU,MYC,,,,,,,,,GFSYS,0 PAGE FAULT	11A	ROM ,/1F2,AWR2,AMB,,,,,,,,,,GFSTOV,0 STDP	15A	ROM ,/0A4,AWA1,ALUA,,,,,,,,,,GCRVZ0,0 DIV MEMO SIGNE DIVD
0DB	ROM ,/122,AR15,ALUA,,,,SYD,,,WEXM,,BUSR,GBOM,0 PAGE FAULT	11B	ROM ,/1E9,ARR1,ALUA,,,,,,,,,BUSR,GCRVML,CRLOG SLA CR	15B	ROM ,/0A3,AWA1,DIVSH,,,SLQ,,,CTP1,,,,GCRDSR,0 DIV 1ST PASS
0DC	ROM ,/08C,ARR1,ALUA,CLUR,MYC,,,,,,,BUSR,GBCH,0 CC	11C	ROM FLAG,/1B5,ARR1,ALUA,,,,,,,,,,,0 SLN TEST NORM	15C	ROM ,/0A2,AWA1,DIVSH,,,SLQ,,,CTP1,,REPEAT,,,0 DIV PROCESS
0DD	ROM ,/11D,,ALUB,,MYQ,,SM2,,,WMEM,SEQBUS,BUSR,,0 PAGE FAULT	11D	ROM ,/13A,AWR2,BSHR,,,,,,,,,,,0 SLN RESULT	15D	ROM FLAG,/1B6,AWA1,DIVALU,,,SLQ,,,,,,,GCRFNU,0 DIV 16TH PASS
0DE	ROM FLAG,/118,,,,,SLQ,,,,,,,,0 EX K1K2 TEST K3	11E	ROM ,/1EC,AW1215,DBIO,,,,,,,RBUS,SEQBIO,,,0 KEYS IN (RCP) LR	15E	ROM ,/0A0,ATWO,APB,CLUR,MYC,,,PYD,,,,,,0 TEST DLA
0DF	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,0 EX K1K2N TEST MD	11F	ROM ,/1FF,AR1215,ALUA,,MYC,,,,,,,,GFKYZ0,0 LOAD OR READ R	15F	ROM ,/091,AWA2,BCR,,,SRQ,,,,,,,,0 TEST DLA
0E0	ROM ,/07B,,ALUB,,,,SYD,,,,,,,0 WER	120	ROM ,/009,,,,,,,,,,RBUS,,,GBOK,0 VERIF MAN. REG L	160	ROM ,/039,,,,,,,,,RBUS,,,GBOK,0 TEST Q,CHARG NO.
0E1	ROM FLAG,/197,AEP,ALUA,CALU,MYC,,,,,,,,,0 MVF	121	ROM FLAG,/1A5,ATEN,TWOA,CALU,MYC,,,PYD,,,,,,U SRN	161	FTA AWA2,ALUB,0 DIV ST CORTD QUOT
0E2	ROM ,/117,ATWO,APB,CALU,MYC,,SM2,,,WEXM,SEQBUS,BUSR,GBOM,0 PAF	122	ROM ,/18B,AWR1,ASHR,,,SRQ,,,CTP1,,REPEAT,,GCRDSR,0 SRLC	162	ROM ,/09F,,,,,,,,,,,,,,0 TMP TEST
0E3	ROM ,/0ED,,,CBIO,MYC,,,,,,SEQBUS,,,0 RT7C	123	FTA ARA1,ALUA,CRLOG ML DSH	163	ROM ,/033,AZ,ALUA,,MYC,,,,,,,,,0 TMP TEST
0E4	ROM ,/057,,ALUB,,MYQ,,,PM2,,WMEM,,BUSR,,0 MVF PREP WRITE	124	ROM ,/0DA,,,,MYQ,,,,,,,,,0 DLA END	164	ROM ,/09A,AWA1,DBIO,,,,,,,RBUS,SEQBIO,,GBOF,U FFX LD A1
0E5	ROM ,/057,,ALUB,,MYQ,,,PM2,,WMEM,,BUSR,GBEX,0 MVSU PREP WRITE	125	ROM ,/0D7,AWA2,BSHR,CALU,MYC,,,,,,,,GCRFNU,0 DLA END	165	ROM ,/170,,,,,,,,,,RBUS,,BUSR,GBOF,0 FFX RESET BSY
0E6	ROM FLAG,/108,,,,,SLQ,,,,,,,,0 EX K1K2K3 TEST K4	126	ROM ,/019,,BINV,,MYC,,,,,,,,,0 ROUT.AFFICH.INCR	166	ROM ,/022,AWCT,ALUB,,,,,,CTP1,,,,,0 TEST NO 3
0E7	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,0 EX K1K2K3N TEST MD	127	ROM ETAT,/07E,,ALUB,CALU,,QYC,,,,,SEQBUS,,,0 DISPLAY INCR	167	ROM ,/023,AWCT,ALUB,,,SLQ,,,CTP1,,,,,0 TEST NO 3
0E8	ROM ,/132,AW15,AMB,CIOR,MYC,,,,,RBUS,SEQBIO,,GFSTOV,0 PAGE FAULT	128	ROM ,/1E9,ARA1,ALUA,,,,,,,,,BUSR,GCRVML,CRLOG DLA END	168	ROM ETAT,/0DE,,,,,,,,,,SEQBUS,,,0 VERIF MAN REG L
0E9	ROM ,/1E9,AEP,ALUA,,,,SYD,,,,SEQBUS,BUSR,,CRRTN RTN 15R2N	129	ROM ,/164,AEP,ASHR,CALU,MYC,,,,,,,,,0 DLN,DRN END	169	ROM FLAG,/09C,APSW,AANDB,,,,,,,,,,GCRFNU,0 TEST TMP-TPM
0EA	ROM ,/110,ATEN,TWOA,CALU,,QYC,,PM2,,,,,,0 TRAP	12A	ROM ,/041,ARA1,ACR,CLUR,MYC,,,,,,,,,0 TEST RB	16A	ROM ,/030,ARA2,ALUA,,,,,,,WBUS,SEQBUS,BUSR,GBTMF,0 FFL LD M1
0EB	ROM ,/073,AEP,ALUA,CBIO,MYQ,QYC,SYD,,,,SEQBUS,,GFPLR,CRRTN RTNA15	12B	ROM FLAG,/034,ARA2,ALUA,,,,,,,,,,GCRFNU,0 TEST DLA	16B	ROM ,/093,AWA2,TWOA,,,,,,,,,,GCRFNU,0 DA
0EC	ROM ,/02E,ATWO,TWOA,CALU,MYC,,,,,,,,,0 CF	12C	ROM ,/043,,BCR,,,,SYD,,,WBUS,,,,0 TEST TMP-TPM	16C	ROM ,/092,AWA1,DADD,,,,,,,,,BUSR,,CRADD DA
0ED	ROM FLAG,/130,AWA0,ALUB,,,,,,,WBUS,,,,0 EX	12D	ROM ,/0CE,,ZERO,,,SRQ,,,,,,,,0 DRA	16D	FTA AWA2,ASHR,0 DAS END
0EE	ROM ,/06B,ATWO,TWOA,CALU,MYC,,,,,,,,,0 CF 15R1	12E	ROM ,/115,ATWO,TWOA,CALU,MYC,,,PM2,,,,,,0 TRAP CORR LONG EX.	16E	ROM ,/090,AEP,TWOA,,MYC,,,,,,,,,0 TEST DLA
0EF	ROM ,/10D,AW15,AMB,,,SLQ,SYD,,,,,,GFSTOV,0 TRAP	12F	ROM ,/115,ATWO,TWOA,CALU,MYC,,,,,,,,,0 TRAP	16F	ROM ,/081,AEP,APB,,MYQ,,,PYD,,,,,,0 TEST DLA
0F0	ROM ,/04E,ATEN,ACR,CALU,MYC,,,,,,,,,0 RER	130	ROM ,/0BF,,,,,,,,,,,,,GFRZ0,0 VERIF MAN. REG M	170	ROM ,/019,,BSHR,CALU,MYC,,,,,,,,,0 ROUT.AFFICH.INCR
0F1	ROM FLAG,/0F7,AEP,ALUA,CALU,MYC,,,,,,,,,0 MVB	131	ROM ,/0CD,,,,MYQ,,,,,,,,,0 DSH	171	ROM ,/152,AEP,ALUA,CBIO,MYC,,SYD,,,,SEQBUS,,,0 DAR* DA
0F2	ROM ,/105,APSW,ALUA,,,SLQ,SP2,,,WMEM,,BUSR,GFSYS,0 TRAP	132	ROM ,/0DC,AWA2,ALUB,,,,,,,,,BUSR,GCRFNU,0 DSH	172	ROM ,/1F7,AEP,ALUA,CBIO,MYC,,SYD,,,,SEQBUS,BUSR,,0 EL
0F3	ROM ,/12C,,DBIO,,,,SYD,,,,SEQBUS,BUSR,,0 RT6C	133	ROM ,/038,,,,MYQ,,,,,,,,,0 TEST Q,CHARG NO.	173	ROM ,/088,AEP,ALUA,CIOR,,GYC,SYD,,,,SEQBUS,,,0 CC
0F4	ROM ,/0EB,ARA1,ALUA,,,,SYD,,,,,BUSR,,0 MVB PREP 1ST AD	134	ROM ,/0CA,AEN,ALUA,CALU,MYC,,,PYD,CTP1,,,,,0 MLRI	174	ROM ,/189,AWA2,TWOA,,,,,,,,,,GCRFNU,0 DS
0F5	ROM ,/0EB,ARA1,ALUA,,,,SYD,,,,,BUSR,GBEX,0 MVUS PREP 1ST AD	135	ROM ,/1D3,AWR2,APB,,,,SYD,PM2,,,,BUSR,,0 MLRI	175	ROM ,/142,AEP,ALUA,CBIO,MYC,,SYD,,,,SEQBUS,,,0 DSR
0F6	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,0 EX K1K2K3K4 TEST MD	136	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA	176	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA
0F7	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 EX 150PC TRAP	137	ROM ,/0C7,AEP,APB,CALU,MYC,,,,CTP1,,,,,0 MLK	177	ROM ,/087,AWA0,ALUB,,MYQ,,,,,,,BUSR,,0 CW CA
0F8	ROM ,/0C8,AEN,ALUA,CALU,MYC,,,PM2,,,,,,0 MLK	138	ROM ,/0C6,AEN,ALUA,,,,,PYD,,,,,,0 ML	178	FTA ARA0,AMB,CRCMP CW CA
0F9	FTA AWR1,BINV,CRLOG C1	139	ROM ,/1C3,ARR2,ALUA,CALU,,QYC,,PM2,,,,BUSR,,0 ML	179	ROM ,/083,AWR1,BCR,CIOR,MYC,,,,,,SEQBUS,,,0 LC
0FA	ROM ,/104,AEP,ALUA,,MYQ,,SP2,,,WMEM,SEQBUS,BUSR,,0 TRAP	13A	ROM ,/0C4,AEN,ALUA,,,,,PYD,,,,,,0 MS	17A	ROM ,/027,,ZERO,,MYQ,SRQ,,,,,,,,0 TEST NO 2
0FB	ROM ,/103,,ALUB,,,,SYD,,,,SEQBUS,,,0 TRAP	13B	ROM ,/1B3,ARCT,ALUA,,,,,PM2,CTP1,WMEM,,BUSR,,0 MS	17B	ROM ,/026,,ALUB,,MYQ,,,PYD,,,,,,0 TEST NO 2
0FC	ROM ,/135,,,,,,SM2,,,,,BUSR,,0 TRAP	13C	ROM ,/0C2,AEN,ALUA,CALU,,QYC,,PYD,,,,,,0 MSRD	17C	ROM ,/1C8,AEP,ALUA,,,,SYD,,,,,BUSR,,0 LC
0FD	ROM ,/101,,BINV,,,,,,,WMEM,,BUSR,,CRLOG C1S	13D	ROM ,/1A3,ARCT,ALUA,,,,,PM2,CTP1,WMEM,,BUSR,,0 MSRD	17D	FTA AWR1,A0B,0 LC END
0FE	ROM ,/1E9,AEP,ALUA,,,,SYD,,,,SEQBUS,BUSR,,0 STORE RESULT	13E	ROM FLAG,/00A,AEP,ALUA,,,,,,,WMEM,,,,0 TEST MEMOIRE NO2	17E	ROM ,/080,AEP,APB,,,,,PYD,,WBUS,,,,0 TEST DLA
0FF	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	13F	ROM ,/053,ARA1,TWOA,,MYC,QYC,SM2,PM2,,WMEM,,,,0 TEST MEMOIRE NO1	17F	ROM ,/18E,AWA0,ZERO,,,,,,,,SEQBIO,,GFETCH,0 TEST DLA

Table 2-4A contd.

ADD.		ADD.	
180	ROM ,/028,,,MYQ,,,,RBUS,,GBOK,O DISPLAY INCR	1C0	ROM ,/1E8,,,,,,,,RBUS,,GBOK,O VERIF MAN. PUPIT
181	ROM ,/020,,CBIO,MYC,,,,,SEQBUS,,O VERIF MAN REGM	1C1	ROM ,/037,AWA1,AMB,,,,,,,,,,O DER MVF REST A1
182	ROM ,/050,,,MYQ,,,,,,,,O ROUT.AFFICH.INCR	1C2	ROM FLAG,/02C,,BSHR,,MYC,QYC,,,,,,GCRFNU,O ROUT.AFFICH.INCR
183	ROM ,/0E9,,,,,,,,,,,,,O ROUT.AFFICH.INCR	1C3	ROM ,/085,ATWO,ACR,,MYC,QYC,,,,,,GFRZO,O TEST NO 2
184	ROM ,/07A,ARR1,ALUA,,,,,,WBUS,,BUSR,GBTME,O WER	1C4	ROM FLAG,/0F1,AWR2,ALUB,,MYQ,,,,,,,,O DER MOVE REST R2
185	ROM ,/188,ARR1,ALUA,,,,,,WBUS,SEQBUS,,,O WER	1C5	ROM ,/188,AWR1,DBIO,CB1O,,,,,,SEQBUS,,,CRRTN RER
186	ROM FLAG,/054,AR15,AMB,,SRQ,,,,,,GCRFNU,O TEST NO 3	1C6	ROM ,/0CC,,CBIO,,QYC,,,,SEQBUS,,O TEST Q,CHARG NO.
187	ROM FLAG,/064,ARCT,AMB,,SLQ,,CTP1,,,GCRFNU,O TEST NO 3	1C7	ROM ,/029,,ALUB,CLUR,MYC,,,,WBUS,,,O TEST Q,CHARG NO.
188	ROM ,/076,AWR2,APB,,,SYD,,,,BUSR,,O RTN	1C8	ROM ,/036,AWA2,AMB,,,,,,,,,,O DER MVF REST A2
189	ROM SNPLA,/100,,DBIO,,,SM2,PYD,,SEQBUS,BUSR,,O RTN	1C9	ROM ,/03B,ATWO,APB,CALU,MYC,,,,,,,,,O DER MOVE M=LGTH
18A	ROM ,/09F,,,,,,,,,,,,,O TEST NO 2	1CA	ROM ,/09F,,,,,,,,,,,,,O TEST DLA
18B	ROM ,/01B,,ZERO,,SRQ,,PM2,,,,,O TEST NO 2	1CB	ROM ,/071,ATWO,ASHR,,MYC,,,,,,,,O TEST DLA
18C	ROM ,/072,ATWO,APB,CALU,MYC,,,,,,,GFENB,O COMPUTE MASK RTNA15	1CC	ROM ,/07F,,BINV,,,QYC,,,,,,,O TEST NO 2
18D	ROM ,/06E,AWAO,BINV,,MYQ,,,,,,,,O COMPUTE MASK RTNA15	1CD	ROM ,/098,ATWO,ASHR,,MYC,QYC,,,,,,GFRZO,O TEST NO 3
18E	ROM FLAG,/024,ARA1,AXB,,,,,,,,,GCRFNU,O TEST DLA	1CE	ROM ,/0AC,AWAO,ALUB,,,,,,,SEQBUS,,GFLOT,O FPP OP/S LD EXP
18F	ROM ,/061,ATWO,APLB1,,MYC,,,,,,,,O TEST RB	1CF	ROM ,/188,AEP,ALUA,,,SYD,,,SEQBUS,,GFLOT,O FPP OP LD EXP
190	ROM ,/019,,BINV,,MYC,,,,,,,,O ROUT.AFFICH.INCR	1D0	ROM ,/02F,,,,,,,,,,,,,O NOT USED
191	ROM ,/1BE,ARAO,AOB,CLUR,MYC,QYC,,,,,,,O MASK STACK PSW RTNA15	1D1	ROM ,/06A,AWR1,AMB,,,SYD,,,,,,O CF.15R2N
192	ROM ,/09F,,,,,,,,,,,,,O TEST MEMOIRE NO1	1D2	ROM FLAG,/07C,,BSHR,,MYC,SRQ,,,,,,GCRFNU,O ROUT.AFFICH.INCR
193	ROM FLAG,/002,,CBIO,MYC,,,,,SEQBUS,,O TEST MEMOIRE NO1	1D3	ROM ,/032,ARAO,,,,,,,,,,GCTLD,O TEST NO 3
194	ROM ,/06A,AWR1,AMB,,,SYD,,,,,GFSTOV,O CF.15R2	1D4	ROM FLAG,/0EA,AWA1,APB,,,SYD,,,SEQBUS,,GMOVE,O MVB WRITE
195	ROM ,/068,APSW,ALUA,,,SP2,,WMEM,,BUSR,,O CF	1D5	ROM FLAG,/0EA,AWA1,APB,,,SYD,,,SEQBUS,,GBEX,O MVUS WRITE
196	ROM ,/0E9,,,,,,,,,,,,,O TEST DLA	1D6	ROM ETAT,/09E,,,,,,,,,SEQBUS,,O TEST Q,CHARG NO.
197	ROM ,/067,AEP,ALUA,,MYQ,,SP2,,WMEM,SEQBUS,BUSR,,O CF	1D7	ROM FLAG,/0D8,,ALUB,CB1O,,QYC,,,WBUS,SEQBUS,,O ROUT.AFFICH.INCR
198	ROM ,/1E9,,ALUB,,,SYD,PYD,,SEQBUS,BUSR,,O CF END	1D8	ROM FLAG,/084,ATEN,AXB,,,,,,,,,GCRFNU,O TEST NO 2
199	ROM ,/05E,,BSHR,CALU,MYC,,,,,,,,O EX	1D9	ROM ,/017,AEP,AMB,,,,PYD,,,,,O TEST NO 2
19A	ROM ,/09F,,,,,,,,,,,,,O TEST NO 3	1DA	ROM ,/09F,,,,,,,,,,,,,O TEST DLA
19B	ROM FLAG,/078,,,MYQ,,,,,,,,O TEST NO 3	1DB	ROM ,/070,ATEN,TWOA,,MYC,,,,,,,,O TEST RB
19C	ROM FLAG,/05C,AEP,AXB,,,SM2,PM2,,,,GCRFNU,O TEST MEMOIRE NO2	1DC	ROM FLAG,/098,,,MYQ,,,,,,,,O TEST NO 3
19D	ROM ,/0FB,,,,,,,,RBUS,,GBOK,O TMP TEST	1DD	ROM ,/078,ATWO,ASHR,,MYC,QYC,,,,,,O TEST NO 3
19E	ROM ,/060,AWAO,BCR,,MYC,,,,,,,,O TEST RB	1DE	ROM FLAG,/00C,,BINV,,,,,,,,,,O TEST NO 1
19F	ROM ,/051,AEP,APB,,MYC,,,PYD,,WBUS,,,,O TEST RB	1DF	ROM ,/049,,ALUB,,,,,,WBUS,,,O VERIF MAN. REG M
1A0	ROM FLAG,/13A,AWAO,ALUB,,MYQ,,PM2,,,,,O MVF TEST LGHT=O	1E0	ROM ,/100,,ALUB,,,,PYD,,,,,O TRAP DLN
1A1	ROM FLAG,/174,ATEN,AXB,,,,,,,,,GCRFNU,O EX TEST IF T2	1E1	ROM FLAG,/195,ARA1,ALUA,,,,,,,,,,O DLN
1A2	ROM ,/09F,,,,,,,,,,,,,O TEST MEMOIRE NO2	1E2	ROM ,/09F,,,,,,,,,,,,,O TEST NO 1
1A3	ROM FLAG,/062,,CBIO,MYC,,,,,SEQBUS,,O TEST MEMOIRE NO2	1E3	ROM ,/021,,BINV,,MYC,,,,,,,,O TEST NO 1
1A4	ROM ,/05A,ARR2,AMB,CALU,MYC,,,,,,,,O MVF LGHT-2 IN M	1E4	ROM FLAG,/01A,,,,,,,PM2,,,,,O TEST NO 2
1A5	ROM ,/12B,AWA1,APB,,MYQ,,SYD,,,,BUSR,,O MVF A1=SOURCE	1E5	ROM ,/033,,,MYQ,,,,,,,,O TEST NO 2
1A6	ROM ,/096,ATWO,ACR,CALU,MYC,,,,WBUS,SEQBIO,,GFKYZO,CRIO TMP T.	1E6	ROM FLAG,/04C,,BSHR,,MYC,,,,,,,GCRFNU,O ROUT.AFFICH.INCR
1A7	ROM FLAG,/058,,,,SLQ,,,WBUS,,GBTMP,O TEST TMP-TPM	1E7	ROM ,/04C,,,MYQ,,,,,,,,O TEST NO 1
1A8	ROM FLAG,/12A,AWA1,AMB,,,SYD,,,SEQBUS,,GMOVE,O MVF WRITE DEST	1E8	ROM ,/016,AEP,APB,CALU,MYC,SRQ,,,,,,,O TEST NO 2
1A9	ROM ,/13A,AWR2,ZERO,,,,,,,,,,O MVF CLEAR LGHT	1E9	ROM FLAG,/074,ATEN,AXB,,,,,,,,,GCRFNU,O TEST NO 2
1AA	ROM ,/09F,,,,,,,,,,,,,O TEST NO 3	1EA	ROM ,/09F,,,,,,,,,,,,,O TEST RB
1AB	ROM ,/013,AWA2,FORA,,,,,,,,,,O TEST NO 3	1EB	ROM ,/033,ATEN,ASHR,CALU,MYC,,,,,,,,,O TEST RB
1AC	ROM FLAG,/052,,ALUB,,,SM2,PM2,,WMEM,SEQBUS,,,O TEST MEMOIRE NO1	1EC	ROM ,/012,ATEN,ALUA,,MYC,QYC,,,,,,,O TEST NO 3
1AD	ROM ,/0B0,ARAO,ALUA,,,SYD,PYD,,,,,O TEST MEMOIRE NO1	1ED	ROM FLAG,/044,ARA2,AMB,,SRQ,,,,,,GCRFNU,O TEST NO 3
1AE	ROM SNPLA,/1FF,AWA1,ALUB,CIOR,MYC,QYC,,PM2,,,SEQBIO,,GFETCH,O TRB	1EE	ROM ,/1E9,AEP,ALUA,,,SYD,,,SEQBUS,,,CRFLO FPP STORE ST EXP
1AF	ROM ,/0B1,AWAO,ALUB,,,SYD,PYD,,,,GFRZO,O TEST MEMOIRE NO1	1EF	ROM FLAG,/010,,,,SLQ,SP2,,WEXM,SEQBUS,BUSR,GBOF,O FPP ST MANTI.
1B0	ROM ,/05B,ATWO,ALUA,CALU,MYC,QYC,,,,,,,O MVF	1F0	ROM ,/100,,ALUB,,,,PYD,,,,,O TRAP DRN
1B1	ROM ,/046,AWAO,BSHR,,MYQ,,,,,,,,O RER	1F1	ROM FLAG,/185,AWA1,ASHR,,SRQ,,,,,,GCRDSR,O DRN
1B2	ROM FLAG,/03C,,BSHR,,MYC,,,,,,,GCRFNU,O ROUT.AFFICH.INCR	1F2	ROM ,/09F,,,,,,,,,,,,,O TEST NO 1
1B3	ROM FLAG,/0B8,ATWO,AOB,,MYC,QYC,,,,,,GFRZO,O TEST NO 1	1F3	ROM ,/07F,,,,,,,,,,,,,O TEST NO 1
1B4	ROM FLAG,/0F2,AWAO,ALUB,,,,PM2,,,,,O MVB TEST LENGTH	1F4	ROM ,/0C1,,,,,SM2,PM2,,SEQBUS,,O TEST MEMOIRE NO2
1B5	ROM FLAG,/10B,AWA2,AMB,,,,,,,,,,O MVB PREPARE A2	1F5	ROM ,/063,,ALUB,,,SYD,PYD,,,,,O TEST MEMOIRE NO2
1B6	ROM ETAT,/0BE,,,,,,,,,SEQBUS,,O VERIF MAN. REG M	1F6	ROM ,/097,,DBIO,,,,,,WBUS,SEQBUS,,O VERIF MAN REG L
1B7	ROM FLAG,/028,,ALUB,,MYQ,,PM2,,WMEM,,BUSR,,O MVB PREPARE WRITE	1F7	ROM ,/0BB,ARR1,ALUA,CALU,MYC,,,,WBUS,,BUSR,GBTMP,O CIO OTR
1B8	ROM ,/162,,ALUB,,,,,,WMEM,,BUSR,,O INT	1F8	ROM ,/036,AEP,ALUA,CALU,MYC,,,,,,,GCTLD,O DER MVB READ
1B9	ROM ,/03A,ARAO,AOB,,,SYD,,RBUS,,BUSR,GBTME,O RER	1F9	ROM ,/007,AWA1,AMB,,,,PP2,,,,GCTLD,O DER MVB WRITE
1BA	ROM ,/09F,,,,,,,,,,,,,O TEST NO 3	1FA	ROM ,/03E,AEP,ALUA,CALU,MYC,,,,,,,GCTLD,O DER MVF READ
1BB	ROM ,/01A,,ZERO,,SRQ,,,,,,,O TEST NO 3	1FB	ROM ,/005,AWA1,APB,,,,PP2,,,,GCTLD,O DER MVF WRITE
1BC	ROM ,/058,ATEN,ALUA,CALU,,QYC,,,WBUS,SEQBIO,,GFETCH,O TEST TMP	1FC	ROM FLAG,/06C,ARA1,AXB,,,SM2,PM2,,,,GCRFNU,O TEST MEMOIRE NO1
1BD	ROM ,/0B3,,,,,SP2,,,SEQBUS,BUSR,GBTMF,O FPP DOUBLE LD 2ND WORD	1FD	ROM FLAG,/0C0,ARAO,ALUA,CALU,MYC,,SYD,PYD,,,,,,O TEST MEMOIRE NO1
1BE	ROM ,/040,,BCR,,MYC,,,,,,,GFKYZO,O TEST RB	1FE	ROM ,/12D,ATWO,TWOA,CALU,MYQ,QYC,,,,,,GCTLD,O PAGE FAULT ML
1BF	ROM FLAG,/014,AEP,AXB,,MYQ,,,,,,,GCRFNU,O TEST RB	1FF	ROM ,/125,ATWO,TWOA,CALU,,QYC,,,,,,GCTLD,O PAGE FAULT

Table 2-4B Control-ROM Microinstruction Listing P856

ADD.		
000	ROM ,/1E8.AR15.ALUB.CALU.MYC.QYC,,,WBUS,,,0	IDLE
001	FTA ,,0	NO JUMP
002	ROM ,/1E9,,,,,SP2.PP2,,,BUSR,,0	INCR P
003	ROM FLAG,/0D0.AZ.ALUA.CALU,,QYC,,,,,,GAEXL.0	NO FLOT PROC
004	ROM FLAG,/1F8.ASYS.TWOA.CALU,,QYC,,,,,,,0	TEST DISPLAY
005	ROM ,/1FB,,,,,SM2.PM2,,,,GFKYZO.0	DECR BEFORE VISU
006	ROM ,/1FF.AEP.ALUA.CALU.MYC,,,,,,,,0	VISUA
007	ROM ,/1FF,.DBIO.CBIO.MYC,,,,,SEQBUS,,0	VISUB
008	FTA AWR1.ALUB.CRLOG	LD
009	ROM ,/1E9,.ALUB,,,SYD.PYD,,,BUSR,,CRLOG	LD P
00A	ROM ,/16D.ASYS.ALUA,,,,,,WBUS,,,0	RCP IN L REG. LR OR RR
00B	ROM ,/1FF.ARA0.ALUA,,MYC,,,,,,,,0	ERRONEOUS START
00C	ROM ,/101.ARR1.ALUA,,,,,,WMEM,,BUSR,,0	ST
00D	ROM ,/101.AEP.ALUA,,,,,,WMEM,,BUSR,,0	ST P
00E	ROM ,/0E6.ATWO.ALUA.CALU.MYC,,,,,,,,0	STD
00F	ROM ,/0E5.ATWO.ALUA.CALU.MYC,,,,,,,,0	STD P
010	ROM FLAG,/1D0.ASYS.ALUA.CALU,,QYC,,,,,,,0	TEST AUTOREST
011	ROM ,/1E9,.ALUB,,,SYD.PYD,,,BUSR,,0	AB
012	ROM FLAG,/0E0.AWA0.ALUB,,,,,,RBUS,,GBOK.0	SAVE P TEST IF L/R R
013	ROM ,/0E0.ARA0.ALUA,,,SYD.PYD,,,,,0	LOAD P AND S LR
014	ROM ,/1D8.ATWO.TWOA.CALU.MYC,,,,,,,,0	INT
015	ROM ,/1D8.ATWO.TWOA.CALU.MYC,,PM2,,,,GFKYZO.0	INT DECR
016	FTA ,,0	FETCH
017	ROM ETAT,/1FE,,,,,,,,,,SEQBUS,,GIDLE.0	VISU IDLE
018	ROM FLAG,/193.AEP.ALUA.CALU.MYC,,,,,,,,GCRVZO.0	SLASLN SAVE P
019	ROM FLAG,/15B.AEP.ALUA.CALU.MYC.SLQ,,,,,,GCRVZO.0	DLADLN SAVE P
01A	ROM FLAG,/17B.ARR1.ALUA.CALU,,QYC,,,,,,,0	SLL SLC
01B	ROM ,/0CE.AWA1.ASHL,,SLQ,,CTP1,,REPEAT,,GCSEL.0	DLL DLC
01C	ROM FLAG,/16B.AEP.ALUA.CALU.MYC,,,,,,,,0	SRA SRN
01D	ROM FLAG,/14B.AEP.ALUA.CALU.MYC.SLQ,,,,,,,0	DRA DRN
01E	ROM ,/0DD.ARR1.ALUA.CALU,,QYC,,,,,,,0	SRL SRC
01F	ROM ,/0CE.AWA1.ASHR,,SRQ,,CTP1,,REPEAT,,GCRDSR.0	DRL DRC
020	FTA AWR1.APB.CRADD	AD
021	ROM ,/1E9.AEP.APB,,,SYD.PYD,,,BUSR,,CRADD	AD P
022	ROM ,/0EC.ARR2.ALUA,,,SYD,,,,BUSR,,0	RT3
023	ROM ,/0E8.ARR2.ALUA,,,SYD,,,,BUSR,,0	RT3S
024	ROM ,/101.ARR1.APB,,,,,,WMEM,,BUSR,,CRADD	AD S
025	ROM ,/101.AZ.APLB1,,,,,,WMEM,,BUSR,,CRADD	IM
026	ROM ,/077.ATWO.TWOA.CALU.MYC,,,,,,,,0	RTN MASTER
027	ROM ,/19E.AW15.AMB,,,SYD,,,,,GFSTOV.0	INT UPDATE A15
028	ROM FLAG,/0D0.AZ.ALUA.CALU,,QYC,,,,,,GAEXL.0	TRAP SLN
029	ROM ,/0E3.ATEN.TWOA.CALU.MYC.QYC,,PYD,,,,,0	SLN
02A	ROM ,/159.ATEN.ACR.CALU.MYC.QYC,,,,,,,0	IPL SET
02B	ROM ,/145.AEP.ALUA,,,SYD,,RBUS,,BUSR.GBOK.0	IPL END
02C	ROM FLAG,/1D2.AWCT.DBIO,,MYQ,,SM2.PM2.CTP1,,SEQBUS,,,0	MLRI
02D	ROM ,/0DC,.ALUB,,,SYD.PYD,,,BUSR,,0	MLRI
02E	ROM ,/134,,CLUR.MYC,,,,,,,GCSEL.0	AUTO RESTART
02F	ROM FLAG,/1C0.APUP.ALUA.CALU,,QYC,,,,,,,0	TEST IPL
030	FTA AWR1.AMB.CRSUB	SU
031	ROM ,/1E9.AEP.AMB,,,SYD.PYD,,,BUSR,,CRSUB	SU P
032	ROM SNPLA,/100,,CBIO.MYC.QYC.SP2.PP2,,SEQBUS,,,0	RT2
033	ROM SNPLA,/100,,CBIO.MYC.QYC,,PP2,,SEQBUS,,,0	RT2S
034	ROM ,/101.ARR1.AMB,,,,,,WMEM,,BUSR,,CRSUB	SU S
035	ROM FLAG,/0BC.AEN.ALUA,,,,,,,,,GCRFNU.0	C2 OR NGR
036	ROM ,/077.ATWO.TWOA.CALU.MYC,,,,,,,,0	RTN USER
037	FTA AWR1.AOB.0	INH.RTN A15 AND LC END
038	ROM FLAG,/0D0.AZ.ALUA.CALU,,QYC,,,,,,GAEXL.0	TRAP SRN
039	ROM ,/0DE.ARR1.ALUA.CALU,,QYC,,,,,,,0	SRN
03A	ROM FLAG,/1C4,.BSHR.CALU.MYC,,,,,,,GCRFNU.0	WAIT IO
03B	ROM ETAT,/1FF.AEP.ALUA,,,SYD,,,,,,0	WAIT IO
03C	ROM FLAG,/1C2.AWCT.DBIO,,,SP2.PM2.CTP1,,SEQBUS,,GCRVML.0	ML LOOP
03D	ROM ,/0DC,.ALUB,,,SYD.PYD,,,BUSR,,0	ML END
03E	ROM ,/15F.ATWO.ACR,,,,PYD,,,,,0	IPL
03F	ROM FLAG,/1B0,,,,SLQ,,,RBUS,,GBOK.0	TEST LOADM

ADD		
040	FTA AWR1.AANDB.CRLOG	AN
041	ROM FLAG,/1C8.APSW.AANDB.CALU.MYC,,,,,,,BUSR,,0	HLT.RIT.INH.RTA15
042	ROM ,/0FD,,CBIO.MYC,,PP2,,SEQBUS,,,0	RT5
043	ROM ,/0F5,,CBIO.MYC,,PP2,,SEQBUS,,,0	RT5S
044	ROM ,/101.ARR1.AANDB,,,,,,WMEM,,BUSR,,CRLOG	AN S
045	ROM ,/101,.ZERO,,,,,,WMEM,,BUSR,,0	CM
046	FTA ARR1.AANDB.CRLOG	TM
047	ROM ,/1E9,,,,,,,,,,BUSR,,CRFLO	FPP WAIT EXEC.
048	ROM FLAG,/1A6.ARA1.DIVALU,,,,,,,,,GCRFNU.0	DIV TEST COR.REM.
049	ROM FLAG,/140.ARA1.AXB.CALU.MYQ.QYC,,,,,,,0	DIV TEST QUOT.
04A	ROM ,/0E2.AEP.AMB.CALU.MYC,,,,,,,,0	SLN
04B	ROM FLAG,/1B5.AWR1.TWOA,,,,PP2,,,,,0	SLN LOOP
04C	ROM FLAG,/1B2.ARCT.ALUA,,,SP2.PM2.CTP1.WMEM.SEQBUS,,,0	MS LOOP
04D	ROM ,/1E9,.ALUB,,,SYD.PYD,,,BUSR,,0	MS END
04E	ROM FLAG,/1A8,,,,SLQ,,,,,,GFRZO.0	TEST READM
04F	ROM ,/172,.DBIO.CBIO.MYC,,,,WMEM.SEQBUS,,GBCP.0	LOAD M
050	FTA AWR1.AOB.CRLOG	OR
051	ROM FLAG,/188.APSW.AOB.CALU.MYC,,,,,,,,0	END SMD LKM
052	ROM ,/0EC,.DBIO,,,SYD.PP2,,SEQBUS.BUSR,,0	RT4
053	ROM ,/0E8,.DBIO,,,SYD.PP2,,SEQBUS.BUSR,,0	RT4S
054	ROM ,/101.ARR1.AOB,,,,,,WMEM,,BUSR,,CRLOG	OR S
055	ROM ,/0A8.AWA0.BSHR,,MYQ,,,,,,,,0	INR
056	ROM FLAG,/1A0,,,,SLQ,,,,,,,0	TEST READST
057	ROM FLAG,/190.ASYS.TWOA.CALU,,QYC,,,,,,GBCP.0	READ M
058	ROM ,/1A6.AWA1.DIVALU,,,,,,,,,,0	DIV CORR.REM.
059	ROM FLAG,/140.ARA1.AXB.CALU.MYQ.QYC,,,,,,,0	DIV TEST QUOT.
05A	ROM ,/0E2.AEP.AMB.CALU.MYC,,,,,,,,0	SRN
05B	ROM FLAG,/1A5.AWR1.ASHR,,SRQ,,PP2,,,,GCRDSR.0	SRN LOOP
05C	ROM FLAG,/1A2.ARCT.ALUA,,,SM2.PM2.CTP1.WMEM.SEQBUS,,,0	MSRD LOOP
05D	ROM ,/0BE,.ALUB,,MYQ,,SYD.PYD,,,,,0	MSRD END
05E	ROM ,/198,,,,SLQ,,,,,,,0	PUP
05F	ROM ,/1FF.APSW.ALUA.CALU.MYC,,,,,,,,0	READ STATUS
060	FTA AWR1.AXB.CRLOG	XR
061	ROM ,/047.APSW.ALUA.CALU.MYC,,SP2,,,,,GFSYS.0	INT
062	ROM ,/0FC,,CBIO.MYC,,,,,SEQBUS,,,0	RT7
063	ROM ,/0F4,,CBIO.MYC,,,,,SEQBUS,,,0	RT7S
064	ROM ,/101.ARR1.AXB,,,,,,WMEM,,BUSR,,CRLOG	XR S
065	FTA AWA2.BSHR.0	DLN END
066	FTA ARR1.AXB.CRLOG	TNM
067	ROM ,/180.AWA0.ALUB,,MYQ,,,,,,,,0	PUP
068	ROM FLAG,/0D0.AZ.ALUA.CALU,,QYC,,,,,,GAEXL.0	MVF 15R2 TRAP
069	ROM ,/05F.ARR2.ALUA.CALU,,QYC,,PYD,,,,,0	MVF 15R2N
06A	ROM ,/0D6.AWA0.ALUB,,,,PM2,,,,,0	DLNDRN END
06B	ROM FLAG,/195.AWA1.ASHL,,SLQ,,PP2,,,,,0	DLN LOOP
06C	ROM ,/0E4.AWR1.TWOA,,,,,CTP1,,REPEAT,,,0	SLA
06D	ROM FLAG,/1D7.AWA0.ALUB,,,,,,,,,,0	SLN
06E	ROM ,/1FF,.DBIO.CBIO.MYC,,SP2.PP2,,SEQBUS,,,0	READ M PUP A
06F	ROM ,/1FF,.DBIO.CBIO.MYC,,,,,SEQBUS,,,0	READM PUP B
070	ROM SNPLA,/100.AR1215,,,,,,,,,,GCTLD.0	SH DSH
071	ROM ,/18F.ARA2.ALUA.CALU,,QYC,,,,,,GFKYZO.0	DSH
072	ROM ,/1AD,.DBIO,,,SYD,,,SEQBUS.BUSR,,0	RT6
073	ROM ,/1AC,.DBIO,,,SYD,,,SEQBUS.BUSR,,0	RT6S
074	FTA ARR1.ALUA.CRLOG	SHZERO
075	ROM ,/0DC.ARA2.ALUA,,,,,,,,BUSR.GCRFNU.0	DSHZERO
076	ROM ,/092.AWA1.DSUB,,,,,,,,BUSR,,CRSUB	DS
077	ROM ,/1C5.ATWO.ALUA.CALU.MYC,,,,,,,,0	WAIT
078	ROM ,/0C7.AEP.ALUA.CALU.MYC,,,CTP1,,,,,0	ML
079	ROM ,/150.ATWO.ASHR.CALU,,QYC,,,,,,GBTMM.0	TL
07A	ROM ,/195.AWA1.ASHL,,SLQ,,,,,,,,0	DRN
07B	ROM FLAG,/185.AWA1.ASHR,,SRQ,,PP2,,,,GCRDSR.0	DRN LOOP
07C	ROM ,/0C5.AEP.ALUA.CALU.MYC,,,CTP1,,,,,0	MS
07D	ROM ,/148.ATWO.ASHR.CALU,,QYC,,,WEXM,,,GBOM.0	TS
07E	ROM ,/0C3.AEP.ALUA.CALU.MYC,,,CTP1,,,,,0	MSRD
07F	ROM FLAG,/1F4,.BCR.CALU,,QYC,,,,,,GCRFNU	PUP

ADD.		
080	ROM ,/008,.ALUB,,,SYD,,,,,,0	CIO OTR
081	ROM ,/100,,,,,,,,,,,,,0	GO TO TRAP (WMP)
082	ROM ,/1B2.AWA2.DBIO,,,,,,RBUS.SEQBIO,,,0	FFX LD A2
083	ROM SNPLA,/100.ARR2.ALUA.CALU.MYC.QYC,,,,,,,,0	RT1
084	ROM ,/18B.AWR1.TWOA,,,,,CTP1,,REPEAT,,,0	SLL
085	ROM ,/18B.AWR1.ASHL,,SLQ,,CTP1,,REPEAT,,GCSEL.0	SLC
086	ROM ,/169,,,,,,,,CTP1,,,,,0	RT1D
087	ROM ,/179.ARR2.ALUA.CALU,,QYC,,,,,,GCTLD.0	RT1D
088	ROM FLAG,/0B3,.ZERO.CALU.MYC,,,,,,BUSR.GBTMF.0	FPP LOAD
089	ROM ,/0BA.AWA1.ZERO.CALU,,QYC,,,,,,GCRVZO.0	MUL-RESET PART.PROD
08A	ROM ,/130.ARA0.ALUA,,,,,,WBUS,,BUSR,,0	EX T1 OR T3
08B	ROM FLAG,/0D0.AZ.ALUA.CALU,,QYC,,,,,,GAEXL.0	EX T2 TRAP
08C	ROM FLAG,/0AB.ATEN.ACR.CALU.MYC.QYC,,,WEXM,,,GBOF.0	FPP STORE
08D	ROM ,/1A8,,,,,,,,,,SEQBUS,,GFRZO.0	LOAD M
08E	ROM FLAG,/1D4.AEP.ALUA,,,SP2,,,SEQBUS,,GCRFNU.0	IPL LOOP
08F	ROM ,/159,.BSHR.CALU.MYC.SLQ,,,,,,,0	IPL LOOP
090	ROM ,/1AA.ATEN.ACR.CALU.MYC,,,,,,,,0	INR
091	ROM SNPLA,/103.AEP.ALUA,,,SYD,,,,,,0	EX BSYZO.RESTORE S
092	ROM ,/1ED.AEP.ALUA.CALU.MYC,,,,RBUS.SEQBIO,,GFETCH.0	K=RCP L/RR
093	ROM SNPLA,/100.AEP.ALUA.CALU.MYC.QYC,,,,,,,0	RT1P
094	ROM ,/18B.AWR1.ASHR,,,,,CTP1,,REPEAT,,GCRDSR.0	SRA
095	ROM FLAG,/1C7.AWA0.ALUB,,,,,,,,,,0	SRN
096	ROM SNPLA,/100.ARCT.ALUA.CALU.MYC,,,,,,,,0	RT1D END
097	ROM ,/169.AEP.ALUA.CALU,,QYC,,,CTP1,,,,,0	RT1DP
098	ROM FLAG,/0B3,.ZERO.CALU.MYC,,,,,,BUSR.GBTMF.0	FPP OPER
099	ROM ,/0A5.ARA2.TWOA.CALU,,QYC,,,,,,,0	DIV
09A	ROM ,/095.ARA1.ALUA,,,,,,WBUS,,BUSR.GBTMF.0	FFL
09B	ROM ,/0FE.AWR2.ALUB,,,,,,,,,,0	DLN,DRN END
09C	ROM FLAG,/0B3.ATWO.ASHR.CALU.MYC,,,,,,BUSR.GBTMF.0	FPP OP/S
09D	ROM ,/143.AEP.ALUA.CLUR.MYC,,SP2,,WMEM.SEQBUS.BUSR.GCSEL.0	INT
09E	ROM ,/160.AEP.ALUA.CALU.MYC,,,,,,,GFLOT.0	FFX
09F	ROM ,/09B,,,,,,,,RBUS,,GBOF.CRFLO	FFX UPD CR
0A0	ROM ,/105,.ZERO,,,SYD,,,,,GFSYS.0	IPL
0A1	ROM ,/1E9.AEP.APB,,,SYD.PYD,,,BUSR,,0	RF
0A2	ROM ,/0E7.ATWO.ALUA.CALU.MYC,,,,,,,,0	RT3B
0A3	ROM SNPLA,/100.ARR2.ALUA.CALU,,QYC.SYD,,,,,,0	RT3C
0A4	ROM ,/0DB.AWA1.ASHL,,SLQ,,CTP1,,REPEAT,,,0	DLA
0A5	ROM FLAG,/01F.ATWO.ALUA,,,,PYD,,,,,0	DLN
0A6	ROM ,/155,.BSHR.CALU.MYC,,PM2,,,,,0	IPL LOOP
0A7	ROM ,/0CB.AEP.ALUA.CALU,,QYC,,,,,,,0	RT3BM
0A8	ROM ,/08D,,,,,,,,,,BUSR.GBEX.0	EL
0A9	ROM ,/08E,,CBIO,,QYC.SP2,,,SEQBUS.BUSR,,0	DA
0AA	ROM ,/14F,.BSHR.CALU.MYC,,,,,,,,0	IPL LOOP
0AB	ROM ,/152,,CBIO.MYC,,SP2.PP2,,SEQBUS,,,0	DAK
0AC	ROM ,/101.ARR1.ALUA,,,,,,WMEM,,BUSR.GBEX.0	ES
0AD	ROM ,/094.AWA2.APB.CALU.MYQ.QYC,,,,,,,0	DA
0AE	ROM ,/1E9.AEP.ALUA,,,SYD,,,SEQBUS,,,0	TL END
0AF	ROM FLAG,/150,,,,SLQ.SP2,,,SEQBUS.BUSR.GBTMM.0	TL LOOP
0B0	ROM ,/14D,.BSHR.CALU.MYC,,,,,,,,0	IPL LOOP
0B1	ROM ,/1E9.AEP.AMB,,,SYD.PYD,,,BUSR,,0	RB
0B2	ROM FLAG,/170.AIPL.APB.CALU.MYC,,,,WMEM,,,0	IPL NEW QUART
0B3	ROM SNPLA,/100,,,,,,PP2,,,,,0	RT2C
0B4	ROM ,/0D2.AWA1.ASHR,,SRQ,,CTP1,,REPEAT,,GCRDSR.0	DRA
0B5	ROM FLAG,/00F.ATWO.ALUA,,,,PYD,,,,,0	DRN
0B6	ROM ,/1E9.AEP.ALUA,,,SYD,,,SEQBUS,,0	TS END
0B7	ROM FLAG,/148,,,,SLQ.SP2,,WEXM.SEQBUS.BUSR.GBOM.0	TS LOOP
0B8	ROM FLAG,/0D0.AZ.ALUA.CALU,,QYC,,,,,,GAEXL.0	TRAP
0B9	ROM ,/08A,,CBIO,,QYC.SP2,,,SEQBUS.BUSR,,0	DS
0BA	ROM ,/1E9.AW15.DBIO,,,,,,,,SEQBUS,,,0	IPL END
0BB	ROM ,/142,,CBIO.MYC,,SP2.PP2,,SEQBUS,,,0	DSK
0BC	ROM ,/130,.BSHR,,,SYD,,,SEQBUS,,,0	INT
0BD	ROM ,/08B.AWA2.AMB.CALU.MYQ.QYC,,,,,,,0	DS
0BE	ROM ,/09E.AZ.APLB1.CALU.MYC,,,,,,BUSR.GCRVML.CRADD	DIV COR.QUOT
0BF	ROM ,/1E9.AWA2.ALUB,,,,,,,,,BUSR.GCRVML.CRLOG	DIV QUOT OK

Table 2-4B contd.

ADD		ADD.		ADD.	
0C0	FTA AWR1,ALUB,0 LDK	100	ROM ,/03F,,,,,,,,,,,,GFSYS,0 VERIF MAN. PUPIT	140	ROM ,/07E,,,,,,,,,RBUS,,GBOK,0 TEST MAN REG M
0C1	ROM ,/1E9,,ALUB,,,,SYD,PYD,,,,BUSR,,0 LDK F	101	ROM ,/19A,ARA0,ALUA,,MYQ,,,PYD,,,,BUSR,,0 DLN,DRN END	141	ROM ,/1E9,AWR2,AMB,,,,,,,,,BUSR,GFSTOV,0 UPD STK MSRD
0C2	ROM ,/135,,BCR,,,,,,,,,BUSR,GFPLR,0 AR PAF INT	102	ROM ,/0EC,ARR2,APB,,,,SYD,,,,,DUSR,,0 RT5	142	ROM ,/1CF,AWR1,ZERO,,,,,,,,,BUSR,,0 NGR
0C3	ROM ,/0EE,,,CBIO,MYC,,,PP2,,,SEQBUS,,,0 RT5C	103	ROM ,/1AD,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT7	143	ROM ,/101,AZ,AMB,,,,,,,WMEM,,BUSR,,CRSUB C2
0C4	ROM ,/04F,AWA2,APB,,,,,,,,,,,0 MVF DEST IN A2	104	ROM ,/0D3,,,CBIO,MYC,,,,,RBUS,SEQBIO,,,0 TMP TEST RD KEYS	144	ROM ,/188,,ALUB,,,,,,,WBUS,SEQBIO,,,CRIO CIO OTR
0C5	ROM ,/1E9,ARA0,ALUA,,,,SYD,PYD,,,,BUSR,,0 MOVE END SLN REST.P	105	ROM FLAG,/0D4,ARA0,ALUA,,,,,,,,,,GCRFNU,0 TEST RETURN	145	ROM ,/0B7,ARA2,ALUA,CALU,,QYC,,,,,,,,0 MU- MUL.TOR IN Q REG.
0C6	ROM ,/088,ARR1,ALUA,CALU,,QYC,,,,,,,,0 CA	106	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 EX OPER T4T7 TRAP	146	ROM FLAG,/01C,ATWO,APB,,MYC,,,,,,,,,GCRFNU,0 TEST NO 1
0C7	FTA ARR1,AMB,CRCMP CA	107	ROM ,/066,,BSHR,CALU,MYC,,,,,,,,,,0 EX OPER T1,T2,T3	147	ROM ,/018,,,,,SLQ,,,,,,,,0 TEST NO 1
0C8	ROM ,/086,ARR1,ACR,CLUR,MYC,,,,,,,BUSR,GBCH,0 LC	108	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 MVB 15R2FU TRAP	148	ROM ,/0B6,AWA1,MULTI,,,SRQ,,,CTP1,,REPEAT,,,0 MU- ALGORITHM
0C9	FTA AWR1,BCR,0 ECR	109	ROM ,/04B,ARR2,ALUA,,,,,PYD,,,,,,0 MVB 15R2FUN	149	ROM ,/0B5,AWA1,MULTI,,MYQ,,,,,,,,,GMULTI,0 MULTI 16TH PASS
0CA	ROM ,/1E9,,DBIO,,,,SYD,PYD,,,SEQBUS,BUSR,,0 AR PAF INT TRAP	10A	ROM ,/0E8,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT5S	14A	ROM ,/0B4,AWA2,BSHR,,,,,,,,,,,,0 MULTI STORE LSB OF PROD.
0CB	ROM ,/132,,,,,,,,,,,,GFSYS,0 AR END	10B	ROM ,/1AC,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT7S	14B	ROM ,/0D7,ARA2,ALUA,,,,,,,,,,,GCRFNU,0 MULTI UPDATE FNU
0CC	ROM ,/101,ARR1,ALUA,,,,,,,WMEM,,BUSR,GBCH,0 SC	10C	ROM ,/04A,ATWO,ALUA,CALU,MYC,QYC,,,,,,,,,0 MVB LENGTH>0	14C	ROM ,/0AD,,ALUB,,,SRQ,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP SINGLE RD MA
0CD	ROM ,/13D,,BSHR,,,,SYD,,,,,,GFKYZ0,0 PAGE FAULT AR	10D	ROM ,/1E9,ARA0,ALUA,,,,SYD,PYD,,,,BUSR,,0 MVB LENGTH=0	14D	ROM ,/042,,,,,,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP OP/S DOUBLE READ M1
0CE	ROM FLAG,/128,ARA0,APB,CLUR,MYC,SLQ,,,,,,,,0 EX T8N TEST K1	10E	ROM ,/1EB,ARA0,AMB,,,,,PYD,,,,,,0 INT MOVE	14E	ROM ,/053,ATWO,ASHR,,MYC,QYC,SM2,PM2,,WMEM,,,,0 T. MEMOIRE NO1
0CF	ROM ,/16E,,,CIOR,MYC,QYC,,,,WBUS,SEQBIO,,GFFTCH,0 EX LOAD K	10F	ROM ,/125,ATWO,TWOA,CALU,,QYC,,,,,,,,,0 PAF MOVE	14F	ROM ,/06C,AWA1,ALUB,,,,SM2,PM2,,,,,,0 TEST MEMOIRE NO1
0D0	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	110	ROM ,/0DF,,,,,,,,,,,,GFRZ0,0 VERIF MAN. REG L	150	ROM ,/09F,,,,,,,,,,,,,GFRZ0,0 TEST Q,CHARG.NO
0D1	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	111	ROM SNPLA,/100,ARR2,APB,CALU,,QYC,SYD,,,,,,,0 RT5C	151	ROM ,/0A1,AWA1,BCR,,MYC,QYC,,,,,,,,,0 TEST DLA
0D2	ROM ,/125,AWR2,ALUB,,,,,,,,,,,,0 PAGE FAULT ML	112	ROM ,/12C,ARR2,APB,,,,SYD,,,,,BUSR,,0 RT7C	152	ROM FLAG,/030,,,,MYQ,,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP READ LAST MA
0D3	ROM SNPLA,/100,,DBIO,CBIO,,QYC,SYD,PP2,,,SEQBUS,,,0 RT4C	113	ROM SNPLA,/100,AEP,ALUA,CBIO,MYC,QYC,SYD,,,,SEQBUS,,,0 RT3	153	ROM ,/173,ARA0,TWOA,,,,SYD,,,,,,,0 FPP OPS WAIT.LD S WITH 1ST AD.
0D4	ROM FLAG,/118,AWA2,AMB,CBIO,MYC,,SYD,,,,SFQBUS,,GMOVE,0 MVF READ	114	ROM ,/048,AWA2,APB,CBIO,MYC,,SYD,,,,SEQBUS,,GMOVE,0 MVB READ	154	ROM ,/010,,BINV,CALU,,QYC,SP2,,,WEXM,SEQBUS,BUSR,GBOF,0 FPP/S SI.
0D5	ROM ,/056,AWA1,APB,,,,,,,,,,,0 MVF REST A1	115	ROM ,/056,AWA2,APB,,,,,,,,,,,0 MVB CORRECT A2	155	ROM ,/010,,,,,,SP2,,,WEXM,SEQBUS,BUSR,GBOF,0 FPP/S DOUBLE ST M1
0D6	ROM FLAG,/120,,,,,SLQ,,,,,,,,0 EX K1 TEST K2	116	ROM ,/0AE,ATEN,ALUA,,MYC,,,,,,,,GFRZ0,0 TEST DLA	156	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA
0D7	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,,0 EX K1N TEST MD	117	ROM SNPLA,/100,,,CBIO,MYC,QYC,,,,,SEQBUS,,,0 RT3S	157	ROM ,/0A7,ARA0,AOB,,,,SYD,,,RBUS,,BUSR,GBTMP,0 INR
0D8	FTA ARR1,AMB,CRCMP CW	118	ROM ,/0EC,AWR2,APB,,,,SYD,,,,,BUSR,,0 RT3B UPDATE STACK	158	ROM ,/0A6,AWR1,DBIO,CIOR,MYC,,,,,,SEQBUS,,GCSEL,CRIO INR
0D9	FTA AEP,AMB,CRCMP CWP	119	ROM ,/1F3,AWR2,AMB,,,,,,,,,,,GFSTOV,0 STD	159	ROM ,/188,AWR1,AOB,,,SLQ,,,,,,,,,0 INR TST SST
0DA	ROM ,/124,APSW,ALUA,CALU,MYC,,,,,,,,,GFSYS,0 PAGE FAULT	11A	ROM ,/1F2,AWR2,AMB,,,,,,,,,,,GFSTOV,0 STDP	15A	ROM ,/0A4,AWA1,ALUA,,,,,,,,,,,GCRVZ0,0 DIV MEMO SIGNE DIVD
0DB	ROM ,/122,AR15,ALUA,,,,SYD,,,WEXM,,BUSR,GBOM,0 PAGE FAULT	11B	ROM ,/1E9,ARR1,ALUA,,,,,,,,,BUSR,GCRVML,CRLOG SLA CR	15B	ROM ,/0A3,AWA1,DIVSH,,,SLQ,,,CTP1,,,,GCRDSR,0 DIV 1ST PASS
0DC	ROM ,/08C,ARR1,ALUA,CLUR,MYC,,,,,,,BUSR,GBCH,0 CC	11C	ROM FLAG,/1B5,ARR1,ALUA,,,,,,,,,,,,0 SLN TEST NORM	15C	ROM ,/0A2,AWA1,DIVSH,,,SLQ,,,CTP1,,REPEAT,,,0 DIV PROCESS
0DD	ROM ,/11D,,ALUB,,MYQ,,SM2,,,WMEM,SEQBUS,BUSR,,0 PAGE FAULT	11D	ROM ,/13A,AWR2,BSHR,,,,,,,,,,,,0 SLN RESULT	15D	ROM FLAG,/1B6,AWA1,DIVALU,,,SLQ,,,,,,,GCRFNU,0 DIV 16TH PASS
0DE	ROM FLAG,/118,,,,,SLQ,,,,,,,,0 EX K1K2 TEST K3	11E	ROM ,/1EC,AW1215,DBIO,,,,,,,RBUS,SEQBIO,,,0 KEYS IN (RCP) LR	15E	ROM ,/0A0,ATWO,APB,CLUR,MYC,,,PYD,,,,,,0 TEST DLA
0DF	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,,0 EX K1K2N TEST MD	11F	ROM ,/1FF,AR1215,ALUA,,MYC,,,,,,,,GFKYZ0,0 LOAD OR READ R	15F	ROM ,/091,AWA2,BCR,,,SRQ,,,,,,,,,0 TEST DLA
0E0	ROM ,/07B,,ALUB,,,,SYD,,,,,,,0 WER	120	ROM ,/009,,,,,,,,,RBUS,,,GBOK,0 VERIF MAN. REG L	160	ROM ,/039,,,,,,,,,RBUS,,,GBOK,0 TEST Q,CHARG NO.
0E1	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	121	ROM FLAG,/1A5,ATEN,TWOA,CALU,MYC,,,PYD,,,,,,0 SRN	161	FTA AWA2,ALUB,0 DIV ST CORTD QUOT
0E2	ROM ,/117,ATWO,APB,CALU,MYC,,SM2,,,WEXM,SEQBUS,BUSR,GBOM,0 PAF	122	ROM ,/18B,AWR1,ASHR,,,SRQ,,,CTP1,,REPEAT,,GCRDSR,0 SRLC	162	ROM ,/09F,,,,,,,,,,,,,,0 TMP TEST
0E3	ROM ,/0ED,,,CBIO,MYC,,,,,,SEQBUS,,,0 RT7C	123	FTA ARA1,ALUA,CRLOG ML DSH	163	ROM ,/033,AZ,ALUA,,MYC,,,,,,,,,,0 TMP TEST
0E4	ROM ,/057,,ALUB,,MYQ,,,PM2,,WMEM,,BUSR,,0 MVF PREP WRITE	124	ROM ,/0DA,,,,MYQ,,,,,,,,,,0 DLA END	164	ROM ,/09A,AWA1,DBIO,,,,,,,RBUS,SEQBIO,,GBOF,0 FFX LD A1
0E5	ROM ,/057,,ALUB,,MYQ,,,PM2,,WMEM,,BUSR,GBEX,0 MVSU PREP WRITE	125	ROM ,/0D7,AWA2,BSHR,CALU,MYC,,,,,,,,GCRFNU,0 DLA END	165	ROM ,/17D,,,,,,,,,,RBUS,,BUSR,GBOF,0 FFX RESET BSY
0E6	ROM FLAG,/108,,,,,SLQ,,,,,,,,0 EX K1K2K3 TEST K4	126	ROM ,/019,,BINV,,MYC,,,,,,,,,,0 ROUT.AFFICH.INCR	166	ROM ,/022,AWCT,ALUB,,,,,,CTP1,,,,,0 TEST NO 3
0E7	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,,0 EX K1K2K3N TEST MD	127	ROM ETAT,/07E,,ALUB,CALU,,QYC,,,,,SEQBUS,,,0 DISPLAY INCR	167	ROM ,/023,AWCT,ALUB,,,SLQ,,,CTP1,,,,,0 TEST NO 3
0E8	ROM ,/132,AW15,AMB,CIOR,MYC,,,,,RBUS,SEQBIO,,GFSTOV,0 PAGE FAULT	128	ROM ,/1E9,ARA1,ALUA,,,,,,,,,BUSR,GCRVML,CRLOG DLA END	168	ROM ETAT,/0DE,,,,,,,,,,,SEQBUS,,,0 VERIF MAN REG L
0E9	ROM ,/1E9,AEP,ALUA,,,,SYD,,,,SEQBUS,BUSR,,CRRTN RTN 15R2N	129	ROM ,/164,AEP,ASHR,CALU,MYC,,,,,,,,,,0 DLN,DRN END	169	ROM FLAG,/09C,APSW,AANDB,,,,,,,,,,,GCRFNU,0 TEST TMP-TPM
0EA	ROM ,/110,ATEN,TWOA,CALU,,QYC,,PM2,,,,,,0 TRAP	12A	ROM ,/041,ARA1,ACR,CLUR,MYC,,,,,,,,,,0 TEST RB	16A	ROM ,/030,ARA2,ALUA,,,,,,,WBUS,SEQBUS,BUSR,GBTMF,0 FFL LD M1
0EB	ROM ,/073,AEP,ALUA,CBIO,MYQ,QYC,SYD,,,,SEQBUS,,GFPLR,CRRTN RTNA15	12B	ROM FLAG,/034,ARA2,ALUA,,,,,,,,,,,GCRFNU,0 TEST DLA	16B	ROM ,/093,AWA2,TWOA,,,,,,,,,,,GCRFNU,0 DA
0EC	ROM ,/02E,ATWO,TWOA,CALU,MYC,,,,,,,,,,0 CF	12C	ROM ,/043,,BCR,,,,SYD,,,WBUS,,,,0 TEST TMP-TPM	16C	ROM ,/092,AWA1,DADD,,,,,,,,,,BUSR,,CRADD DA
0ED	ROM FLAG,/130,AWA0,ALUB,,,,,,,WBUS,,,,0 EX	12D	ROM ,/0CE,,ZERO,,,SRQ,,,,,,,,,0 DRA	16D	FTA AWA2,ASHR,0 DAS END
0EE	ROM ,/06B,ATWO,TWOA,CALU,MYC,,,,,,,,,,0 CF 15R1	12E	ROM ,/115,ATWO,TWOA,CALU,MYC,,,PM2,,,,,,0 TRAP CORR LONG EX.	16E	ROM ,/090,AEP,TWOA,,MYC,,,,,,,,,,0 TEST DLA
0EF	ROM ,/10D,AW15,AMB,,,SLQ,SYD,,,,,,GFSTOV,0 TRAP	12F	ROM ,/115,ATWO,TWOA,CALU,MYC,,,,,,,,,,0 TRAP	16F	ROM ,/081,AEP,APB,,MYQ,,,PYD,,,,,,0 TEST DLA
0F0	ROM ,/04E,ATEN,ACR,CALU,MYC,,,,,,,,,,0 RER	130	ROM ,/0BF,,,,,,,,,,,,GFRZ0,0 VERIF MAN. REG M	170	ROM ,/019,,BSHR,CALU,MYC,,,,,,,,,,0 ROUT.AFFICH.INCR
0F1	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	131	ROM ,/0CD,,,,MYQ,,,,,,,,,0 DSH	171	ROM ,/152,AEP,ALUA,CBIO,MYC,,SYD,,,,SEQBUS,,,0 DAR* DA
0F2	ROM ,/105,APSW,ALUA,,,SLQ,SP2,,,WMEM,,BUSR,GFSYS,0 TRAP	132	ROM ,/0DC,AWA2,ALUB,,,,,,,,,BUSR,GCRFNU,0 DSH	172	ROM ,/1F7,AEP,ALUA,CBIO,MYC,,SYD,,,,SEQBUS,BUSR,,0 EL
0F3	ROM ,/12C,,DBIO,,,,SYD,,,,SEQBUS,BUSR,,0 RT6C	133	ROM ,/038,,,,MYQ,,,,,,,,,0 TEST Q,CHARG NO.	173	ROM ,/088,AEP,ALUA,CIOR,,QYC,SYD,,,,SEQBUS,,,U CC
0F4	ROM ,/0EB,ARA1,ALUA,,,,SYD,,,,,BUSR,,0 MVB PREP 1ST AD	134	ROM ,/0CA,AEN,ALUA,CALU,MYC,,,PYD,CTP1,,,,,0 MLRI	174	ROM ,/189,AWA2,TWOA,,,,,,,,,,,GCRFNU,0 DS
0F5	ROM ,/0EB,ARA1,ALUA,,,,SYD,,,,,BUSR,GBEX,0 MVUS PREP 1ST AD	135	ROM ,/1D3,AWR2,APB,,,,SYD,PM2,,,,BUSR,,0 MLRI	175	ROM ,/142,AEP,ALUA,CBIO,MYC,,SYD,,,,SEQBUS,,,0 DSR
0F6	ROM FLAG,/0F8,,BCR,CALU,,QYC,,,,,,,,,0 EX K1K2K3K4 TEST MD	136	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA	176	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA
0F7	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 EX 150PC TRAP	137	ROM ,/0C7,AEP,APB,CALU,MYC,,,,CTP1,,,,,0 MLK	177	ROM ,/087,AWA0,ALUB,,MYQ,,,,,,,BUSR,,0 CW CA
0F8	ROM ,/0C8,AEN,ALUA,CALU,MYC,,,PM2,,,,,,0 MLK	138	ROM ,/0C6,AEN,ALUA,,,,,PYD,,,,,,0 ML	178	FTA ARA0,AMB,CRCMP CW CA
0F9	FTA AWR1,BINV,CRLOG C1	139	ROM ,/1C3,ARR2,ALUA,CALU,,QYC,,PM2,,,,BUSR,,0 ML	179	ROM ,/083,AWR1,BCR,CIOR,MYC,,,,,,SEQBUS,,,0 LC
0FA	ROM ,/104,AEP,ALUA,,MYQ,,SP2,,,WMEM,SEQBUS,BUSR,,0 TRAP	13A	ROM ,/0C4,AEN,ALUA,,,,,PYD,,,,,,0 MS	17A	ROM ,/027,,ZERO,,MYQ,SRQ,,,,,,,,,0 TEST NO 2
0FB	ROM ,/103,,ALUB,,,,SYD,,,,SEQBUS,,,0 TRAP	13B	ROM ,/1B3,ARCT,ALUA,,,,,PM2,CTP1,WMEM,,BUSR,,U MS	17B	ROM ,/026,,ALUB,,MYQ,,,PYD,,,,,,0 TEST NO 2
0FC	ROM ,/135,,,,,,SM2,,,,,BUSR,,0 TRAP	13C	ROM ,/0C2,AEN,ALUA,CALU,,QYC,,PYD,,,,,,0 MSRD	17C	ROM ,/1C8,AEP,ALUA,,,,SYD,,,,,BUSR,,0 LC
0FD	ROM ,/101,,BINV,,,,,,,WMEM,,BUSR,,CRLOG C1S	13D	ROM ,/1A3,ARCT,ALUA,,,,,PM2,CTP1,WMEM,,BUSR,,0 MSRD	17D	FTA AWR1,AOB,0 LC END
0FE	ROM ,/1E9,AEP,ALUA,,,,SYD,,,,SEQBUS,BUSR,,0 STORE RESULT	13E	ROM FLAG,/00A,AEP,ALUA,,,,,,,WMEM,,,,0 TEST MEMOIRE NO2	17E	ROM ,/080,AEP,APB,,,,,PYD,,WBUS,,,,0 TEST DLA
0FF	ROM FLAG,/0D0,AZ,ALUA,CALU,,QYC,,,,,,,GAEXL,0 TRAP	13F	ROM ,/053,ARA1,TWOA,,MYC,QYC,SM2,PM2,,WMEM,,,,0 TEST MEMOIRE NO1	17F	ROM ,/18E,AWA0,ZERO,,,,,,,,SEQBIO,,GFETCH,0 TEST DLA

Table 2-4B contd.

ADD.		ADD.	
180	ROM ,/028,,,,MYQ,,,,,RBUS,,,GBOK,0 DISPLAY INCR	1C0	ROM ,/1E8,,,,,,,,,RBUS,,,GBOK,0 VERIF MAN. PUPIT
181	ROM ,/020,,,CBIO,MYC,,,,,,SEQBUS,,,0 VERIF MAN REGM	1C1	ROM ,/037,AWA1,AMB,,,,,,,,,,,0 DER MVF REST A1
182	ROM ,/050,ATEN,TWOA,CALU,MYC,,,,,,,,,0 (P856 ROUT AFFICH)	1C2	ROM FLAG,/02C,,BSHR,,MYC,QYC,,,,,,,GCRFNU,0 ROUT.AFFICH.INCR
183	ROM ,/0E9,,,,,,,,,,,,,,0 ROUT.AFFICH.INCR	1C3	ROM ,/085,ATWO,ACR,,MYC,QYC,,,,,,,GFRZO,0 TEST NO 2
184	ROM ,/07A,ARR1,ALUA,,,,,,,WBUS,,BUSR,GBTME,0 WER	1C4	ROM FLAG,/0F1,AWR2,ALUB,,MYQ,,,,,,,,,0 DER MOVE REST R2
185	ROM ,/188,ARR1,ALUA,,,,,,,WBUS,SEQBUS,,,0 WER	1C5	ROM ,/188,AWR1,DBIO,CBIO,,,,,,,SEQBUS,,,CRRTN RER
186	ROM FLAG,/054,AR15,AMB,,,SRQ,,,,,,,,GCRFNU,0 TEST NO 3	1C6	ROM ,/0CC,,,CBIO,,QYC,,,,,SEQBUS,,,0 TEST Q.CHARG NO.
187	ROM FLAG,/064,ARCT,AMB,,,SLQ,,,CTP1,,,,GCRFNU,0 TEST NO 3	1C7	ROM ,/029,,ALUB,CLUR,MYC,,,,,WDUS,,,,0 TEST Q.CHARC NO.
188	ROM ,/076,AWR2,APB,,,,SYD,,,,,BUSR,,0 RTN	1C8	ROM ,/036,AWA2,AMB,,,,,,,,,,,0 DER MVF REST A2
189	ROM SNPLA,/100,,DBIO,,,,SM2,PYD,,,SEQBUS,BUSR,,0 RTN	1C9	ROM ,/03B,ATWO,APB,CALU,MYC,,,,,,,,,0 DER MOVE M=LGTH
18A	ROM ,/09F,,,,,,,,,,,,,,0 TEST NO 2	1CA	ROM ,/09F,,,,,,,,,,,,,,0 TEST DLA
18B	ROM ,/01B,,ZERO,,,SRQ,,PM2,,,,,,0 TEST NO 2	1CB	ROM ,/071,ATWO,ASHR,,MYC,,,,,,,,,0 TEST DLA
18C	ROM ,/072,ATWO,APB,CALU,MYC,,,,,,,,GFENB,0 COMPUTE MASK RTNA15	1CC	ROM ,/07F,,BINV,,,QYC,,,,,,,,,0 TEST NO 2
18D	ROM ,/06E,AWAO,BINV,,MYQ,,,,,,,,,0 COMPUTE MASK RTNA15	1CD	ROM ,/098,ATWO,ASHR,,MYC,QYC,,,,,,,GFRZO,0 TEST NO 3
18E	ROM FLAG,/024,ARA1,AXB,,,,,,,,,,GCRFNU,0 TEST DLA	1CE	ROM ,/0AC,AWAO,ALUB,,,,,,,,,SEQBUS,,GFLOT,0 FPP OP/S LD EXP
18F	ROM ,/061,ATWO,APLB1,,MYC,,,,,,,,,0 TEST RB	1CF	ROM ,/1B8,AEP,ALUA,,,,SYD,,,,SEQBUS,,GFLOT,0 FPP OP LD EXP
190	ROM ,/019,,BINV,,MYC,,,,,,,,,0 ROUT.AFFICH.INCR	1D0	ROM ,/02F,,,,,,,,,,,,,,0 NOT USED
191	ROM ,/1BE,ARAO,AOB,CLUR,MYC,QYC,,,,,,,,0 MASK STACK PSW RTNA15	1D1	ROM ,/06A,AWR1,AMB,,,,SYD,,,,,,,0 CF.15R2N
192	ROM ,/09F,,,,,,,,,,,,,,0 TEST MEMOIRE NO1	1D2	ROM FLAG,/07C,,BSHR,,MYC,SRQ,,,,,,,GCRFNU,0 ROUT.AFFICH.INCR
193	ROM FLAG,/002,,,CBIO,MYC,,,,,,SEQBUS,,,0 TEST MEMOIRE NO1	1D3	ROM ,/032,ARAO,,,,,,,,,,,GCTLD,0 TEST NO 3
194	ROM ,/06A,AWR1,AMB,,,,SYD,,,,,,GFSTOV,0 CF.15R2	1D4	ROM FLAG,/0EA,AWA1,APB,,,,SYD,,,,SEQBUS,,GMOVE,0 MVB WRITE
195	ROM ,/068,APSW,ALUA,,,,SP2,,,WMEM,,BUSR,,0 CF	1D5	ROM FLAG,/0EA,AWA1,APB,,,,SYD,,,,SEQBUS,,GBEX,0 MVUS WRITE
196	ROM ,/0E9,,,,,,,,,,,,,,0 TEST DLA	1D6	ROM ETAT,/09E,,,,,,,,,,SEQBUS,,,0 TEST Q.CHARG NO.
197	ROM ,/067,AEP,ALUA,,MYQ,,SP2,,,WMEM,SEQBUS,BUSR,,0 CF	1D7	ROM FLAG,/0D8,,ALUB,CBIO,,QYC,,,,WBUS,SEQBUS,,,0 ROUT.AFFICH,INCR
198	ROM ,/1E9,,ALUB,,,,SYD,PYD,,,SEQBUS,BUSR,,0 CF END	1D8	ROM FLAG,/084,ATEN,AXB,,,,,,,,,,GCRFNU,0 TEST NO 2
199	ROM ,/05E,,BSHR,CALU,MYC,,,,,,,,,0 EX	1D9	ROM ,/017,AEP,AMB,,,,,PYD,,,,,,0 TEST NO 2
19A	ROM ,/09F,,,,,,,,,,,,,,0 TEST NO 3	1DA	ROM ,/09F,,,,,,,,,,,,,,0 TEST DLA
19B	ROM FLAG,/078,,,,MYQ,,,,,,,,,0 TEST NO 3	1DB	ROM ,/070,ATEN,TWOA,,MYC,,,,,,,,,0 TEST RB
19C	ROM FLAG,/05C,AEP,AXB,,,,SM2,PM2,,,,,GCRFNU,0 TEST MEMOIRE NO2	1DC	ROM FLAG,/098,,,,MYQ,,,,,,,,,0 TEST NO 3
19D	ROM ,/0FB,,,,,,,,,RBUS,,,GBOK,0 TMP TEST	1DD	ROM ,/078,ATWO,ASHR,,MYC,QYC,,,,,,,0 TEST NO 3
19E	ROM ,/060,AWAO,BCR,,MYC,,,,,,,,,0 TEST RB	1DE	ROM FLAG,/00C,,BINV,,,,,,,,,,,0 TEST NO 1
19F	ROM ,/051,AEP,APB,,MYC,,,PYD,,WBUS,,,,0 TEST RB	1DF	ROM ,/049,,ALUB,,,,,,,WBUS,,,,0 VERIF MAN. REG M
1A0	ROM FLAG,/13A,AWAO,ALUB,,MYQ,,,PM2,,,,,,0 MVF TEST LGHT=0	1E0	ROM ,/100,,ALUB,,,,,PYD,,,,,,0 TRAP DLN
1A1	ROM FLAG,/174,ATEN,AXB,,,,,,,,,,GCRFNU,0 EX TEST IF T2	1E1	ROM FLAG,/195,ARA1,ALUA,,,,,,,,,,,,0 DLN
1A2	ROM ,/09F,,,,,,,,,,,,,,0 TEST MEMOIRE NO2	1E2	ROM ,/09F,,,,,,,,,,,,,,0 TEST NO 1
1A3	ROM FLAG,/062,,,CBIO,MYC,,,,,,SEQBUS,,,0 TEST MEMOIRE NO2	1E3	ROM ,/021,,BINV,,MYC,,,,,,,,,0 TEST NO 1
1A4	ROM ,/05A,ARR2,AMB,CALU,MYC,,,,,,,,,0 MVF LGHT-2 IN M	1E4	ROM FLAG,/01A,,,,,,,,PM2,,,,,,0 TEST NO 2
1A5	ROM ,/12B,AWA1,APB,,MYQ,,SYD,,,,,BUSR,,0 MVF A1=SOURCE	1E5	ROM ,/033,,,,MYQ,,,,,,,,,0 TEST NO 2
1A6	ROM ,/096,ATWO,ACR,CALU,MYC,,,,,WBUS,SEQBIO,,GFKYZO,CRIO TMP T	1E6	ROM FLAG,/04C,,BSHR,,MYC,,,,,,,,GCRFNU,0 ROUT.AFFICH.INCR
1A7	ROM FLAG,/058,,,,,SLQ,,,,WBUS,,,CBTMP,0 TEST TMP TPM	1E7	ROM ,/04C,,,,MYQ,,,,,,,,,0 TEST NO 1
1A8	ROM FLAG,/12A,AWA1,AMB,,,,SYD,,,,SEQBUS,,GMOVE,0 MVF WRITE DEST	1E8	ROM ,/016,AEP,APB,CALU,MYC,SRQ,,,,,,,,0 TEST NO 2
1A9	ROM ,/13A,AWR2,ZERO,,,,,,,,,,,0 MVF CLEAR LGHT	1E9	ROM FLAG,/074,ATEN,AXB,,,,,,,,,,GCRFNU,0 TEST NO 2
1AA	ROM ,/09F,,,,,,,,,,,,,,0 TEST NO 3	1EA	ROM ,/09F,,,,,,,,,,,,,,0 TEST RB
1AB	ROM ,/013,AWA2,FORA,,,,,,,,,,,0 TEST NO 3	1EB	ROM ,/033,ATEN,ASHR,CALU,MYC,,,,,,,,,0 TEST RB
1AC	ROM FLAG,/052,,ALUB,,,,SM2,PM2,,WMEM,SEQBUS,,,0 TEST MEMOIRE NO1	1EC	ROM ,/012,ATEN,ALUA,,MYC,QYC,,,,,,,,0 TEST NO 3
1AD	ROM ,/0B0,ARAO,ALUA,,,,SYD,PYD,,,,,,0 TEST MEMOIRE NO1	1ED	ROM FLAG,/044,ARA2,AMB,,,SRQ,,,,,,,GCRFNU,0 TEST NO 3
1AE	ROM SNPLA,/1FF,AWA1,ALUB,CIOR,MYC,QYC,,PM2,,,SEQBIO,,GFETCH,0 TRB	1EE	ROM ,/1E9,AEP,ALUA,,,,SYD,,,,SEQBUS,,,CRFLO FPP STORE ST EXP
1AF	ROM ,/0B1,AWAO,BCR,,,,SYD,PYD,,,,,GFRZO,0 (P856 TEST MEM NO1)	1EF	ROM FLAG,/010,,,,,SLQ,SP2,,,WEXM,SEQBUS,BUSR,GBOF,0 FPP ST MANTI.
1B0	ROM ,/05B,ATWO,ALUA,CALU,MYC,QYC,,,,,,,,0 MVF	1F0	ROM ,/100,,ALUB,,,,,PYD,,,,,,0 TRAP DRN
1B1	ROM ,/046,AWAO,BSHR,,MYQ,,,,,,,,,0 RER	1F1	ROM FLAG,/185,AWA1,ASHR,,,SRQ,,,,,,,GCRDSR,0 DRN
1B2	ROM FLAG,/03C,,BSHR,,MYC,,,,,,,,GCRFNU,0 ROUT.AFFICH.INCR	1F2	ROM ,/09F,,,,,,,,,,,,,,0 TEST NO 1
1B3	ROM FLAG,/0B8,ATWO,AOB,,MYC,QYC,,,,,,,GFRZO,0 TEST NO 1	1F3	ROM ,/07F,,,,,,,,,,,,,,0 TEST NO 1
1B4	ROM FLAG,/0F2,AWAO,ALUB,,,,,PM2,,,,,,0 MVB TEST LENGTH	1F4	ROM ,/0C1,,,,,,SM2,PM2,,,SEQBUS,,,0 TEST MEMOIRE NO2
1B5	ROM FLAG,/10B,AWA2,AMB,,,,,,,,,,,0 MVB PREPARE A2	1F5	ROM ,/063,,ALUB,,,,SYD,PYD,,,,,,0 TEST MEMOIRE NO2
1B6	ROM ETAT,/0BE,,,,,,,,,,SEQBUS,,,0 VERIF MAN. REG M	1F6	ROM ,/097,,DBIO,,,,,,,WBUS,SEQBUS,,,0 VERIF MAN REC L
1B7	ROM FLAG,/02B,,ALUB,,MYQ,,,PM2,,WMEM,,BUSR,,0 MVB PREPARE WRITE	1F7	ROM ,/0BB,ARR1,ALUA,CALU,MYC,,,,,WBUS,,BUSR,GBTMP,0 CIO OTR
1B8	ROM ,/162,,ALUB,,,,,,,WMEM,,BUSR,,0 INT	1F8	ROM ,/036,AEP,ALUA,CALU,MYC,,,,,,,,GCTLD,0 DER MVB READ
1B9	ROM ,/03A,ARAO,AOB,,,,SYD,,,RBUS,,BUSR,GBTME,0 RER	1F9	ROM ,/007,AWA1,AMB,,,,,PP2,,,,,GCTLD,0 DER MVB WRITE
1BA	ROM ,/09F,,,,,,,,,,,,,,0 TEST NO 3	1FA	ROM ,/03E,AEP,ALUA,CALU,MYC,,,,,,,,GCTLD,0 DER MVF READ
1BB	ROM ,/01A,,ZERO,,,SRQ,,,,,,,,0 TEST NO 3	1FB	ROM ,/005,AWA1,APB,,,,,PP2,,,,,CCTLD,0 DER MVF WRITE
1BC	ROM ,/058,ATEN,ALUA,CALU,,QYC,,,,WBUS,SEQBIO,,GFETCH,0 TEST TMP	1FC	ROM FLAG,/06C,ARA1,AXB,,,,SM2,PM2,,,,,GCRFNU,0 TEST MEMOIRE NO1
1BD	ROM ,/0B3,,,,,,SP2,,,,SEQBUS,BUSR,GBTMF,0 FPP DOUBLE LD 2ND WORD	1FD	ROM FLAG,/0C0,ARAO,ALUA,CALU,MYC,,SYD,PYD,,,,,,0 TEST MEMOIRE NO1
1BE	ROM ,/040,,BCR,,MYC,,,,,,,,GFKYZO,0 TEST RB	1FE	ROM ,/12D,ATWO,TWOA,CALU,MYQ,QYC,,,,,,,GCTLD,0 PAGE FAULT ML
1BF	ROM FLAG,/014,AEP,AXB,,MYQ,,,,,,,,GCRFNU,0 TEST RB	1FF	ROM ,/125,ATWO,TWOA,CALU,,QYC,,,,,,,GCTLD,0 PAGE FAULT

Table 2-5A Control-ROM Binary Content P857

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
000	00	111101000	00011	00001	00	10	11	00	00	0	111	00	0	00000	00000
001	11	111111111	00000	00000	11	10	11	10	11	0	000	10	0	11100	00000
002	00	111101001	00000	00000	00	00	00	10	11	0	000	00	1	00000	00000
003	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
004	01	111111000	11110	10010	00	00	11	00	00	0	000	00	0	00000	00000
005	00	111111011	00000	00000	00	00	00	11	10	0	000	00	0	01110	00000
006	00	111111111	10100	00011	00	10	00	00	00	0	000	00	0	00000	00000
007	00	111111111	00000	01110	01	10	00	00	00	0	000	10	0	00000	00000
008	11	111111111	01100	00001	11	10	11	10	11	0	000	10	0	11100	10000
009	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	10000
00A	00	101101101	11110	00011	00	00	00	00	00	0	111	00	0	00000	00000
00B	00	111111111	00000	00011	00	10	00	00	00	0	000	00	0	00000	00000
00C	00	100000001	00100	00011	00	00	00	00	00	0	011	00	1	00000	00000
00D	00	100000001	10100	00011	00	00	00	00	00	0	011	00	1	00000	00000
00E	00	011100110	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
00F	00	011100101	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
010	01	111010000	11110	00011	00	00	11	00	00	0	000	00	0	00000	00000
011	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
012	01	011100000	01000	00001	00	00	00	00	00	0	100	00	0	10011	00000
013	00	011100000	00000	00011	00	00	00	01	01	0	000	00	0	00000	00000
014	00	111011000	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
015	00	111011000	11011	10010	00	10	00	00	10	0	000	00	0	01110	00000
016	11	111111111	00000	00000	11	10	11	10	11	0	000	10	0	11100	00000
017	11	111111110	00000	00000	00	00	00	00	00	0	000	10	0	00001	00000
018	01	110010011	10100	00011	00	10	00	00	00	0	000	00	0	01111	00000
019	01	101011011	10100	00011	00	10	01	00	00	0	000	00	0	01111	00000
01A	01	101111011	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
01B	00	011001110	01001	10001	00	00	01	00	00	1	000	01	0	00110	00000
01C	01	101101011	10100	00011	00	10	00	00	00	0	000	00	0	00000	00000
01D	01	101001011	10100	00011	00	10	01	00	00	0	000	00	0	00000	00000
01E	00	011011101	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
01F	00	011001110	01001	01111	00	00	10	00	00	1	000	01	0	00011	00000
020	11	111111111	01100	00110	11	10	11	10	11	0	000	10	0	11100	10100
021	00	111101001	10100	00110	00	00	00	01	01	0	000	00	1	00000	10100
022	00	011101100	00101	00011	00	00	00	01	00	0	000	00	1	00000	00000
023	00	011101000	00101	00011	00	00	00	01	00	0	000	00	1	00000	00000
024	00	100000001	00100	00110	00	00	00	00	00	0	011	00	1	00000	10100
025	00	100000001	10000	00100	00	00	00	00	00	0	011	00	1	00000	10100
026	00	001110111	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
027	00	110011110	01011	00000	00	00	00	01	00	0	000	00	0	01011	00000
028	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
029	00	011100011	11010	10010	00	10	11	00	01	0	000	00	0	00000	00000
02A	00	101011001	11010	01011	00	10	11	00	00	0	000	00	0	00000	00000
02B	00	101000101	10100	00011	00	00	00	01	00	0	100	00	1	10011	00000
02C	01	111010010	01111	01110	00	11	00	11	10	1	000	10	0	00000	00000
02D	00	011011100	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
02E	00	100110100	00000	00000	10	10	00	00	00	0	000	00	0	00110	00000
02F	01	111000000	11100	00011	00	00	11	00	00	0	000	00	0	00000	00000
030	11	111111111	01100	00000	11	10	11	10	11	0	000	10	0	11100	11000
031	00	111101001	10100	00000	00	00	00	01	01	0	000	00	1	00000	11000
032	10	100000000	00000	00000	01	10	11	10	11	0	000	10	0	00000	00000
033	10	100000000	00000	00000	01	10	11	00	11	0	000	10	0	00000	00000
034	00	100000001	00100	00000	00	00	00	00	00	0	011	00	1	00000	11000
035	01	010111100	11000	00011	00	00	00	00	00	0	000	00	0	00100	00000
036	00	001110111	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
037	11	111111111	01100	00010	11	10	11	10	11	0	000	10	0	11100	00000
038	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
039	00	011011110	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
03A	01	111000100	00000	01101	00	10	00	00	00	0	000	00	0	00100	00000
03B	11	111111111	10100	00011	00	00	00	01	00	0	000	00	0	00000	00000
03C	01	111000010	01111	01110	00	00	00	10	10	1	000	10	0	00101	00000
03D	00	011011100	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
03E	00	101011111	11011	01011	00	00	00	00	01	0	000	00	0	00000	00000
03F	01	110110000	00000	00000	00	00	01	00	00	0	100	00	0	10011	00000
040	11	111111111	01100	01000	11	10	11	10	11	0	000	10	0	11100	10000
041	01	111001000	10110	01000	00	10	00	00	00	0	000	00	1	00000	00000
042	00	011111101	00000	00000	01	10	00	00	11	0	000	10	0	00000	00000
043	00	011110101	00000	00000	01	10	00	00	11	0	000	10	0	00000	00000
044	00	100000001	00100	01000	00	00	00	00	00	0	011	00	1	00000	10000
045	00	100000001	00000	01010	00	00	00	00	00	0	011	00	1	00000	00000
046	11	111111111	00100	01000	11	10	11	10	11	0	000	10	0	11100	10000
047	00	111101001	00000	00000	00	00	00	00	00	0	000	00	1	00000	01100
048	01	110100110	00001	11100	00	00	00	00	00	0	000	00	0	00100	00000
049	01	101000000	00001	00111	00	11	11	00	00	0	000	00	0	00000	00000
04A	00	011100010	10100	00000	00	10	00	00	00	0	000	00	0	00000	00000
04B	01	110110101	01100	10010	00	00	00	00	11	0	000	00	0	00000	00000
04C	01	110110010	00111	00011	00	00	00	10	10	1	011	10	0	00000	00000
04D	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
04E	01	110101000	00000	00000	00	00	01	00	00	0	000	00	0	01101	00000
04F	00	101110010	00000	01110	01	10	00	00	00	0	011	10	0	10001	00000
050	11	111111111	01100	00010	11	10	11	10	11	0	000	10	0	11100	10000
051	01	110001000	10110	00010	00	10	00	00	00	0	000	00	0	00000	00000
052	00	011101100	00000	01110	00	00	00	01	11	0	000	10	1	00000	00000
053	00	011101000	00000	01110	00	00	00	01	11	0	000	10	1	00000	00000
054	00	100000001	00100	00010	00	00	00	00	00	0	011	00	1	00000	10000
055	00	010101000	01000	01101	00	11	00	00	00	0	000	00	0	00000	00000
056	01	110100000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
057	01	110010000	11110	10010	00	00	11	00	00	0	000	00	0	10001	00000
058	00	110100110	01001	11100	00	00	00	00	00	0	000	00	0	00000	00000
059	01	101000000	00001	00111	00	11	11	00	00	0	000	00	0	00000	00000
05A	00	011100010	10100	00000	00	10	00	00	00	0	000	00	0	00000	00000
05B	01	110100101	01100	01111	00	00	10	00	11	0	000	00	0	00011	00000
05C	01	110100010	00111	00011	00	00	00	11	10	1	011	10	0	00000	00000
05D	00	010111110	00000	00001	00	11	00	01	01	0	000	00	0	00000	00000
05E	00	110011000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
05F	00	111111111	10110	00011	00	10	00	00	00	0	000	00	0	00000	00000
060	11	111111111	01100	00111	11	10	11	10	11	0	000	10	0	11100	10000
061	00	001000111	10110	00011	00	10	00	10	00	0	000	00	0	01001	00000
062	00	011111100	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
063	00	011110100	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
064	00	100000001	00100	00111	00	00	00	00	00	0	011	00	1	00000	10000
065	11	111111111	01010	01101	11	10	11	10	11	0	000	10	0	11100	00000
066	11	111111111	00100	00111	11	10	11	10	11	0	000	10	0	11100	10000
067	00	110000000	01000	00001	00	11	00	00	00	0	000	00	0	00000	00000
068	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
069	00	001011111	00101	00011	00	00	11	00	01	0	000	00	0	00000	00000
06A	00	011010110	01000	00001	00	00	00	00	10	0	000	00	0	00000	00000
06B	01	110010101	01001	10001	00	00	01	00	11	0	000	00	0	00000	00000
06C	00	011100100	01100	10010	00	00	00	00	00	1	000	01	0	00000	00000
06D	01	111010111	01000	00001	00	00	00	00	00	0	000	00	0	00000	00000
06E	00	111111111	00000	01110	01	10	00	10	11	0	000	10	0	00000	00000
06F	00	111111111	00000	01110	01	10	00	00	00	0	000	10	0	00000	00000
070	10	100000000	00110	00000	00	00	00	00	00	0	000	00	0	00010	00000
071	00	110001111	00010	00011	00	00	11	00	00	0	000	00	0	01110	00000
072	00	110101101	00000	01110	00	00	00	01	00	0	000	10	1	00000	00000
073	00	110101100	00000	01110	00	00	00	01	00	0	000	10	1	00000	00000
074	11	111111111	00100	00011	11	10	11	10	11	0	000	10	0	11100	10000
075	00	011011100	00010	00011	00	00	00	00	00	0	000	00	1	00100	00000
076	00	010010010	01001	11000	00	00	00	00	00	0	000	00	1	00000	11000
077	00	111000101	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
078	00	011000111	10100	00011	00	10	00	00	00	1	000	00	0	00000	00000
079	00	101010000	11011	01111	00	00	11	00	00	0	000	00	0	11000	00000
07A	00	110010101	01001	10001	00	00	01	00	00	0	000	00	0	00000	00000
07B	01	110000101	01001	01111	00	00	10	00	11	0	000	00	0	00011	00000
07C	00	011000101	10100	00011	00	10	00	00	00	1	000	00	0	00000	00000
07D	00	101001000	11011	01111	00	00	11	00	00	0	001	00	0	10101	00000
07E	00	011000011	10100	00011	00	10	00	00	00	1	000	00	0	00000	00000
07F	01	111110100	00000	01001	00	00	11	00	00	0	000	00	0	00100	00000
080	00	000001000	00000	00001	00	00	00	01	00	0	000	00	0	00000	00000
081	00	100000000	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
082	00	110110010	01010	01110	00	00	00	00	00	0	100	11	0	00000	00000
083	10	100000000	00101	00011	00	10	11	00	00	0	000	00	0	00000	00000
084	00	110001011	01100	10010	00	00	00	00	00	1	000	01	0	00000	00000
085	00	110001011	01100	10001	00	00	01	00	00	1	000	01	0	00110	00000
086	00	101101001	00000	00000	00	00	00	00	00	1	000	00	0	00000	00000
087	00	101111001	00101	00011	00	00	11	00	00	0	000	00	0	00010	00000
088	01	010110011	00000	01010	00	10	00	00	00	0	000	00	1	11001	00000
089	00	010111010	01001	01010	00	00	11	00	00	0	000	00	0	01111	00000
08A	00	100110000	00000	00011	00	00	00	00	00	0	111	00	1	00000	00000
08B	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
08C	01	010101011	11010	01011	00	10	11	00	00	0	001	00	0	10100	00000
08D	00	110101000	00000	00000	00	00	00	00	00	0	000	10	0	01101	00000
08E	01	111010100	10100	00011	00	00	00	10	00	0	000	10	0	00100	00000
08F	00	101011001	00000	01101	00	10	01	00	00	0	000	00	0	00000	00000
090	00	110101010	11010	01011	00	10	00	00	00	0	000	00	0	00000	00000
091	10	100000011	10100	00011	00	00	00	01	00	0	000	00	0	00000	00000
092	00	111101101	10100	00011	00	10	00	00	00	0	100	11	0	11100	00000
093	10	100000000	10100	00011	00	10	11	00	00	0	000	00	0	00000	00000
094	00	110001011	01100	01111	00	00	00	00	00	1	000	01	0	00011	00000
095	01	111000111	01000	00001	00	00	00	00	00	0	000	00	0	00000	00000
096	10	100000000	00111	00011	00	10	00	00	00	0	000	00	0	00000	00000
097	00	101101001	10100	00011	00	00	11	00	00	1	000	00	0	00000	00000
098	01	010110011	00000	01010	00	10	00	00	00	0	000	00	1	11001	00000
099	00	010100101	00010	10010	00	00	11	00	00	0	000	00	0	00000	00000
09A	00	010010101	00001	00011	00	00	00	00	00	0	111	00	1	11001	00000
09B	00	011111110	01101	00001	00	00	00	00	00	0	000	00	0	00000	00000
09C	01	010110011	11011	01111	00	10	00	00	00	0	000	00	1	11001	00000
09D	00	101000011	10100	00011	10	10	00	10	00	0	011	10	1	00110	00000
09E	00	101100000	10100	00011	00	10	00	00	00	0	000	00	0	11010	00000
09F	00	010011011	00000	00000	00	00	00	00	00	0	100	00	0	10100	01100
0A0	00	111010101	00000	01010	00	00	00	01	00	0	000	00	0	01001	00000
0A1	00	111101001	10100	00110	00	00	00	01	01	0	000	00	1	00000	00000
0A2	00	011100111	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
0A3	10	100000000	00101	00011	00	00	11	01	00	0	000	00	0	00000	00000
0A4	00	011011011	01001	10001	00	00	01	00	00	1	000	01	0	00000	00000
0A5	01	000011111	11011	00011	00	00	00	00	01	0	000	00	0	00000	00000
0A6	00	101010101	00000	01101	00	10	00	00	10	0	000	00	0	00000	00000
0A7	00	011001011	10100	00011	00	00	11	00	00	0	000	00	0	00000	00000
0A8	00	010001101	00000	00000	00	00	00	00	00	0	000	00	1	10010	00000
0A9	00	010001110	00000	00000	01	00	11	10	00	0	000	10	1	00000	00000
0AA	00	101001111	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
0AB	00	101010010	00000	00000	01	10	00	10	11	0	000	10	0	00000	00000
0AC	00	100000001	00100	00011	00	00	00	00	00	0	011	00	1	10010	00000
0AD	00	010010100	01010	00110	00	11	11	00	00	0	000	00	0	00000	00000
0AE	00	111101001	10100	00011	00	00	00	01	00	0	000	10	0	00000	00000
0AF	01	101010000	00000	00000	00	00	01	10	00	0	000	10	1	11000	00000
0B0	00	101001101	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
0B1	00	111101001	10100	00000	00	00	00	01	01	0	000	00	1	00000	00000
0B2	01	101110000	10010	00110	00	10	00	00	00	0	011	00	0	00000	00000
0B3	10	100000000	00000	00000	00	00	00	00	11	0	000	00	0	00000	00000
0B4	00	011010010	01001	01111	00	00	10	00	00	1	000	01	0	00011	00000
0B5	01	000001111	11011	00011	00	00	00	00	01	0	000	00	0	00000	00000
0B6	00	111101001	10100	00011	00	00	00	01	00	0	000	10	0	00000	00000
0B7	01	101001000	00000	00000	00	00	01	10	00	0	001	10	1	10101	00000
0B8	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0B9	00	010001010	00000	00000	01	00	11	10	00	0	000	10	1	00000	00000
0BA	00	111101001	01011	01110	00	00	00	00	00	0	000	10	0	00000	00000
0BB	00	101000010	00000	00000	01	10	00	10	11	0	000	10	0	00000	00000
0BC	00	100111101	00000	01101	00	00	00	01	00	0	000	10	0	00000	00000
0BD	00	010001011	01010	00000	00	11	11	00	00	0	000	00	0	00000	00000
0BE	00	010011110	10000	00100	00	10	00	00	00	0	000	00	1	00101	10100
0BF	00	111101001	01010	00001	00	00	00	00	00	0	000	00	1	00101	10000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
0C0	11	111111111	01100	00001	11	10	11	10	11	0	000	10	0	11100	00000
0C1	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
0C2	00	100110101	00000	01001	00	00	00	00	00	0	000	00	1	01100	00000
0C3	00	011101110	00000	00000	01	10	00	00	11	0	000	10	0	00000	00000
0C4	00	001001111	01010	00110	00	00	00	00	00	0	000	00	0	00000	00000
0C5	00	111101001	00000	00011	00	00	00	01	01	0	000	00	1	00000	00000
0C6	00	010001000	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
0C7	11	111111111	00100	00000	11	10	11	10	11	0	000	10	0	11100	11100
0C8	00	010000110	00100	01011	10	10	00	00	00	0	000	00	1	10000	00000
0C9	11	111111111	01100	01001	11	10	11	10	11	0	000	10	0	11100	00000
0CA	00	111101001	00000	01110	00	00	00	01	01	0	000	10	1	00000	00000
0CB	00	100110010	00000	00000	00	00	00	00	00	0	000	00	0	01001	00000
0CC	00	100000001	00100	00011	00	00	00	00	00	0	011	00	1	10000	00000
0CD	00	100111101	00000	01101	00	00	00	01	00	0	000	00	0	01110	00000
0CE	01	100101000	00000	00110	10	10	01	00	00	0	000	00	0	00000	00000
0CF	00	101101110	00000	00000	11	10	11	00	00	0	111	11	0	11100	00000
0D0	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0D1	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0D2	00	100100101	01101	00001	00	00	00	00	00	0	000	00	0	00000	00000
0D3	10	100000000	00000	01110	01	00	11	01	11	0	000	10	0	00000	00000
0D4	01	100011011	01010	00000	01	10	00	01	00	0	000	10	0	00111	00000
0D5	00	001010110	01001	00110	00	00	00	00	00	0	000	00	0	00000	00000
0D6	01	100100000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
0D7	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0D8	11	111111111	00100	00000	11	10	11	10	11	0	000	10	0	11100	11100
0D9	11	111111111	10100	00000	11	10	11	10	11	0	000	10	0	11100	11100
0DA	00	100100100	10110	00011	00	10	00	00	00	0	000	00	0	01001	00000
0DB	00	100100010	00011	00011	00	00	00	01	00	0	001	00	1	10101	00000
0DC	00	010001100	00100	00011	10	10	00	00	00	0	000	00	1	10000	00000
0DD	00	100011101	00000	00001	00	11	00	11	00	0	011	10	1	00000	00000
0DE	01	100011000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
0DF	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0E0	00	001111011	00000	00001	00	00	00	01	00	0	000	00	0	00000	00000
0E1	01	110010111	10100	00011	00	10	00	00	00	0	000	00	0	00000	00000
0E2	00	100010111	11011	00110	00	10	00	11	00	0	001	10	1	10101	00000
0E3	00	011101101	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
0E4	00	001010111	00000	00001	00	11	00	00	10	0	011	00	1	00000	00000
0E5	00	001010111	00000	00001	00	11	00	00	10	0	011	00	1	10010	00000
0E6	01	100001000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
0E7	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0E8	00	100110010	01011	00000	11	10	00	00	00	0	100	11	0	01011	00000
0E9	00	111101001	10100	00011	00	00	00	01	00	0	000	10	1	00000	00100
0EA	00	100010000	11010	10010	00	00	11	00	10	0	000	00	0	00000	00000
0EB	00	001110011	10100	00011	01	11	11	01	00	0	000	10	0	01100	00100
0EC	00	000101110	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
0ED	01	100110000	01000	00001	00	00	00	00	00	0	111	00	0	00000	00000
0EE	00	001101011	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
0EF	00	100001101	01011	00000	00	00	01	01	00	0	000	00	0	01011	00000
0F0	00	001001110	11010	01011	00	10	00	00	00	0	000	00	0	00000	00000
0F1	01	011110111	10100	00011	00	10	00	00	00	0	000	00	0	00000	00000
0F2	00	100000101	10110	00011	00	00	01	10	00	0	011	00	1	01001	00000
0F3	00	100101100	00000	01110	00	00	00	01	00	0	000	10	1	00000	00000
0F4	00	011101011	00001	00011	00	00	00	01	00	0	000	00	1	00000	00000
0F5	00	011101011	00001	00011	00	00	00	01	00	0	000	00	1	10010	00000
0F6	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0F7	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0F8	00	011001000	11000	00011	00	10	00	00	10	0	000	00	0	00000	00000
0F9	11	111111111	01100	00101	11	10	11	10	11	0	000	10	0	11100	10000
0FA	00	100000100	10100	00011	00	11	00	10	00	0	011	10	1	00000	00000
0FB	00	100000011	00000	00001	00	00	00	01	00	0	000	10	0	00000	00000
0FC	00	100110101	00000	00000	00	00	00	11	00	0	000	00	1	00000	00000
0FD	00	100000001	00000	00101	00	00	00	00	00	0	011	00	1	00000	10000
0FE	00	111101001	10100	00011	00	00	00	01	00	0	000	10	1	00000	00000
0FF	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
100	00	000111111	00000	00000	00	00	00	00	00	0	000	00	0	01001	00000
101	00	110011010	00000	00011	00	11	00	00	01	0	000	00	1	00000	00000
102	00	011101100	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
103	00	110101101	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
104	00	011010011	00000	00000	01	10	00	00	00	0	100	11	0	00000	00000
105	01	011010100	00000	00011	00	00	00	00	00	0	000	00	0	00100	00000
106	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
107	00	001100110	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
108	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
109	00	001001011	00101	00011	00	00	00	00	01	0	000	00	0	00000	00000
10A	00	011101000	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
10B	00	110101100	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
10C	00	001001010	11011	00011	00	10	11	00	00	0	000	00	0	00000	00000
10D	00	111101001	00000	00011	00	00	00	01	01	0	000	00	1	00000	00000
10E	00	111101011	00000	00000	00	00	00	00	01	0	000	00	0	00000	00000
10F	00	100100101	11011	10010	00	00	11	00	00	0	000	00	0	00000	00000
110	00	011011111	00000	00000	00	00	00	00	00	0	000	00	0	01101	00000
111	10	100000000	00101	00110	00	00	11	01	00	0	000	00	0	00000	00000
112	00	100101100	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
113	10	100000000	10100	00011	01	10	11	01	00	0	000	10	0	00000	00000
114	00	001001000	01010	00110	01	10	00	01	00	0	000	10	0	00111	00000
115	00	001010110	01010	00110	00	00	00	00	00	0	000	00	0	00000	00000
116	00	010101110	11010	00011	00	10	00	00	00	0	000	00	0	01101	00000
117	10	100000000	00000	00000	01	10	11	00	00	0	000	10	0	00000	00000
118	00	011101100	01101	00110	00	00	00	01	00	0	000	00	1	00000	00000
119	00	111110011	01101	00000	00	00	00	00	00	0	000	00	0	01011	00000
11A	00	111110010	01101	00000	00	00	00	00	00	0	000	00	0	01011	00000
11B	00	111101001	00100	00011	00	00	00	00	00	0	000	00	1	00101	10000
11C	01	110110101	00100	00011	00	00	00	00	00	0	000	00	0	00000	00000
11D	00	100111010	01101	01101	00	00	00	00	00	0	000	00	0	00000	00000
11E	00	111101100	01110	01110	00	00	00	00	00	0	100	11	0	00000	00000
11F	00	111111111	00110	00011	00	10	00	00	00	0	000	00	0	01110	00000
120	00	000001001	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
121	01	110100101	11010	10010	00	10	00	00	01	0	000	00	0	00000	00000
122	00	110001011	01100	01111	00	00	10	00	00	1	000	01	0	00011	00000
123	11	111111111	00001	00011	11	10	11	10	11	0	000	10	0	11100	10000
124	00	011011010	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
125	00	011010111	01010	01101	00	10	00	00	00	0	000	00	0	00100	00000
126	00	000011001	00000	00101	00	10	00	00	00	0	000	00	0	00000	00000
127	11	001111110	00000	00001	00	00	11	00	00	0	000	10	0	00000	00000
128	00	111101001	00001	00011	00	00	00	00	00	0	000	00	1	00101	10000
129	00	101100100	10100	01111	00	10	00	00	00	0	000	00	0	00000	00000
12A	00	001000001	00001	01011	10	10	00	00	00	0	000	00	0	00000	00000
12B	01	000110100	00010	00011	00	00	00	00	00	0	000	00	0	00100	00000
12C	00	001000011	00000	01001	00	00	00	01	00	0	111	00	0	00000	00000
12D	00	011001110	00000	01010	00	00	10	00	00	0	000	00	0	00000	00000
12E	00	100010101	11011	10010	00	10	00	00	10	0	000	00	0	00000	00000
12F	00	100010101	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
130	00	010111111	00000	00000	00	00	00	00	00	0	000	00	0	01101	00000
131	00	011001101	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
132	00	011011100	01010	00001	00	00	00	00	00	0	000	00	1	00100	00000
133	00	000111000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
134	00	011001010	11000	00011	00	10	00	00	01	1	000	00	0	00000	00000
135	00	111010011	01101	00110	00	00	00	01	10	0	000	00	1	00000	00000
136	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
137	00	011000111	10100	00110	00	10	00	00	00	1	000	00	0	00000	00000
138	00	011000110	11000	00011	00	00	00	00	01	0	000	00	0	00000	00000
139	00	111000011	00101	00011	00	00	11	00	10	0	000	00	1	00000	00000
13A	00	011000100	11000	00011	00	00	00	00	01	0	000	00	0	00000	00000
13B	00	110110011	00111	00011	00	00	00	00	10	1	011	00	1	00000	00000
13C	00	011000010	11000	00011	00	00	11	00	01	0	000	00	0	00000	00000
13D	00	110100011	00111	00011	00	00	00	00	10	1	011	00	1	00000	00000
13E	01	000001010	10100	00011	00	00	00	00	00	0	011	00	0	00000	00000
13F	00	001010011	00001	10010	00	10	11	11	10	0	011	00	0	00000	00000
140	00	001111110	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
141	00	111101001	01101	00000	00	00	00	00	00	0	000	00	1	01011	00000
142	00	111001111	01100	01010	00	00	00	00	00	0	000	00	1	00000	00000
143	00	100000001	10000	00000	00	00	00	00	00	0	011	00	1	00000	11000
144	00	110001000	00000	00001	00	00	00	00	00	0	111	11	0	00000	01000
145	00	010110111	00010	00011	00	00	11	00	00	0	000	00	0	00000	00000
146	01	000011100	11011	00110	00	10	00	00	00	0	000	00	0	00100	00000
147	00	000011000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
148	00	010110110	01001	10100	00	00	10	00	00	1	000	01	0	00000	00000
149	00	010110101	01001	10100	00	11	00	00	00	0	000	00	0	11101	00000
14A	00	010110100	01010	01101	00	00	00	00	00	0	000	00	0	00000	00000
14B	00	011010111	00010	00011	00	00	00	00	00	0	000	00	0	00100	00000
14C	00	010101101	00000	00001	00	00	10	10	00	0	000	10	1	11001	00000
14D	00	001000010	00000	00000	00	00	00	10	00	0	000	10	1	11001	00000
14E	00	001010011	11011	01111	00	10	11	11	10	0	011	00	0	00000	00000
14F	00	001101100	01001	00001	00	00	00	11	10	0	000	00	0	00000	00000
150	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	01101	00000
151	00	010100001	01001	01001	00	10	11	00	00	0	000	00	0	00000	00000
152	01	000110000	00000	00000	00	11	00	10	00	0	000	10	1	11001	00000
153	00	101110011	00000	10010	00	00	00	01	00	0	000	00	0	00000	00000
154	00	000010000	00000	00101	00	00	11	10	00	0	001	10	1	10100	00000
155	00	000010000	00000	00000	00	00	00	10	00	0	001	10	1	10100	00000
156	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
157	00	010100111	00000	00010	00	00	00	01	00	0	100	00	1	10111	00000
158	00	010100110	01100	01110	11	10	00	00	00	0	000	10	0	00110	01000
159	00	110001000	01100	00010	00	00	01	00	00	0	000	00	0	00000	00000
15A	00	010100100	01001	00011	00	00	00	00	00	0	000	00	0	01111	00000
15B	00	010100011	01001	11110	00	00	01	00	00	1	000	00	0	00011	00000
15C	00	010100010	01001	11110	00	00	01	00	00	1	000	01	0	00000	00000
15D	01	110110110	01001	11100	00	00	01	00	00	0	000	00	0	00100	00000
15E	00	010100000	11011	00110	10	10	00	00	01	0	000	00	0	00000	00000
15F	00	010010001	01010	01001	00	00	10	00	00	0	000	00	0	00000	00000
160	00	000111001	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
161	11	111111111	01010	00001	11	10	11	10	11	0	000	10	0	11100	00000
162	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
163	00	000110011	10000	00011	00	10	00	00	00	0	000	00	0	00000	00000
164	00	010011010	01001	01110	00	00	00	00	00	0	100	11	0	10100	00000
165	00	101111101	00000	00000	00	00	00	00	00	0	100	00	1	10100	00000
166	00	000100010	01111	00001	00	00	00	00	00	1	000	00	0	00000	00000
167	00	000100011	01111	00001	00	00	01	00	00	1	000	00	0	00000	00000
168	11	011011110	00000	00000	00	00	00	00	00	0	000	10	0	00000	00000
169	01	010011100	10110	01000	00	00	00	00	00	0	000	00	0	00100	00000
16A	00	000110000	00010	00011	00	00	00	00	00	0	111	10	1	11001	00000
16B	00	010010011	01010	10010	00	00	00	00	00	0	000	00	0	00100	00000
16C	00	010010010	01001	11010	00	00	00	00	00	0	000	00	1	00000	10100
16D	11	111111111	01010	01111	11	10	11	10	11	0	000	10	0	11100	00000
16E	00	010010000	10100	10010	00	10	00	00	00	0	000	00	0	00000	00000
16F	00	010000001	10100	00110	00	11	00	00	01	0	000	00	0	00000	00000
170	00	000011001	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
171	00	101010010	10100	00011	01	10	00	01	00	0	000	10	0	00000	00000
172	00	111110111	10100	00011	01	10	00	01	00	0	000	10	1	00000	00000
173	00	010001000	10100	00011	11	00	11	01	00	0	000	10	0	00000	00000
174	00	110001001	01010	10010	00	00	00	00	00	0	000	00	0	00100	00000
175	00	101000010	10100	00011	01	10	00	01	00	0	000	10	0	00000	00000
176	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
177	00	010000111	01000	00001	00	11	00	00	00	0	000	00	1	00000	00000
178	11	111111111	00000	00000	11	10	11	10	11	0	000	10	0	11100	11100
179	00	010000011	01100	01001	11	10	00	00	00	0	000	10	0	00000	00000
17A	00	000100111	00000	01010	00	11	10	00	00	0	000	00	0	00000	00000
17B	00	000100110	00000	00001	00	11	00	00	01	0	000	00	0	00000	00000
17C	00	111001000	10100	00011	00	00	00	01	00	0	000	00	1	00000	00000
17D	11	111111111	01100	00010	11	10	11	10	11	0	000	10	0	11100	00000
17E	00	010000000	10100	00110	00	00	00	00	01	0	111	00	0	00000	00000
17F	00	110001110	01000	01010	00	00	00	00	00	0	000	11	0	11100	00000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
180	00	000101000	00000	00000	00	11	00	00	00	0	100	00	0	10011	00000
181	00	000100000	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
182	00	001010000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
183	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
184	00	001111010	00100	00011	00	00	00	00	00	0	111	00	1	10110	00000
185	00	110001000	00100	00011	00	00	00	00	00	0	111	10	0	00000	00000
186	01	001010100	00011	00000	00	00	10	00	00	0	000	00	0	00100	00000
187	01	001100100	00111	00000	00	00	01	00	00	1	000	00	0	00100	00000
188	00	001110110	01101	00110	00	00	00	01	00	0	000	00	1	00000	00000
189	10	100000000	00000	01110	00	00	00	11	01	0	000	10	1	00000	00000
18A	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
18B	00	000011011	00000	01010	00	00	10	00	10	0	000	00	0	00000	00000
18C	00	001110010	11011	00110	00	10	00	00	00	0	000	00	0	01010	00000
18D	00	001101110	01000	00101	00	11	00	00	00	0	000	00	0	00000	00000
18E	01	000100100	00001	00111	00	00	00	00	00	0	000	00	0	00100	00000
18F	00	001100001	11011	00100	00	10	00	00	00	0	000	00	0	00000	00000
190	00	000011001	00000	00101	00	10	00	00	00	0	000	00	0	00000	00000
191	00	110111110	00000	00010	10	10	11	00	00	0	000	00	0	00000	00000
192	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
193	01	000000010	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
194	00	001101010	01100	00000	00	00	00	01	00	0	000	00	0	01011	00000
195	00	001101000	10110	00011	00	00	00	10	00	0	011	00	1	00000	00000
196	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
197	00	001100111	10100	00011	00	11	00	10	00	0	011	10	1	00000	00000
198	00	111101001	00000	00001	00	00	00	01	01	0	000	10	1	00000	00000
199	00	001011110	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
19A	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
19B	01	001111000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
19C	01	001011100	10100	00111	00	00	00	11	10	0	000	00	0	00100	00000
19D	00	011111011	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
19E	00	001100000	01000	01001	00	10	00	00	00	0	000	00	0	00000	00000
19F	00	001010001	10100	00110	00	10	00	00	01	0	111	00	0	00000	00000
1C0	00	111101000	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
1C1	00	000110111	01001	00000	00	00	00	00	00	0	000	00	0	00000	00000
1C2	01	000101100	00000	01101	00	10	11	00	00	0	000	00	0	00100	00000
1C3	00	010000101	11011	01011	00	10	11	00	00	0	000	00	0	01101	00000
1C4	01	011110001	01101	00001	00	11	00	00	00	0	000	00	0	00000	00000
1C5	00	110001000	01100	01110	01	00	00	00	00	0	000	10	0	00000	00100
1C6	00	011001100	00000	00000	01	00	11	00	00	0	000	10	0	00000	00000
1C7	00	000101001	00000	00001	10	10	00	00	00	0	111	00	0	00000	00000
1C8	00	000110110	01010	00000	00	00	00	00	00	0	000	00	0	00000	00000
1C9	00	000111011	11011	00110	00	10	00	00	00	0	000	00	0	00000	00000
1CA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1CB	00	001110001	11011	01111	00	10	00	00	00	0	000	00	0	00000	00000
1CC	00	001111111	00000	00101	00	00	11	00	00	0	000	00	0	00000	00000
1CD	00	010011000	11011	01111	00	10	11	00	00	0	000	00	0	01101	00000
1CE	00	010101100	01000	00001	00	00	00	00	00	0	000	10	0	11010	00000
1CF	00	110111000	10100	00011	00	00	00	01	00	0	000	10	0	11010	00000
1D0	00	000101111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1D1	00	001101010	01100	00000	00	00	00	01	00	0	000	00	0	00000	00000
1D2	01	001111100	00000	01101	00	10	10	00	00	0	000	00	0	00100	00000
1D3	00	000110010	00000	00000	00	00	00	00	00	0	000	00	0	00010	00000
1D4	01	011101010	01001	00110	00	00	00	01	00	0	000	10	0	00111	00000
1D5	01	011101010	01001	00110	00	00	00	01	00	0	000	10	0	10010	00000
1D6	11	010011110	00000	00000	00	00	00	00	00	0	000	10	0	00000	00000
1D7	01	011011000	00000	00001	01	00	11	00	00	0	111	10	0	00000	00000
1D8	01	010000100	11010	00111	00	00	00	00	00	0	000	00	0	00100	00000
1D9	00	000010111	10100	00000	00	00	00	00	01	0	000	00	0	00000	00000
1DA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1DB	00	001110000	11010	10010	00	10	00	00	00	0	000	00	0	00000	00000
1DC	01	010011000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
1DD	00	001111000	11011	01111	00	10	11	00	00	0	000	00	0	00000	00000
1DE	01	000001100	00000	00101	00	00	00	00	00	0	000	00	0	00000	00000
1DF	00	001001001	00000	00001	00	00	00	00	00	0	111	00	0	00000	00000
1A0	01	100111010	01000	00001	00	11	00	00	10	0	000	00	0	00000	00000
1A1	01	101110100	11010	00111	00	00	00	00	00	0	000	00	0	00100	00000
1A2	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1A3	01	001100010	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
1A4	00	001011010	00101	00000	00	10	00	00	00	0	000	00	0	00000	00000
1A5	00	100101011	01001	00110	00	11	00	01	00	0	000	00	1	00000	00000
1A6	00	010010110	11011	01011	00	10	00	00	00	0	111	11	0	01110	01000
1A7	01	001011000	00000	00000	00	00	01	00	00	0	111	00	0	10111	00000
1A8	01	100101010	01001	00000	00	00	00	01	00	0	000	10	0	00111	00000
1A9	00	100111010	01101	01010	00	00	00	00	00	0	000	00	0	00000	00000
1AA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1AB	00	000010011	01010	10000	00	00	00	00	00	0	000	00	0	00000	00000
1AC	01	001010010	00000	00001	00	00	00	11	10	0	011	10	0	00000	00000
1AD	00	010110000	00000	00011	00	00	00	01	01	0	000	00	0	00000	00000
1AE	10	111111111	01001	00001	11	10	11	00	10	0	000	11	0	11100	00000
1AF	00	010110001	01000	00001	00	00	00	01	01	0	000	00	0	01101	00000
1B0	00	001011011	11011	00011	00	10	11	00	00	0	000	00	0	00000	00000
1B1	00	001000110	01000	01101	00	11	00	00	00	0	000	00	0	00000	00000
1B2	01	000111100	00000	01101	00	10	00	00	00	0	000	00	0	00100	00000
1B3	01	010111000	11011	00010	00	10	11	00	00	0	000	00	0	01101	00000
1B4	01	011110010	01000	00001	00	00	00	00	10	0	000	00	0	00000	00000
1B5	01	100001011	01010	00000	00	00	00	00	00	0	000	00	0	00000	00000
1B6	11	010111110	00000	00000	00	00	00	00	00	0	000	10	0	00000	00000
1B7	01	000101011	00000	00001	00	11	00	00	10	0	011	00	1	00000	00000
1B8	00	101100010	00000	00001	00	00	00	00	00	0	011	00	1	00000	00000
1B9	00	000111010	00000	00010	00	00	00	01	00	0	100	00	1	10110	00000
1BA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1BB	00	000011010	00000	01010	00	00	10	00	00	0	000	00	0	00000	00000
1BC	00	001011000	11010	00011	00	00	11	00	00	0	111	11	0	11100	00000
1BD	00	010110011	00000	00000	00	00	00	10	00	0	000	10	1	11001	00000
1BE	00	001000000	00000	01001	00	10	00	00	00	0	000	00	0	01110	00000
1BF	01	000010100	10100	00111	00	11	00	00	00	0	000	00	0	00100	00000
1E0	00	100000000	00000	00001	00	00	00	00	01	0	000	00	0	00000	00000
1E1	01	110010101	00001	00011	00	00	00	00	00	0	000	00	0	00000	00000
1E2	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1E3	00	000100001	00000	00101	00	10	00	00	00	0	000	00	0	00000	00000
1E4	01	000011010	00000	00000	00	00	00	00	10	0	000	00	0	00000	00000
1E5	00	000110011	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
1E6	01	001001100	00000	01101	00	10	00	00	00	0	000	00	0	00100	00000
1E7	00	001001100	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
1E8	00	000010110	10100	00110	00	10	10	00	00	0	000	00	0	00000	00000
1E9	01	001110100	11010	00111	00	00	00	00	00	0	000	00	0	00100	00000
1EA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1EB	00	000110011	11010	01111	00	10	00	00	00	0	000	00	0	00000	00000
1EC	00	000010010	11010	00011	00	10	11	00	00	0	000	00	0	00000	00000
1ED	01	001000100	00010	00000	00	00	10	00	00	0	000	00	0	00100	00000
1EE	00	111101001	10100	00011	00	00	00	01	00	0	000	10	0	00000	01100
1EF	01	000010000	00000	00000	00	00	01	10	00	0	001	10	1	10100	00000
1F0	00	100000000	00000	00001	00	00	00	00	01	0	000	00	0	00000	00000
1F1	01	110000101	01001	01111	00	00	10	00	00	0	000	00	0	00011	00000
1F2	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1F3	00	001111111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1F4	00	011000001	00000	00000	00	00	00	11	10	0	000	10	0	00000	00000
1F5	00	001100011	00000	00001	00	00	00	01	01	0	000	00	0	00000	00000
1F6	00	010010111	00000	01110	00	00	00	00	00	0	111	10	0	00000	00000
1F7	00	010111011	00100	00011	00	10	00	00	00	0	111	00	1	10111	00000
1F8	00	000110110	10100	00011	00	10	00	00	00	0	000	00	0	00010	00000
1F9	00	000000111	01001	00000	00	00	00	00	11	0	000	00	0	00010	00000
1FA	00	000111110	10100	00011	00	10	00	00	00	0	000	00	0	00010	00000
1FB	00	000000101	01001	00110	00	00	00	00	11	0	000	00	0	00010	00000
1FC	01	001101100	00001	00111	00	00	00	11	10	0	000	00	0	00100	00000
1FD	01	011000000	00000	00011	00	10	00	01	01	0	000	00	0	00000	00000
1FE	00	100101101	11011	10010	00	11	11	00	00	0	000	00	0	00010	00000
1FF	00	100100101	11011	10010	00	00	11	00	00	0	000	00	0	00010	00000

Table 2-5B Control-ROM Binary Content P856

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
000	00	111101000	00011	00001	00	10	11	00	00	0	111	00	0	00000	00000
001	11	111111111	00000	00000	11	10	11	10	11	0	000	10	0	11100	00000
002	00	111101001	00000	00000	00	00	00	10	11	0	000	00	1	00000	00000
003	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
004	01	111111000	11110	10010	00	00	11	00	00	0	000	00	0	00000	00000
005	00	111111011	00000	00000	00	00	00	11	10	0	000	00	0	01110	00000
006	00	111111111	10100	00011	00	10	00	00	00	0	000	00	0	00000	00000
007	00	111111111	00000	01110	01	10	00	00	00	0	000	10	0	00000	00000
008	11	111111111	01100	00001	11	10	11	10	11	0	000	10	0	11100	10000
009	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	10000
00A	00	101101101	11110	00011	00	00	00	00	00	0	111	00	0	00000	00000
00B	00	111111111	00000	00011	00	10	00	00	00	0	000	00	0	00000	00000
00C	00	100000001	00100	00011	00	00	00	00	00	0	011	00	1	00000	00000
00D	00	100000001	10100	00011	00	00	00	00	00	0	011	00	1	00000	00000
00E	00	011100110	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
00F	00	011100101	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
010	01	111010000	11110	00011	00	00	11	00	00	0	000	00	0	00000	00000
011	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
012	01	011100000	01000	00001	00	00	00	00	00	0	100	00	0	10011	00000
013	00	011100000	00000	00011	00	00	00	01	01	0	000	00	0	00000	00000
014	00	111011000	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
015	00	111011000	11011	10010	00	10	00	00	10	0	000	00	0	01110	00000
016	11	111111111	00000	00000	11	10	11	10	11	0	000	10	0	11100	00000
017	11	111111110	00000	00000	00	00	00	00	00	0	000	10	0	00001	00000
018	01	110010011	10100	00011	00	10	00	00	00	0	000	00	0	01111	00000
019	01	101011011	10100	00011	00	10	01	00	00	0	000	00	0	01111	00000
01A	01	101111011	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
01B	00	011001110	01001	10001	00	00	01	00	00	1	000	01	0	00110	00000
01C	01	101101011	10100	00011	00	10	00	00	00	0	000	00	0	00000	00000
01D	01	101001011	10100	00011	00	10	01	00	00	0	000	00	0	00000	00000
01E	00	011011101	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
01F	00	011001110	01001	01111	00	00	10	00	00	1	000	01	0	00011	00000
020	11	111111111	01100	00110	11	10	11	10	11	0	000	10	0	11100	10100
021	00	111101001	10100	00110	00	00	00	01	01	0	000	00	1	00000	10100
022	00	011101100	00101	00011	00	00	00	01	00	0	000	00	1	00000	00000
023	00	011101000	00101	00011	00	00	00	01	00	0	000	00	1	00000	00000
024	00	100000001	00100	00110	00	00	00	00	00	0	011	00	1	00000	10100
025	00	100000001	10000	00100	00	00	00	00	00	0	011	00	1	00000	10100
026	00	001110111	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
027	00	110011110	01011	00000	00	00	00	01	00	0	000	00	0	01011	00000
028	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
029	00	011100011	11010	10010	00	10	11	00	01	0	000	00	0	00000	00000
02A	00	101011001	11010	01011	00	10	11	00	00	0	000	00	0	00000	00000
02B	00	101000101	10100	00011	00	00	00	01	00	0	100	00	1	10011	00000
02C	01	111010010	01111	01110	00	11	00	11	10	1	000	10	0	00000	00000
02D	00	011011100	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
02E	00	100110100	00000	00000	10	10	00	00	00	0	000	00	0	00110	00000
02F	01	111000000	11100	00011	00	00	11	00	00	0	000	00	0	00000	00000
030	11	111111111	01100	00000	11	10	11	10	11	0	000	10	0	11100	11000
031	00	111101001	10100	00000	00	00	00	01	01	0	000	00	1	00000	11000
032	10	100000000	00000	00000	01	10	11	10	11	0	000	10	0	00000	00000
033	10	100000000	00000	00000	01	10	11	00	11	0	000	10	0	00000	00000
034	00	100000001	00100	00000	00	00	00	00	00	0	011	00	1	00000	11000
035	01	010111100	11000	00011	00	00	00	00	00	0	000	00	0	00100	00000
036	00	001110111	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
037	11	111111111	01100	00010	11	10	11	10	11	0	000	10	0	11100	00000
038	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
039	00	011011110	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
03A	01	111000100	00000	01101	00	10	00	00	00	0	000	00	0	00100	00000
03B	11	111111111	10100	00011	00	00	00	01	00	0	000	00	0	00000	00000
03C	01	111000010	01111	01110	00	00	00	10	10	1	000	10	0	00101	00000
03D	00	011011100	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
03E	00	101011111	11011	01011	00	00	00	00	01	0	000	00	0	00000	00000
03F	01	110110000	00000	00000	00	00	01	00	00	0	100	00	0	10011	00000
040	11	111111111	01100	01000	11	10	11	10	11	0	000	10	0	11100	10000
041	01	111001000	10110	01000	00	10	00	00	00	0	000	00	1	00000	00000
042	00	011111101	00000	00000	01	10	00	00	11	0	000	10	0	00000	00000
043	00	011110101	00000	00000	01	10	00	00	11	0	000	10	0	00000	00000
044	00	100000001	00100	01000	00	00	00	00	00	0	011	00	1	00000	10000
045	00	100000001	00000	01010	00	00	00	00	00	0	011	00	1	00000	00000
046	11	111111111	00100	01000	11	10	11	10	11	0	000	10	0	11100	10000
047	00	111101001	00000	00000	00	00	00	00	00	0	000	00	1	00000	01100
048	01	110100110	00001	11100	00	00	00	00	00	0	000	00	0	00100	00000
049	01	101000000	00001	00111	00	11	11	00	00	0	000	00	0	00000	00000
04A	00	011100010	10100	00000	00	10	00	00	00	0	000	00	0	00000	00000
04B	01	110110101	01100	10010	00	00	00	00	11	0	000	00	0	00000	00000
04C	01	110110010	00111	00011	00	00	00	10	10	1	011	10	0	00000	00000
04D	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
04E	01	110101000	00000	00000	00	00	01	00	00	0	000	00	0	01101	00000
04F	00	101110010	00000	01110	01	10	00	00	00	0	011	10	0	10001	00000
050	11	111111111	01100	00010	11	10	11	10	11	0	000	10	0	11100	10000
051	01	110001000	10110	00010	00	10	00	00	00	0	000	00	0	00000	00000
052	00	011101100	00000	01110	00	00	00	01	11	0	000	10	1	00000	00000
053	00	011101000	00000	01110	00	00	00	01	11	0	000	10	1	00000	00000
054	00	100000001	00100	00010	00	00	00	00	00	0	011	00	1	00000	10000
055	00	010101000	01000	01101	00	11	00	00	00	0	000	00	0	00000	00000
056	01	110100000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
057	01	110010000	11110	10010	00	00	11	00	00	0	000	00	0	10001	00000
058	00	110100110	01001	11100	00	00	00	00	00	0	000	00	0	00000	00000
059	01	101000000	00001	00111	00	11	11	00	00	0	000	00	0	00000	00000
05A	00	011100010	10100	00000	00	10	00	00	00	0	000	00	0	00000	00000
05B	01	110100101	01100	01111	00	00	10	00	11	0	000	00	0	00011	00000
05C	01	110100010	00111	00011	00	00	00	11	10	1	011	10	0	00000	00000
05D	00	010111110	00000	00001	00	11	00	01	01	0	000	00	0	00000	00000
05E	00	110011000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
05F	00	111111111	10110	00011	00	10	00	00	00	0	000	00	0	00000	00000
060	11	111111111	01100	00111	11	10	11	10	11	0	000	10	0	11100	10000
061	00	001000111	10110	00011	00	10	00	10	00	0	000	00	0	01001	00000
062	00	011111100	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
063	00	011110100	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
064	00	100000001	00100	00111	00	00	00	00	00	0	011	00	1	00000	10000
065	11	111111111	01010	01101	11	10	11	10	11	0	000	10	0	11100	00000
066	11	111111111	00100	00111	11	10	11	10	11	0	000	10	0	11100	10000
067	00	110000000	01000	00001	00	11	00	00	00	0	000	00	0	00000	00000
068	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
069	00	001011111	00101	00011	00	00	11	00	01	0	000	00	0	00000	00000
06A	00	011010110	01000	00001	00	00	00	00	10	0	000	00	0	00000	00000
06B	01	110010101	01001	10001	00	00	01	00	11	0	000	00	0	00000	00000
06C	00	011100100	01100	10010	00	00	00	00	00	1	000	01	0	00000	00000
06D	01	111010111	01000	00001	00	00	00	00	00	0	000	00	0	00000	00000
06E	00	111111111	00000	01110	01	10	00	10	11	0	000	10	0	00000	00000
06F	00	111111111	00000	01110	01	10	00	00	00	0	000	10	0	00000	00000
070	10	100000000	00110	00000	00	00	00	00	00	0	000	00	0	00010	00000
071	00	110001111	00010	00011	00	00	11	00	00	0	000	00	0	01110	00000
072	00	110101101	00000	01110	00	00	00	01	00	0	000	10	1	00000	00000
073	00	110101100	00000	01110	00	00	00	01	00	0	000	10	1	00000	00000
074	11	111111111	00100	00011	11	10	11	10	11	0	000	10	0	11100	10000
075	00	011011100	00010	00011	00	00	00	00	00	0	000	00	1	00100	00000
076	00	010010010	01001	11000	00	00	00	00	00	0	000	00	1	00000	11000
077	00	111000101	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
078	00	011000111	10100	00011	00	10	00	00	00	1	000	00	0	00000	00000
079	00	101010000	11011	01111	00	00	11	00	00	0	000	00	0	11111	00000
07A	00	110010101	01001	10001	00	00	01	00	00	0	000	00	0	00000	00000
07B	01	110000101	01001	01111	00	00	10	00	11	0	000	00	0	00011	00000
07C	00	011000101	10100	00011	00	10	00	00	00	1	000	00	0	00000	00000
07D	00	101001000	11011	01111	00	00	11	00	00	0	001	00	0	10101	00000
07E	00	011000011	10100	00011	00	10	00	00	00	1	000	00	0	00000	00000
07F	01	111110100	00000	01001	00	00	11	00	00	0	000	00	0	00100	00000
080	00	000001000	00000	00001	00	00	00	01	00	0	000	00	0	00000	00000
081	00	100000000	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
082	00	110110010	01010	01110	00	00	00	00	00	0	100	11	0	00000	00000
083	10	100000000	00101	00011	00	10	11	00	00	0	000	00	0	00000	00000
084	00	110001011	01100	10010	00	00	00	00	00	1	000	01	0	00000	00000
085	00	110001011	01100	10001	00	00	01	00	00	1	000	01	0	00110	00000
086	00	101101001	00000	00000	00	00	00	00	00	1	000	00	0	00000	00000
087	00	101111001	00101	00011	00	00	11	00	00	0	000	00	0	00010	00000
088	01	010110011	00000	01010	00	10	00	00	00	0	000	00	1	11111	00000
089	00	010111010	01001	01010	00	00	11	00	00	0	000	00	0	01111	00000
08A	00	100110000	00000	00011	00	00	00	00	00	0	111	00	1	00000	00000
08B	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
08C	01	010101011	11010	01011	00	10	11	00	00	0	001	00	0	10100	00000
08D	00	110101000	00000	00000	00	00	00	00	00	0	000	10	0	01101	00000
08E	01	111010100	10100	00011	00	00	00	10	00	0	000	10	0	00100	00000
08F	00	101011001	00000	01101	00	10	01	00	00	0	000	00	0	00000	00000
090	00	110101010	11010	01011	00	10	00	00	00	0	000	00	0	00000	00000
091	10	100000011	10100	00011	00	00	00	01	00	0	000	00	0	00000	00000
092	00	111101101	10100	00011	00	10	00	00	00	0	100	11	0	11100	00000
093	10	100000000	10100	00011	00	10	11	00	00	0	000	00	0	00000	00000
094	00	110001011	01100	01111	00	00	00	00	00	1	000	01	0	00011	00000
095	01	111000111	01000	00001	00	00	00	00	00	0	000	00	0	00000	00000
096	10	100000000	00111	00011	00	10	00	00	00	0	000	00	0	00000	00000
097	00	101101001	10100	00011	00	00	11	00	00	1	000	00	0	00000	00000
098	01	010110011	00000	01010	00	10	00	00	00	0	000	00	1	11111	00000
099	00	010100101	00010	10010	00	00	11	00	00	0	000	00	0	00000	00000
09A	00	010010101	00001	00011	00	00	00	00	00	0	111	00	1	11111	00000
09B	00	011111110	01101	00001	00	00	00	00	00	0	000	00	0	00000	00000
09C	01	010110011	11011	01111	00	10	00	00	00	0	000	00	1	11111	00000
09D	00	101000011	10100	00011	10	10	00	10	00	0	011	10	1	00110	00000
09E	00	101100000	10100	00011	00	10	00	00	00	0	000	00	0	11111	00000
09F	00	010011011	00000	00000	00	00	00	00	00	0	100	00	0	10100	01100
0A0	00	111010101	00000	01010	00	00	00	01	00	0	000	00	0	01001	00000
0A1	00	111101001	10100	00110	00	00	00	01	01	0	000	00	1	00000	00000
0A2	00	011100111	11011	00011	00	10	00	00	00	0	000	00	0	00000	00000
0A3	10	100000000	00101	00011	00	00	11	01	00	0	000	00	0	00000	00000
0A4	00	011011011	01001	10001	00	00	01	00	00	1	000	01	0	00000	00000
0A5	01	000011111	11011	00011	00	00	00	00	01	0	000	00	0	00000	00000
0A6	00	101010101	00000	01101	00	10	00	00	10	0	000	00	0	00000	00000
0A7	00	011001011	10100	00011	00	00	11	00	00	0	000	00	0	00000	00000
0A8	00	010001101	00000	00000	00	00	00	00	00	0	000	00	1	10010	00000
0A9	00	010001110	00000	00000	01	00	11	10	00	0	000	10	1	00000	00000
0AA	00	101001111	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
0AB	00	101010010	00000	00000	01	10	00	10	11	0	000	10	0	00000	00000
0AC	00	100000001	00100	00011	00	00	00	00	00	0	011	00	1	10010	00000
0AD	00	010010100	01010	00110	00	11	11	00	00	0	000	00	0	00000	00000
0AE	00	111101001	10100	00011	00	00	00	01	00	0	000	10	0	00000	00000
0AF	01	101010000	00000	00000	00	00	01	10	00	0	000	10	1	11111	00000
0B0	00	101001101	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
0B1	00	111101001	10100	00000	00	00	00	01	01	0	000	00	1	00000	00000
0B2	01	101110000	10010	00110	00	10	00	00	00	0	011	00	0	00000	00000
0B3	10	100000000	00000	00000	00	00	00	00	11	0	000	00	0	00000	00000
0B4	00	011010010	01001	01111	00	00	10	00	00	1	000	01	0	00011	00000
0B5	01	000001111	11011	00011	00	00	00	00	01	0	000	00	0	00000	00000
0B6	00	111101001	10100	00011	00	00	00	01	00	0	000	10	0	00000	00000
0B7	01	101001000	00000	00000	00	00	01	10	00	0	001	10	1	10101	00000
0B8	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0B9	00	010001010	00000	00000	01	00	11	10	00	0	000	10	1	00000	00000
0BA	00	111101001	01011	01110	00	00	00	00	00	0	000	10	0	00000	00000
0BB	00	101000010	00000	00000	01	10	00	10	11	0	000	10	0	00000	00000
0BC	00	100111101	00000	01101	00	00	00	01	00	0	000	10	0	00000	00000
0BD	00	010001011	01010	00000	00	11	11	00	00	0	000	00	0	00000	00000
0BE	00	010011110	10000	00100	00	10	00	00	00	0	000	00	1	00101	10100
0BF	00	111101001	01010	00001	00	00	00	00	00	0	000	00	1	00101	10000

Table 2-5B contd.

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
0C0	11	111111111	01100	00001	11	10	11	10	11	0	000	10	0	11100	00000
0C1	00	111101001	00000	00001	00	00	00	01	01	0	000	00	1	00000	00000
0C2	00	100110101	00000	01001	00	00	00	00	00	0	000	00	1	01100	00000
0C3	00	011101110	00000	00000	01	10	00	00	11	0	000	10	0	00000	00000
0C4	00	001001111	01010	00110	00	00	00	00	00	0	000	00	0	00000	00000
0C5	00	111101001	00000	00011	00	00	00	01	01	0	000	00	1	00000	00000
0C6	00	010001000	00100	00011	00	00	11	00	00	0	000	00	0	00000	00000
0C7	11	111111111	00100	00000	11	10	11	10	11	0	000	10	0	11100	11100
0C8	00	010000110	00100	01011	10	10	00	00	00	0	000	00	1	10000	00000
0C9	11	111111111	01100	01001	11	10	11	10	11	0	000	10	0	11100	00000
0CA	00	111101001	00000	01110	00	00	00	01	01	0	000	10	1	00000	00000
0CB	00	100110010	00000	00000	00	00	00	00	00	0	000	00	0	01001	00000
0CC	00	100000001	00100	00011	00	00	00	00	00	0	011	00	1	10000	00000
0CD	00	100111101	00000	01101	00	00	00	01	00	0	000	00	0	01110	00000
0CE	01	100101000	00000	00110	10	10	01	00	00	0	000	00	0	00000	00000
0CF	00	101101110	00000	00000	11	10	11	00	00	0	111	11	0	11100	00000
0D0	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0D1	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0D2	00	100100101	01101	00001	00	00	00	00	00	0	000	00	0	00000	00000
0D3	10	100000000	00000	01110	01	00	11	01	11	0	000	10	0	00000	00000
0D4	01	100011011	01010	00000	01	10	00	01	00	0	000	10	0	00111	00000
0D5	00	001010110	01001	00110	00	00	00	00	00	0	000	00	0	00000	00000
0D6	01	100100000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
0D7	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0D8	11	111111111	00100	00000	11	10	11	10	11	0	000	10	0	11100	11100
0D9	11	111111111	10100	00000	11	10	11	10	11	0	000	10	0	11100	11100
0DA	00	100100100	10110	00011	00	10	00	00	00	0	000	00	0	01001	00000
0DB	00	100100010	00011	00011	00	00	00	01	00	0	001	00	1	10101	00000
0DC	00	010001100	00100	00011	10	10	00	00	00	0	000	00	1	10000	00000
0DD	00	100011101	00000	00001	00	11	00	11	00	0	011	10	1	00000	00000
0DE	01	100011000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
0DF	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
100	00	000111111	00000	00000	00	00	00	00	00	0	000	00	0	01001	00000
101	00	110011010	00000	00011	00	11	00	00	01	0	000	00	1	00000	00000
102	00	011101100	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
103	00	110101101	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
104	00	011010011	00000	00000	01	10	00	00	00	0	100	11	0	00000	00000
105	01	011010100	00000	00011	00	00	00	00	00	0	000	00	0	00100	00000
106	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
107	00	001100110	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
108	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
109	00	001001011	00101	00011	00	00	00	00	01	0	000	00	0	00000	00000
10A	00	011101000	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
10B	00	110101100	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
10C	00	001001010	11011	00011	00	10	11	00	00	0	000	00	0	00000	00000
10D	00	111101001	00000	00011	00	00	00	01	01	0	000	00	1	00000	00000
10E	00	111101011	00000	00000	00	00	00	00	01	0	000	00	0	00000	00000
10F	00	100100101	11011	10010	00	00	11	00	00	0	000	00	0	00000	00000
110	00	011011111	00000	00000	00	00	00	00	00	0	000	00	0	01101	00000
111	10	100000000	00101	00110	00	00	11	01	00	0	000	00	0	00000	00000
112	00	100101100	00101	00110	00	00	00	01	00	0	000	00	1	00000	00000
113	10	100000000	10100	00011	01	10	11	01	00	0	000	10	0	00000	00000
114	00	001001000	01010	00110	01	10	00	01	00	0	000	10	0	00111	00000
115	00	001010110	01010	00110	00	00	00	00	00	0	000	00	0	00000	00000
116	00	010101110	11010	00011	00	10	00	00	00	0	000	00	0	01101	00000
117	10	100000000	00000	00000	01	10	11	00	00	0	000	10	0	00000	00000
118	00	011101100	01101	00110	00	00	00	01	00	0	000	00	1	00000	00000
119	00	111110011	01101	00000	00	00	00	00	00	0	000	00	0	01011	00000
11A	00	111110010	01101	00000	00	00	00	00	00	0	000	00	0	01011	00000
11B	00	111101001	00100	00011	00	00	00	00	00	0	000	00	1	00101	10000
11C	01	110110101	00100	00011	00	00	00	00	00	0	000	00	0	00000	00000
11D	00	100111010	01101	01101	00	00	00	00	00	0	000	00	0	00000	00000
11E	00	111101100	01110	01110	00	00	00	00	00	0	100	11	0	00000	00000
11F	00	111111111	00110	00011	00	10	00	00	00	0	000	00	0	01110	00000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
140	00	001111110	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
141	00	111101001	01101	00000	00	00	00	00	00	0	000	00	1	01011	00000
142	00	111001111	01100	01010	00	00	00	00	00	0	000	00	1	00000	00000
143	00	100000001	10000	00000	00	00	00	00	00	0	011	00	1	00000	11000
144	00	110001000	00000	00001	00	00	00	00	00	0	111	11	0	00000	01000
145	00	010110111	00010	00011	00	00	11	00	00	0	000	00	0	00000	00000
146	01	000011100	11011	00110	00	10	00	00	00	0	000	00	0	00100	00000
147	00	000011000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
148	00	010110110	01001	10100	00	00	10	00	00	1	000	01	0	00000	00000
149	00	010110101	01001	10100	00	11	00	00	00	0	000	00	0	11101	00000
14A	00	010110100	01010	01101	00	00	00	00	00	0	000	00	0	00000	00000
14B	00	011010111	00010	00011	00	00	00	00	00	0	000	00	0	00100	00000
14C	00	010101101	00000	00001	00	00	10	10	00	0	000	10	1	11111	00000
14D	00	001000010	00000	00000	00	00	00	10	00	0	000	10	1	11111	00000
14E	00	001010011	11011	01111	00	10	11	11	10	0	011	00	0	00000	00000
14F	00	001101100	01001	00001	00	00	00	11	10	0	000	00	0	00000	00000
150	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	01101	00000
151	00	010100001	01001	01001	00	10	11	00	00	0	000	00	0	00000	00000
152	01	000110000	00000	00000	00	11	00	10	00	0	000	10	1	11111	00000
153	00	101110011	00000	10010	00	00	00	01	00	0	000	00	0	00000	00000
154	00	000010000	00000	00101	00	00	11	10	00	0	001	10	1	10100	00000
155	00	000010000	00000	00000	00	00	00	10	00	0	001	10	1	10100	00000
156	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
157	00	010100111	00000	00010	00	00	00	01	00	0	100	00	1	10111	00000
158	00	010100110	01100	01110	11	10	00	00	00	0	000	10	0	00110	01000
159	00	110001000	01100	00010	00	00	01	00	00	0	000	00	0	00000	00000
15A	00	010100100	01001	00011	00	00	00	00	00	0	000	00	0	01111	00000
15B	00	010100011	01001	11110	00	00	01	00	00	1	000	00	0	00011	00000
15C	00	010100010	01001	11110	00	00	01	00	00	1	000	01	0	00000	00000
15D	01	110110110	01001	11100	00	00	01	00	00	0	000	00	0	00100	00000
15E	00	010100000	11011	00110	10	10	00	00	01	0	000	00	0	00000	00000
15F	00	010010001	01010	01001	00	00	10	00	00	0	000	00	0	00000	00000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
0E0	00	001111011	00000	00001	00	00	00	01	00	0	000	00	0	00000	00000
0E1	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0E2	00	100010111	11011	00110	00	10	00	11	00	0	001	10	1	10101	00000
0E3	00	011101101	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
0E4	00	001010111	00000	00001	00	11	00	00	10	0	011	00	1	00000	00000
0E5	00	001010111	00000	00001	00	11	00	00	10	0	011	00	1	10010	00000
0E6	01	100001000	00000	00000	00	00	01	00	00	0	000	00	0	00000	00000
0E7	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0E8	00	100110010	01011	00000	11	10	00	00	00	0	100	11	0	01011	00000
0E9	00	111101001	10100	00011	00	00	00	01	00	0	000	10	1	00000	00100
0EA	00	100010000	11010	10010	00	00	11	00	10	0	000	00	0	00000	00000
0EB	00	001110011	10100	00011	01	11	11	01	00	0	000	10	0	01100	00100
0EC	00	000101110	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
0ED	01	100110000	01000	00001	00	00	00	00	00	0	111	00	0	00000	00000
0EE	00	001101011	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
0EF	00	100001101	01011	00000	00	00	01	01	00	0	000	00	0	01011	00000
0F0	00	001001110	11010	01011	00	10	00	00	00	0	000	00	0	00000	00000
0F1	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0F2	00	100000101	10110	00011	00	00	01	10	00	0	011	00	1	01001	00000
0F3	00	100101100	00000	01110	00	00	00	01	00	0	000	10	1	00000	00000
0F4	00	011101011	00001	00011	00	00	00	01	00	0	000	00	1	00000	00000
0F5	00	011101011	00001	00011	00	00	00	01	00	0	000	00	1	10010	00000
0F6	01	011111000	00000	01001	00	00	11	00	00	0	000	00	0	00000	00000
0F7	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000
0F8	00	011001000	11000	00011	00	10	00	00	10	0	000	00	0	00000	00000
0F9	11	111111111	01100	00101	11	10	11	10	11	0	000	10	0	11100	10000
0FA	00	100000100	10100	00011	00	11	00	10	00	0	011	10	1	00000	00000
0FB	00	100000011	00000	00001	00	00	00	01	00	0	000	10	0	00000	00000
0FC	00	100110101	00000	00000	00	00	00	11	00	0	000	00	1	00000	00000
0FD	00	100000001	00000	00101	00	00	00	00	00	0	011	00	1	00000	10000
0FE	00	111101001	10100	00011	00	00	00	01	00	0	000	10	1	00000	00000
0FF	01	011010000	10000	00011	00	00	11	00	00	0	000	00	0	11011	00000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
120	00	000001001	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
121	01	110100101	11010	10010	00	10	00	00	01	0	000	00	0	00000	00000
122	00	110001011	01100	01111	00	00	10	00	00	1	000	01	0	00011	00000
123	11	111111111	00001	00011	11	10	11	10	11	0	000	10	0	11100	10000
124	00	011011010	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
125	00	011010111	01010	01101	00	10	00	00	00	0	000	00	0	00100	00000
126	00	000011001	00000	00101	00	10	00	00	00	0	000	00	0	00000	00000
127	11	001111110	00000	00001	00	00	11	00	00	0	000	10	0	00000	00000
128	00	111101001	00001	00011	00	00	00	00	00	0	000	00	1	00101	10000
129	00	101100100	10100	01111	00	10	00	00	00	0	000	00	0	00000	00000
12A	00	001000001	00001	01011	10	10	00	00	00	0	000	00	0	00000	00000
12B	01	000110100	00010	00011	00	00	00	00	00	0	000	00	0	00100	00000
12C	00	001000011	00000	01001	00	00	00	01	00	0	111	00	0	00000	00000
12D	00	011001110	00000	01010	00	00	10	00	00	0	000	00	0	00000	00000
12E	00	100010101	11011	10010	00	10	00	00	10	0	000	00	0	00000	00000
12F	00	100010101	11011	10010	00	10	00	00	00	0	000	00	0	00000	00000
130	00	010111111	00000	00000	00	00	00	00	00	0	000	00	0	01101	00000
131	00	011001101	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
132	00	011011100	01010	00001	00	00	00	00	00	0	000	00	1	00100	00000
133	00	000111000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
134	00	011001010	11000	00011	00	10	00	00	01	1	000	00	0	00000	00000
135	00	111010011	01101	00110	00	00	00	01	10	0	000	00	1	00000	00000
136	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
137	00	011000111	10100	00110	00	10	00	00	00	1	000	00	0	00000	00000
138	00	011000110	11000	00011	00	00	00	00	01	0	000	00	0	00000	00000
139	00	111000011	00101	00011	00	00	11	00	10	0	000	00	1	00000	00000
13A	00	011000100	11000	00011	00	00	00	00	01	0	000	00	0	00000	00000
13B	00	110110011	00111	00011	00	00	00	00	10	1	011	00	1	00000	00000
13C	00	011000010	11000	00011	00	00	11	00	01	0	000	00	0	00000	00000
13D	00	110100011	00111	00011	00	00	00	00	10	1	011	00	1	00000	00000
13E	01	000001010	10100	00011	00	00	00	00	00	0	011	00	0	00000	00000
13F	00	001010011	00001	10010	00	10	11	11	10	0	011	00	0	00000	00000

ADD.	SNA	NAN	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
160	00	000111001	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
161	11	111111111	01010	00001	11	10	11	10	11	0	000	10	0	11100	00000
162	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
163	00	000110011	10000	00011	00	10	00	00	00	0	000	00	0	00000	00000
164	00	010011010	01001	01110	00	00	00	00	00	0	100	11	0	10100	00000
165	00	101111101	00000	00000	00	00	00	00	00	0	100	00	1	10100	00000
166	00	000100010	01111	00001	00	00	00	00	00	1	000	00	0	00000	00000
167	00	000100011	01111	00001	00	00	01	00	00	1	000	00	0	00000	00000
168	11	011011110	00000	00000	00	00	00	00	00	0	000	10	0	00000	00000
169	01	010011100	10110	01000	00	00	00	00	00	0	000	00	0	00100	00000
16A	00	000110000	00010	00011	00	00	00	00	00	0	111	10	1	11111	00000
16B	00	010010011	01010	10010	00	00	00	00	00	0	000	00	0	00100	00000
16C	00	010010010	01001	11010	00	00	00	00	00	0	000	00	1	00000	10100
16D	11	111111111	01010	01111	11	10	11	10	11	0	000	10	0	11100	00000
16E	00	010010000	10100	10010	00	10	00	00	00	0	000	00	0	00000	00000
16F	00	010000001	10100	00110	00	11	00	00	01	0	000	00	0	00000	00000
170	00	000011001	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
171	00	101010010	10100	00011	01	10	00	01	00	0	000	10	0	00000	00000
172	00	111110111	10100	00011	01	10	00	01	00	0	000	10	1	00000	00000
173	00	010001000	10100	00011	11	00	11	01	00	0	000	10	0	00000	00000
174	00	110001001	01010	10010	00	00	00	00	00	0	000	00	0	00100	00000
175	00	101000010	10100	00011	01	10	00	01	00	0	000	10	0	00000	00000
176	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
177	00	010000111	01000	00001	00	11	00	00	00	0	000	00	1	00000	00000
178	11	111111111	00000	00000	11	10	11	10	11	0	000	10	0	11100	11100
179	00	010000011	01100	01001	11	10	00	00	00	0	000	10	0	00000	00000
17A	00	000100111	00000	01010	00	11	10	00	00	0	000	00	0	00000	00000
17B	00	000100110	00000	00001	00	11	00	00	01	0	000	00	0	00000	00000
17C	00	111001000	10100	00011	00	00	00	01	00	0	000	00	1	00000	00000
17D	11	111111111	01100	00010	11	10	11	10	11	0	000	10	0	11100	00000
17E	00	010000000	10100	00110	00	00	00	00	01	0	111	00	0	00000	00000
17F	00	110001110	01000	01010	00	00	00	00	00	0	000	11	0	11100	00000

ADD.	SNA	NA N	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
180	00	000101000	00000	00000	00	11	00	00	00	0	100	00	0	10011	00000
181	00	000100000	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
182	00	001010000	11010	10010	00	10	00	00	00	0	000	00	0	00000	00000
183	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
184	00	001111010	00100	00011	00	00	00	00	00	0	111	00	1	10110	00000
185	00	110001000	00100	00011	00	00	00	00	00	0	111	10	0	00000	00000
186	01	001010100	00011	00000	00	00	10	00	00	0	000	00	0	00100	00000
187	01	001100100	00111	00000	00	00	01	00	00	1	000	00	0	00100	00000
188	00	001110110	01101	00110	00	00	00	01	00	0	000	00	1	00000	00000
189	10	100000000	00000	01110	00	00	00	11	01	0	000	10	1	00000	00000
18A	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
18B	00	000011011	00000	01010	00	00	10	00	10	0	000	00	0	00000	00000
18C	00	001110010	11011	00110	00	10	00	00	00	0	000	00	0	01010	00000
18D	00	001101110	01000	00101	00	11	00	00	00	0	000	00	0	00000	00000
18E	01	000100100	00001	00111	00	00	00	00	00	0	000	00	0	00100	00000
18F	00	001100001	11011	00100	00	10	00	00	00	0	000	00	0	00000	00000
190	00	000011001	00000	00101	00	10	00	00	00	0	000	00	0	00000	00000
191	00	110111110	00000	00010	10	10	11	00	00	0	000	00	0	00000	00000
192	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
193	01	000000010	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
194	00	001101010	01100	00000	00	00	00	01	00	0	000	00	0	01011	00000
195	00	001101000	10110	00011	00	00	00	10	00	0	011	00	1	00000	00000
196	00	011101001	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
197	00	001100111	10100	00011	00	11	00	10	00	0	011	10	1	00000	00000
198	00	111101001	00000	00001	00	00	00	01	01	0	000	10	1	00000	00000
199	00	001011110	00000	01101	00	10	00	00	00	0	000	00	0	00000	00000
19A	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
19B	01	001111000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
19C	01	001011100	10100	00111	00	00	00	11	10	0	000	00	0	00100	00000
19D	00	011111011	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
19E	00	001100000	01000	01001	00	10	00	00	00	0	000	00	0	00000	00000
19F	00	001010001	10100	00110	00	10	00	00	01	0	111	00	0	00000	00000

ADD.	SNA	NA N	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
1C0	00	111101000	00000	00000	00	00	00	00	00	0	100	00	0	10011	00000
1C1	00	000110111	01001	00000	00	00	00	00	00	0	000	00	0	00000	00000
1C2	01	000101100	00000	01101	00	10	11	00	00	0	000	00	0	00100	00000
1C3	00	010000101	11011	01011	00	10	11	00	00	0	000	00	0	01101	00000
1C4	01	011110001	01101	00001	00	11	00	00	00	0	000	00	0	00000	00000
1C5	00	110001000	01100	01110	01	00	00	00	00	0	000	10	0	00000	00100
1C6	00	011001100	00000	00000	01	00	11	00	00	0	000	10	0	00000	00000
1C7	00	000101001	00000	00001	10	10	00	00	00	0	111	00	0	00000	00000
1C8	00	000110110	01010	00000	00	00	00	00	00	0	000	00	0	00000	00000
1C9	00	000111011	11011	00110	00	10	00	00	00	0	000	00	0	00000	00000
1CA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1CB	00	001110001	11011	01111	00	10	00	00	00	0	000	00	0	00000	00000
1CC	00	001111111	00000	00101	00	00	11	00	00	0	000	00	0	00000	00000
1CD	00	010011000	11011	01111	00	10	11	00	00	0	000	00	0	01101	00000
1CE	00	010101100	01000	00001	00	00	00	00	00	0	000	10	0	11111	00000
1CF	00	110111000	10100	00011	00	00	00	01	00	0	000	10	0	11111	00000
1D0	00	000101111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1D1	00	001101010	01100	00000	00	00	00	01	00	0	000	00	0	00000	00000
1D2	01	001111100	00000	01101	00	10	10	00	00	0	000	00	0	00100	00000
1D3	00	000110010	00000	00000	00	00	00	00	00	0	000	00	0	00010	00000
1D4	01	011101010	01001	00110	00	00	00	01	00	0	000	10	0	00111	00000
1D5	01	011101010	01001	00110	00	00	00	01	00	0	000	10	0	10010	00000
1D6	11	010011110	00000	00000	00	00	00	00	00	0	000	10	0	00000	00000
1D7	01	011011000	00000	00001	01	00	11	00	00	0	111	10	0	00000	00000
1D8	01	010000100	11010	00111	00	00	00	00	00	0	000	00	0	00100	00000
1D9	00	000010111	10100	00000	00	00	00	00	01	0	000	00	0	00000	00000
1DA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1DB	00	001110000	11010	10010	00	10	00	00	00	0	000	00	0	00000	00000
1DC	01	010011000	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
1DD	00	001111000	11011	01111	00	10	11	00	00	0	000	00	0	00000	00000
1DE	01	000001100	00000	00101	00	00	00	00	00	0	000	00	0	00000	00000
1DF	00	001001001	00000	00001	00	00	00	00	00	0	111	00	0	00000	00000

ADD.	SNA	NA N	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
1A0	01	100111010	01000	00001	00	11	00	00	10	0	000	00	0	00000	00000
1A1	01	101110100	11010	00111	00	00	00	00	00	0	000	00	0	00100	00000
1A2	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1A3	01	001100010	00000	00000	01	10	00	00	00	0	000	10	0	00000	00000
1A4	00	001011010	00101	00000	00	10	00	00	00	0	000	00	0	00000	00000
1A5	00	100101011	01001	00110	00	11	00	01	00	0	000	00	1	00000	00000
1A6	00	010010110	11011	01011	00	10	00	00	00	0	111	11	0	01110	01000
1A7	01	001011000	00000	00000	00	00	01	00	00	0	111	00	0	10111	00000
1A8	01	100101010	01001	00000	00	00	00	01	00	0	000	10	0	00111	00000
1A9	00	100111010	01101	01010	00	00	00	00	00	0	000	00	0	00000	00000
1AA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1AB	00	000010011	01010	10000	00	00	00	00	00	0	000	00	0	00000	00000
1AC	01	001010010	00000	00001	00	00	00	11	10	0	011	10	0	00000	00000
1AD	00	010110000	00000	00011	00	00	00	01	01	0	000	00	0	00000	00000
1AE	10	111111111	01001	00001	11	10	11	00	10	0	000	11	0	11100	00000
1AF	00	010110001	01000	01001	00	00	00	01	01	0	000	00	0	01101	00000
1B0	00	001011011	11011	00011	00	10	11	00	00	0	000	00	0	00000	00000
1B1	00	001000110	01000	01101	00	11	00	00	00	0	000	00	0	00000	00000
1B2	01	000111100	00000	01101	00	10	00	00	00	0	000	00	0	00100	00000
1B3	01	010111000	11011	00010	00	10	11	00	00	0	000	00	0	01101	00000
1B4	01	011110010	01000	00001	00	00	00	00	10	0	000	00	0	00000	00000
1B5	01	100001011	01010	00000	00	00	00	00	00	0	000	00	0	00000	00000
1B6	11	010111110	00000	00000	00	00	00	00	00	0	000	10	0	00000	00000
1B7	01	000101011	00000	00001	00	11	00	00	10	0	011	00	1	00000	00000
1B8	00	101100010	00000	00001	00	00	00	00	00	0	011	00	1	00000	00000
1B9	00	000111010	00000	00010	00	00	00	01	00	0	100	00	1	10110	00000
1BA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1BB	00	000011010	00000	01010	00	00	10	00	00	0	000	00	0	00000	00000
1BC	00	001011000	11010	00011	00	00	11	00	00	0	111	11	0	11100	00000
1BD	00	010110011	00000	00000	00	00	00	10	00	0	000	10	1	11111	00000
1BE	00	001000000	00000	01001	00	10	00	00	00	0	000	00	0	01110	00000
1BF	01	000010100	10100	00111	00	11	00	00	00	0	000	00	0	00100	00000

ADD.	SNA	NA N	A	ADL	C	M	Q	S	P	CT	B	SEQ	R	GP	CR
1E0	00	100000000	00000	00001	00	00	00	00	01	0	000	00	0	00000	00000
1E1	01	110010101	00001	00011	00	00	00	00	00	0	000	00	0	00000	00000
1E2	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1E3	00	000100001	00000	00101	00	10	00	00	00	0	000	00	0	00000	00000
1E4	01	000011010	00000	00000	00	00	00	00	10	0	000	00	0	00000	00000
1E5	00	000110011	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
1E6	01	001001100	00000	01101	00	10	00	00	00	0	000	00	0	00100	00000
1E7	00	001001100	00000	00000	00	11	00	00	00	0	000	00	0	00000	00000
1E8	00	000010110	10100	00110	00	10	10	00	00	0	000	00	0	00000	00000
1E9	01	001110100	11010	00111	00	00	00	00	00	0	000	00	0	00100	00000
1EA	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1EB	00	000110011	11010	01111	00	10	00	00	00	0	000	00	0	00000	00000
1EC	00	000010010	11010	00011	00	10	11	00	00	0	000	00	0	00000	00000
1ED	01	001000100	00010	00000	00	00	10	00	00	0	000	00	0	00100	00000
1EE	00	111101001	10100	00011	00	00	00	01	00	0	000	10	0	00000	01100
1EF	01	000010000	00000	00000	00	00	01	10	00	0	001	10	1	10100	00000
1F0	00	100000000	00000	00001	00	00	00	00	01	0	000	00	0	00000	00000
1F1	01	110000101	01001	01111	00	00	10	00	00	0	000	00	0	00011	00000
1F2	00	010011111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1F3	00	001111111	00000	00000	00	00	00	00	00	0	000	00	0	00000	00000
1F4	00	011000001	00000	00000	00	00	00	11	10	0	000	10	0	00000	00000
1F5	00	001100011	00000	00001	00	00	00	01	01	0	000	00	0	00000	00000
1F6	00	010010111	00000	01110	00	00	00	00	00	0	111	10	0	00000	00000
1F7	00	010111011	00100	00011	00	10	00	00	00	0	111	00	1	10111	00000
1F8	00	000110110	10100	00011	00	10	00	00	00	0	000	00	0	00010	00000
1F9	00	000000111	01001	00000	00	00	00	00	11	0	000	00	0	00010	00000
1FA	00	000111110	10100	00011	00	10	00	00	00	0	000	00	0	00010	00000
1FB	00	000000101	01001	00110	00	00	00	00	11	0	000	00	0	00010	00000
1FC	01	001101100	00001	00111	00	00	00	11	10	0	000	00	0	00100	00000
1FD	01	011000000	00000	00011	00	10	00	01	01	0	000	00	0	00000	00000
1FE	00	100101101	11011	10010	00	11	11	00	00	0	000	00	0	00010	00000
1FF	00	100100101	11011	10010	00	00	11	00	00	0	000	00	0	00010	00000

2.48 INSTRUCTION WORD LOGIC

The instruction word logic (Figure 2-8AA) comprises the K register and the instruction decoder. This logic stores and decodes the program instruction word for use by the Microprogram Control.

2.49 K -- Instruction Register

This 16-bit register holds the instruction word which is obtained from memory via the GP-Bus BIO lines. The instruction-word format is given in Section I (paragraph 1.62). Four type 74175 IC chips are used as a straight 16-bit buffer register, with both high and low outputs available for decoding. Bus data BIO00-15R are loaded by CLK at the trailing edge of BP, when GFETCH is active. GFETCH is a general-field command bit (Table 2-3).

2.50 Instruction Decoder

The instruction decoder uses a programmable logic array (PLA) as the main decoding logic. The PLA provides an address code to the microinstruction control-store for 96 different input combinations. A set of gate-logic decoders in parallel with the PLA are used for ineffective branches and FPP instructions without attached FPP. The PLA0, PLA1 control logic controls some of the PLA and the gate-logic decoding. These instruction-decoder outputs are used with the Microprogram Control and are described in paragraphs 2.37 and 2.40 (Microinstr. Instr. Word, M.S. Pointer).

2.51 Logic gates are used to decode the K register into basic fields for controlling the PLA decoder and gate decoders. These gates may indicate when the instruction R1 or R2 fields contain 0 or 15, or when a branch condition is indicated and verified. Some bits of the K register are used as direct control bits for the PLA decoder. Complete PLA decoding is shown in Table 2-6.

2.52 PLA ROM maps are provided (Table 2-7) to show the PLA output code (as a two-bit hexadecimal number) produced for the various instructions or addressing types. The PLA output code relates directly to the control-store ROM address used by the microprogram control.

Table 2-6 Instruction Decoder PLA

	Data Input	Output	Comments	Group
	I14 ←→ I0	F8 ←→ F1		
	PLA0 PLA1 K0 K1 K2 K3 K4 R1E0 R1E15 K9 K10 R2E0 R2E15 K15	RAD 1 —— 8		
1	LL------------	LLLLLLLL	Inhibit mode	
2	HHLLHLLH------	LLLLLLLL	INH, RIT, HLT	Privileged Instruction Detection
3	HHLHLL-L------	LLLLLLLL	I/O instructions	
4	HHLHHH-L------	LLLLLLLL	RER, WER	
5	HHH-----H-----	LLLLLLLL	Format 1 with R1=15	
6	HHHLLLL--LHLH-	LLLLLLLL	LD, ST with R2=15	
7	HHHLHHH--LHLH-	LLLLLLLL	ML, MS with R2=15	
8	HH-HHH-H----HL	LLLLLLLL	MVF, MVB, RTN with R2=15	
9	HHHLHHHH------	LLLLLLLL	TS, TS	
10	HHHHLHLL------	LLLLLLLL	EL, ES	
11	H-LH----------	L-------	K1	Format 0 (T8)
12	H-L-H---------	-L------	K2	
13	H-L--H--------	--L-----	K3 Generals	
14	H-L---H-------	---L----	K4	
15	H-L----H------	-------L	OR1	
16	H-LLLLL-------	LL------	LDK	
17	H-LLLLH-------	-------L	ABK	
18	H-LHLH--------	-------L	RF, RB	
19	H-LHLLL--HLH-L	-------L	WMP	
20	H-LLHHH----H-L	-----L--	Shift with n=0	
21	H-H-----------	------L-	K0	Addressing Routines
22	H-H------H----	-L------	K9 Generals	
23	H-H-------H---	--L-----	K10	
24	H-H--------H--	---L----	OR2	
25	H-H----------H	-------L	K15	
26	H-HLLLH--LHL--	-------L	OPC-1 T3 Store	
27	H-HHHLL-------	L------L	OPC-12 Char. Instr.	
28	H-HLLLL--LHLHL	L-------	LDR T3 15R2	
29	H-HLHHHL-LHLHL	-----L--	MLR T3 15R2	
30	H-HHLL-L------	L------L	Floating Point	
31	H-H-------LL---	L------L	T1 routine	Format 1 (T1-7)
32	H-H-LLL------H	L------L	ST, TS, MS	
33	H-HLHLLH-----H	L------L	CM	
34	H-HLHHH-------	L------L	OPC-7, K0	
35	H-HHLHLL------	L------L	EL, ES	
36	H-HHHLH------H	L------L	CC	
37	H-HHLH-H-LHL--	L------L	DAR*, DSR *	
38	H-HHLH-H-H----	L------L	DA, DS	
39	H-HHHHL------L	-----L--	Return	
40	HHHHHHL------L	---L----	Return, User	

	Data input I14 → I0 (PLA0 PLA1 K0 K1 K2 K3 K4 R1E0 R1E15 K9 K10 R2E0 R2E15 K15)	Output F8 ↔ F1 (RAD 1 — 8)	Comments		Group
41	LHHH----------	L---L---	K0.K1		
42	LHH-H---------	-L------	K0.K2		
43	LHH--H--------	--L-----	K0.K3	Generals	Execution
44	LH----H-------	---L----	K4		
45	LHH------H---H	-----L--	K0.K9.K15		
46	LH-----H------	-------L	0R1		
47	LHHLLHH--LL--H	-----L-L	NGR		
48	LHHLH-L--LL--H	-----LL-	TM, TNM		
49	LH-LLLH-------	-------L	AB		
50	LH-LLLL-------	----L---	LD, ST		
51	LHHLLLL--LH-HH	------L-	STR with R2=15		
52	LHHHLH-H-LHH--	------L-	DAK, DSK		
53	--HHLH-H-LL---	-----L--	DAR, DSR	(ADD. also)	
54	LHHHHLL--LL---	-------L	ECR		
55	LH-HHHL--LL---	-----L--	CR, EX		
56	LHHHHHH-------	-------L	C1		
57	LH-LHHH-------	----L---	ML, MS, TL, TS, Shift		
58	LHHLHHHL-LHLHH	------L-	MSR with R2=15		
59	LHHLHHH--LHH--	L-------	MLK		
60	LHLLHHH--H----	------L-	Shift K9		
61	LHLLHHH---H---	-----L--	Shift K10		
62	LH-HHHL-----HL	------L-	RTN with R2=15		
63	LHHHLLHL-LL---	------L-	FFX, FFL		
64	H--LHHLH------	LLLLLLLL	XRK	0R1	
65	H-LHLL-H------	LLLLLLLL	I/O instr.	0R1	
66	H--HLHH------H	LLLLLLLL	RB	K15	
67	H-LHHLL-------	LLLLLLLL	OPC-12	K0	
68	H-HL-----LHH-H	LLLLLLLL	K0.$\overline{K1}$.K15	(T2)	
69	H-HLLLH--H---H	LLLLLLLL	AB	K15	
70	H-HLLLH---H--H	LLLLLLLL			
71	H-HLLHHH-LL--H	LLLLLLLL	SUR	K15	
72	H-HLHL-H-----L	LLLLLLLL	AN, 0R	0R1 , $\overline{K15}$	Trap
73	H-HLHLLH-LL---	LLLLLLLL	ANR	0R1	
74	H-HLHLHH-----H	LLLLLLLL	0RS	0R1	Detection
75	H-HLHLH--LL--H	LLLLLLLL	0RR	K15	
76	H-HLHHH--LL---	LLLLLLLL	OPC-7	(T1)	
77	H-HLHHHH-LHH--	LLLLLLLL	MLK	0R1	
78	H-HHL--H-----H	LLLLLLLL	OPC-8 to 11	0R1 .K15	
79	H-HHL----LHH-H	LLLLLLLL	OPC-8 to 11	T2.K15	
80	H-HHL-LL-LL---	LLLLLLLL	OPC-8, 10	T1.$\overline{0R1}$	
81	H-HHLL-L-LHH--	LLLLLLLL	OPC-8, 9	T2.$\overline{0R1}$	
82	H-HHLHLL-LHH--	LLLLLLLL	DAK	$\overline{0R1}$	
83	H-HHHLLH------	LLLLLLLL	OPC-12	0R1	
84	H-HHHLL--LHH-H	LLLLLLLL	OPC-12	T2.K15	
85	H-HHHLHH-H---H	LLLLLLLL	CC	0R1	Trap Detection
86	H-HHHLHH--H--H	LLLLLLLL			
87	H-HHHHLL-----L	LLLLLLLL	OPC-14	$\overline{0R1}$, $\overline{K15}$	
88	H-HHHH-H-LL--L	LLLLLLLL	OPC-14,15	0R1 , $\overline{K15}$, T2	
89	H-HHHH---LHH-L	LLLLLLLL	OPC-14,15	$\overline{K15}$, T2	
90	H-HHHH-H-H---L	LLLLLLLL	OPC-14,15	0R1 , $\overline{K15}$, K9	
91	H-HHHHHH-LL---	LLLLLLLL	C1R	0R1	
92	H-HHHHH--LHH--	LLLLLLLL	C2	T2	
93	H-HHHHHH-LHL-L	LLLLLLLL	C1R *	0R1	
94	LHH-------H--H	-----L--	K0 K10 K15		Execution Misc.
95	LHHHLLHL-LL--H	-----L--	FFX		
96	LHHHHHL-H----H	------L-	CF with R1=15		

Table 2-7 PLA ROM Map

PLA Output

F: | 8 7 6 5 | 4 3 2 1 |

MSB LSB

Example: Addressing type RT5S = /43 = $0100\ 0011_2$

MSB \ LSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		NO JUMP	NO JUMP P+2	NO FPP												
1		ABK														
2	ADK	ADKP	RT3	RT3S			RTN									
3	SUK	SUKP	RT2	RT2S			RTN USER									
4	ANK	HLT RIT INH	RT5	RT5S												
5	ORK	ENB SMD LKN	RT4	RT4S												
6	XRK		RT7	RT7S												
7	SM $n \neq 0$	DSH $n \neq 0$	RT6	RT6S	SH $n \neq 0$	DSH $n \neq 0$										
8	CIO OTR	WMP		RT1				RT1D								
9	SST TST INR			RT1P				RT1DP								
A		RF	RT3B	RT3C				RT3BM								
B		RB		RT2C												
C	LDK	LDKP		RT5C												
D	CWC	CWCP		RT4C												
E	WER	MVF		RT7C												
F	RER	MVB		RT6C												

ADDRESSING MODE

MSB \ LSB	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0									LC	LCP			ST	STP	STD	STDP
1		AB							SLA SLN	CLA CLN	SLL SLC	DLL DLC	SRA SRN	DRA DRN	SRL SRC	DRL DRC
2	AD	ADP			ADS	IM										
3	SU	SUP			SUS	C2 NGR										
4	AN				ANS	CP	TM									
5	OR				ORS											
6	XR				XRS		TNM									
7									ML	TL			MS	TS	MSRO	
8									FL	MU			FS			
9									FO	DV	FFL		FOS		FFX	
A									EL	DAR* DA		DAK	ES		DAR	
B										DSR* DS		DSK			DSR	
C									LC	ECP			SC			
D									CW	CWP			CC			
E										RTN		RTN A15	CF	EX	CF 15R1	
F									MLK	C1				C1S		

EXECUTION MODE

2.53 DATA HANDLING LOGIC

The Data Handling Logic (Figure 2-4) comprises the arithmetic logic unit (ALU), data and address storage and handling registers, and the logic for the data path. The basic components of the Data Handling Logic are :

ALU -- Arithmetic Unit	2.59	2-8 GG
M -- CPU Working Register (multi-in) : load C ; load Q.	2.64	2-8 GG
D Selector : ALU direct ; ALU exchange character ; ALU shift right ; BIO.	2.61	2-8 HH
L -- Data Register (multi-in) : Load D direct ; load D shift left. (operation result; output buffer)	2.63	2-8 HH
C Selector : D0-15 or 8-15; BIO0-15 or 8-15; INTAD 0-5; ASRO-7.	2.65	2-8 JJ
Q -- Shift Register : load; shift right; shift left.	2.67	2-8 JJ
S -- Address Register/Counter	2.71	2-8 KK
A - Bus Selection	2.72	2-8 LL
IPL -- Initial Program Loader	2.76	2-8 LL
P -- Program Register/Counter	2.77	2-8 LL
A0-A15 -- Scratchpad	2.79	2-8 MM
PSW - Program Status Word	2.84	
PLR -- Priority Level Register	2.85	2-8 NN
CR -- Condition Register	2.86	2-8 NN
GF -- General Flip-Flops	2.89	2-8 PP

The Data Handling Logic performs parallel processing of 16-bit words. Double-length operations are provided for processing of 31-bit words. Two 16-bit data paths loop through the arithmetic unit (ALU) : one loop for operand A and one loop for operand B.

2.54 Operand A is supplied by the A-Bus. A-Bus control logic selects which of the sources on the A-Bus is to be used as the operand. The various sources are connected by open-collector gates so only the selected inputs activate the bus. Operand A includes the scratchpad A0-A15, program counter P, program-status word PSW, initial program loader IPL, constant selector, and control-panel selector which supplies control-panel commands and system commands. Operand B is supplied by CPU working register M, shift register Q, and data selector C. The Q register is used as a shift register in double-length operations and as an auxiliary operand accumulator.

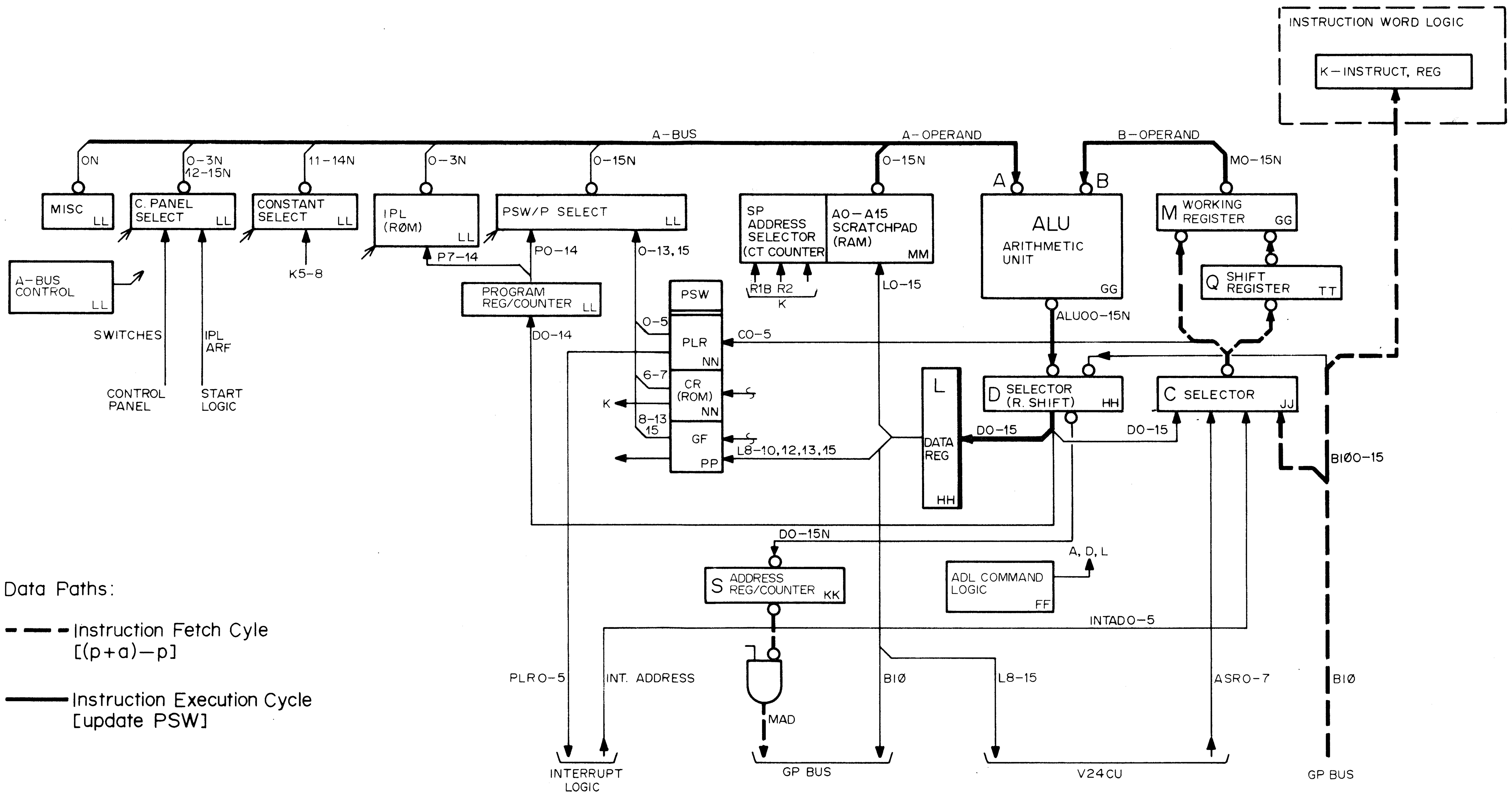
2.55 Data Path

Figure 2-4 shows the data path for instruction fetch operations (heavy dashed line) and instruction execution operations (heavy solid line). The fetch operation loads the instruction word into the instruction register K, and the eight least significant bits, via C, into the Q and M registers. The execution operation processes both operands through the ALU and outputs the result via D to the L register. Actual processing of data may be done by registers other than the ALU : the D selector and L and Q registers can perform shift operations in the data-path loop. With D selector at BIO direct, there is a direct loop from L, via BIO and D, to L. Scratchpad read, rewritten as same clock : path with D (exch. char.; shift right) and L (shift left).

2.56 A,D,L Command

The A,D,L command logic (Figure 2-8FF) controls the arithmetic operations by providing simultaneous command signals to the ALU, the D-selector, and the multiple-input L register. Thirty-two 8-bit command codes are stored in the ADL read-only memory (ROM). The command codes are selected by microinstruction bits μ ADL0-4 and some internal flags : Q15, Q16, FSIG, and M00N.

2.57 The first three address bits (μADL0-2) are used for direct ROM addressing (inputs E,D,C); they also control a dual 4-input multiplexer (type 9309) which selects the last two address bits (inputs B,A) of the ROM. ROM inputs B and A may be the last two microinstruction bits (muADL3-4, inverted)



Data Paths:

- - - Instruction Fetch Cyle [(p+a)—p]

—— Instruction Execution Cycle [update PSW]

Note: AA, BB etc. refers to logic diagrams 2-8AA, BB etc.

Figure 2-4 Data Handling Logic

or the flag bits Q15, Q16, FSIG, M00N, as follows :

Multiplexer Select Input		Multiplexer Output to ADL Command ROM	
μADL: 0, 1, 2	RALUA: 1N, 2N	RADL3	RADL4
0 0 0	1 1	μADL3N	μADL4N
0 0 1	1 1	μADL3N	μADL4N
0 1 0	1 1	μADL3N	μADL4N
0 1 1	1 1	μADL3N	μADL4N
1 0 0	1 1	μADL3N	μADL4N
1 0 1	1 0	Q15	Q16
1 1 0	0 1	μADL3N	FSIGN
1 1 1	0 0	μADL3N	DIVFLAGN

$$\text{DIVFLAG} = \text{FSIGN} \oplus \text{M00N}$$

Table 2-8 shows the complete microinstruction/flag control of the ADL ROM, the ROM contents, and the output functions.

2.58 The ADL-ROM outputs ALUS0N-3N are used as the ALU selection inputs S0-4 (Figure 2-8 GG). The ALUCO16 signal is sent to the ALU and the ALU look-ahead-carry unit as a carry control signal. The microinstruction bit μADL4 is also sent directly to the ALU CE input to select logic functions (CE-high) or arithmetic functions(CE-low). The ADL-ROM outputs DS0-1 (combined with microinstruction field bit GMULTI) are the control signals for the four-input D selector (Figure 2-8 HH). The D00DN signal that is generated as an auxiliary part of the ADL command logic is used as one input to the D selector, bit 0; this logic is described with the D selector. The ADL-ROM output LSEL is sent to the L register (Figure 2-8HH) to select direct load (LSEL-low) or shift left (LSEL-high).

Table 2-8 ADL Command ROM

MUADL 0 1 2 3 4	FLAGS	ADDRESS	ALUS0N ALUS1N ALUS2N ALUS3N ALUCO16 DS0N DS1N LSEL	MNEMO	FUNCTION
0 0 0 1 1		0 0	0 0 0 0 0 1 1 0	ALUA	Operand A
0 0 0 1 0		0 1	0 0 1 0 0 1 1 0	AOB	Logical OR of A and B
0 0 0 0 1		0 2	1 0 1 0 0 1 1 0	ALUB	Operand B
0 0 0 0 0		0 3	1 0 0 1 1 1 1 0	AMB	Arithmetic Subtract A minus B
0 0 1 1 1		0 4	0 1 1 0 0 1 1 0	AXB	Logical XOR of A and B
0 0 1 1 0		0 5	0 1 1 0 0 1 1 0	APB	Arithmetic Addition A plus B
0 0 1 0 1		0 6	0 1 0 1 0 1 1 0	BINV	Operand B Inverted
0 0 1 0 0		0 7	0 1 1 0 1 1 1 0	APLB1	Arithmetic Addition with Carry A plus B plus 1
0 1 0 1 1		0 8	0 0 0 0 0 0 1 0	ACR	Operand A characters crossed on Selector D
0 1 0 1 0		0 9	0 0 1 1 1 1 1 0	ZERO	ALU output is zero
0 1 0 0 1		0 A	1 0 1 0 0 0 1 0	BCR	Operand B characters crossed on Selector D
0 1 0 0 0		0 B	0 1 1 1 1 1 1 0	AANDB	Logical AND of A and B
0 1 1 1 1		0 C	0 0 0 0 0 1 0 0	ASHR	Operand A shifted right one position
0 1 1 1 0		0 D	0 0 0 0 0 0 0 0	DBIO	BIO Receivers selected on D
0 1 1 0 1		0 E	1 0 1 0 0 1 0 0	BSHR	Operand B shifted right one position
- - - - -		0 F	1 1 1 1 1 1 1 1	-	Not used
- - - - -		1 0	1 1 1 1 1 1 1 1	-	Not used
1 0 0 1 0		1 1	1 1 0 0 0 1 1 0	TWOA	Operand A added to itself
1 0 0 0 1		1 2	0 0 0 0 0 1 1 1	ASHL	Operand A shifted left one position
1 0 0 0 0	Q15 Q16	1 3	1 1 0 0 0 1 1 1	FORA	Operand A four times (two in ALU and SAL)
1 0 1 0 0	0 0	1 4	0 0 0 0 0 1 0 0		Operand ASHR
1 0 1 0 0	0 1	1 5	0 1 1 0 0 1 0 0	MULTI	Operands A plus B and SHR
1 0 1 0 0	1 0	1 6	1 0 0 1 1 1 0 0		Operand A minus B and SHR
1 0 1 0 0	1 1	1 7	0 0 0 0 0 1 0 0		Operand ASHR

MUADL 01234	FLAGS	ADDRESS	ALUS0N ALUS1N ALUS2N ALUS3N ALUCO16 DS0N DS1N LSEL	MNEMO	FUNCTION
1 1 0 1 0	FSIG 1	1 8	0 1 1 0 1 1 1 0	DADD	Arithmetic addition A plus B plus 1
1 1 0 1 0	FSIG 0	1 9	0 1 1 0 0 1 1 0	DADD	Arithmetic addition A plus B
1 1 0 0 0	FSIG 1	1 A	1 0 0 1 0 1 1 0	DSVB	Arithmetic subtract A minus B minus 1
1 1 0 0 0	FSIG 0	1 B	1 0 0 1 1 1 1 0	DSVB	Arithmetic Subtract A minus B
1 1 1 1 0	FSIG ⊕ M00 1	1 C	0 1 1 0 0 1 1 1	DIVSH	Addition and SHR 2 (A+B) + Q0
1 1 1 1 0	FSIG ⊕ M00 0	1 D	1 0 0 1 1 1 1 1	DIVSH	Subtract and SHR 2 (A-B) + Q0
1 1 1 0 0	FSIG ⊕ M00 1	1 E	0 1 1 0 0 1 1 0	DIVALU	Addition A + B
1 1 1 0 0	FSIG ⊕ M00 0	1 F	1 0 0 1 1 1 1 0	DIVALU	Subtract A - B

2.59 ALU -- Arithmetic Unit

The arithmetic unit (Figure 2- 8 GG) uses four 4-bit ALU chips (type 74181) to perform arithmetic and logic functions on two 16-bit operands. The ALU chips are controlled by microinstruction bit μADL4 and the ADL-command logic (using bits μADL0-4). Bit μADL4 is applied to the CE input to select arithmetic (CE-low) or logic (CE-high) functions. All control bits are active high, while the data in and out (in this application) are active low. The ALU functions are shown in the following table. The control-signal logic, and the complete arithmetic unit functions combined with the D and L register operations, are discussed in the previous section (A,D,L Command) and shown in Table 2-8.

ALU Function Table

μADL4	ADL-Control ROM: ALUCO16	ADL-Control ROM: ALUSO-----3				Operation	
CE	CIN	S3	S2	S1	S0		
0	0	1	0	0	1	A plus B	ARITHMETIC
0	0	1	0	1	1	A or B	ARITHMETIC
0	0	1	1	0	0	A plus A	ARITHMETIC
0	1	0	0	0	1	A and B	ARITHMETIC
0	1	0	0	1	1	Zero	ARITHMETIC
0	1	0	1	1	0	A minus B	ARITHMETIC
1	-	0	1	0	1	B inverted	LOGIC
1	-	1	0	0	1	A ⊕ B	LOGIC
1	-	1	0	1	0	B	LOGIC
1	-	1	1	1	1	A	LOGIC
1	-	1	0	0	1	A ⊕ B + 1	LOGIC

Control signals are all active high. Functions are for active-low data in and out.

A carry-look-ahead unit (type 74182) is used in conjunction with the ALU chips to provide anticipated carry across all four ALU circuits.

2.60 The zero-ALU (0ALUN) output is generated in subtract mode when both operands are equal (A=B). A zero-ALU, left-half (0ALUL) output sets the stack overflow indicator (STOV). Since stack operations forbid a memory address equal-to or less-than 128 (100 hexadecimal), the left-most character of the address (bits 0-7) must have at least one bit set. When S is loaded from the ALU during a stack operation, the all-zero ALU output (A=B) indicator STOV is therefore gated by GFSTOV to set GF flip-flop PIF (Figure 2- 8 PP).

2.61 D Selector

The D selector (Figure 2- 8 HH) is used as a control element in the arithmetic loop. The four D-selector functions (ALU direct, exchange ALU characters left and right, ALU shifted right, and BIO direct) are selected by the ADL-command logic (Table 2-8). The ADL-command signals DS0, DS1 operate the D selector as follows :

S1 S0 / DS1, 0			Function
0 0 :	ALU 00-15N		ALU direct
0 1 :	ALU 08-15N	ALU 00-07N	Exchange ALU characters
1 0 :	D00DN	ALU 00-14N	ALU shift right
1 1 :	BIO 00-15AN		BIO direct
D	00 07	08 15	N

The D selector comprises eight type 9309 (dual 4-input multiplexer) circuits. These circuits provide both the true and complementary outputs of the active-low inputs. The inverted D-selector output (active-high) is sent to the L register, C selector, and P register/counter. The true output (active low) is sent to the S register/counter.

2.62 The ALU shift-right mode is used for shift-right and multiplication operations. The input bit D00DN used during this mode is provided by the ADL command logic (Figure 2-8 FF). Shift Right Arithmetic (SRA) and Shift Right Normalize (SRN) instructions use the ALU sign bit, ALU00. The Double Shift Right Circular (DRC) instruction uses Q15 for D00DN. Multiplication instructions set D00DN directly when negative overflow is detected (NOVF), or use the ALU sign bit ALU00 when positive overflow is not detected (POVF). The D00DN selection is shown in the following table.

GCRDSR.K9.ALU00	Shift Right, Arithmetic or Normalize (SRA,SRN)
GCRDSR.K08Q15	Double Shift Right Circular (DCR)
μADL0.NOVF	Multiply and negative overflow
μADL0.POVN.ALU00	Multiply and positive overflow

2.63 L -- Data Register

The L register (Figure 2-8 HH) is a 16-bit, multiple-input register comprised of four type 74298 chips. The L register is used for storing the data from the ALU (or BIO) to be output to the scratchpad (A0-15) or the GP bus. The input data to the L register is provided by the D selector. The L register is in the operand-A loop between the ALU and the scratchpad. Since the L register is loaded by the leading-edge of BP, an operand can be read from and re-written into the scratchpad on the same clock cycle. The two inputs to the L register are D direct and D shifted left. These two operating modes for L are selected by the ADL command logic (Table 2-4) bit LSEL :

LSEL	Data Source		Function
0	D 00-15		D direct
1	D 01-15	Q00	D shifted left
L	00————14	15	

Bit Q00 is shifted into the least-significant position during a Division instruction when the new remainder is loaded.

The L register stores active-high data with no inversion between input and output.

2.64 M -- CPU Working Register

The M register (Figure 2-8 GG) is a 16-bit, multiple-input register comprised of four type 74298 chips. The M register is used as the CPU working register in the operand-B data path. Microinstruction control of the register uses bits μMLOAD and μMSEL to load the register and to select the C selector or Q shift register for inputs, as follows :

Enable Clock μMLOAD	Select C or Q μMSEL	Mnemonic	Data Source
0	-	No Op	Off; previously stored data available at output.
1	0	MYC	C 00 15
1	1	MYQ	Q 00 15
			M 00 15

The M register stores active-low data with no inversion between input and output. The μLOAD signal enables the CLMN input to the clock to store the selected data on the leading edge of BP.

2.65 C Selector

The C selector (Figure 2-8 JJ) uses two, quad 2-input multiplexers (type 8234) for the bit 0-7 selection, and four, dual 4-input multiplexers (type 9309) for the bit 8-15 selection. The result is a four-input multiplexer with two 16-bit sources and two 8-bit sources. Three microinstruction selection bits are used in such a manner that the two 16-bit sources (D and BIO) can be selected whole or as short constants (bits 8-15 only), or either of the 8-bit sources (INTAD and ASR) can be selected. The sources are selected by microinstruction bits μC0, μC1, and GCSLEN as follows:

(μC0+$\overline{\text{C1}}$) CLEFS0	μC0	μC1	GCSLEN	Mnemonic	Source	
00-07		08-15				
S0	S1	S0	S1			
1	0	0	1	CALU	D00 ———	——— 15
0	0	1	1	CBIO	BIO00 ———	——— 15
1	1	0	1	CLUR		D08 ——— 15
1	1	1	1	CIOR		BIO08 ——— 15
1	1	0	0	CLUR		INTAD0 ——— 5 | 0 | 0
1	1	1	0	CIOR		ASR0 ——— 7
1	0	0	0	(CALU)	D00 ——— 07	INTAD0 ——— 5 | 0 | 0
0	0	1	0	(CBIO)	BIO00 ——— 07	ASR0 ——— 7
				C	00N ——— 07N	08N ——— 15N

The last two possibilities are not used.

2.66 The C selector inverts all selected data from active high to active low. The 0's supplied for the not-used bit positions are therefore output as inactive highs. The C-selector outputs are sent to the Q and M registers in the operand-B data path. The six least-significant bits (C00N-05N) are also sent to the priority level register during an interrupt routine.

2.67 Q Shift Register

The Q register (Figure 2-8 JJ) uses four type 74194 circuits as a 16-bit, left/right shift register. Q may be parallel-loaded from the C selector, shifted right (with ALU15N as a serial input), or shifted left (with the selected bit, Q15DN, as a serial input). Loading and shifting is done on the leading-edge of clock pulse BP (BPQ) according to the function selected by microinstruction bits μQ0, μQ1 as follows :

μQ0, Q1		Mnemonic	Function
0	0	QI	Hold contents (No Op)
0	1	SLQ	Shift Left Q
1	0	SRQ	Shift Right Q
1	1	QYC	Parallel load Q from C

In the operand-B arithmetic loop, Q can be loaded from the M register, via ALU--D--C, and reloaded into the M register on the same BP clock pulse. No Q-register shifting may be performed on a single-clock-pulse loop since Q must be in the parallel-load mode.

2.68 The shift-right serial input, ALU15N, is used for double-length, shift-right operations. The shift-left serial input, Q15DN, is used for double-length left circular (DLC) shift and Divide instructions. Q15DN is derived from a 74151A circuit by the selection-inputs K008N, M00N, and ALU00N. For DLC shift instructions (K08N active low), the serial input is provided by microinstruction bit GCSELN if M00N is active; the serial input is a logic zero (high) if M00N is inactive (high). For Divide instructions (K08N high and RALUA1N (μADL 0.1) = 0), Q15DN is produced from the quotient bit ALU00 ⊕ M00. See following diagram.

K08N	M00N	ALU00N		
S2	S1	S0	Q15DN =	
L	L	L	GCSELN	DLC
L	L	H		
L	H	L	H	
L	H	H	H	
H	L	L	(μADL0. 1)	Divide
H	L	H	H	
H	H	L	H	
H	H	H	(μADL0. 1)	

L = low = active

2.69 The two least-significant bits of Q are used to control the next-cycle operation of a Multiply execution. One bit position (Q16) of a fifth 74194 is used to save the previous contents of Q15 at the end of the Multiply cycle, while Q14 is shifted into Q15. Q00, Q15N, and Q16N are used by the ADL command logic (Figure 2-8 FF) for arithmetic operation control. The three other bit positions of the fifth 74194 are used in the Bus Controller logic (Figure 2-8 TT). This fifth 74194 is used in the Hold (GCRFNUN = 0) or Parallel-Load (GCRFNUN = 1) modes only.

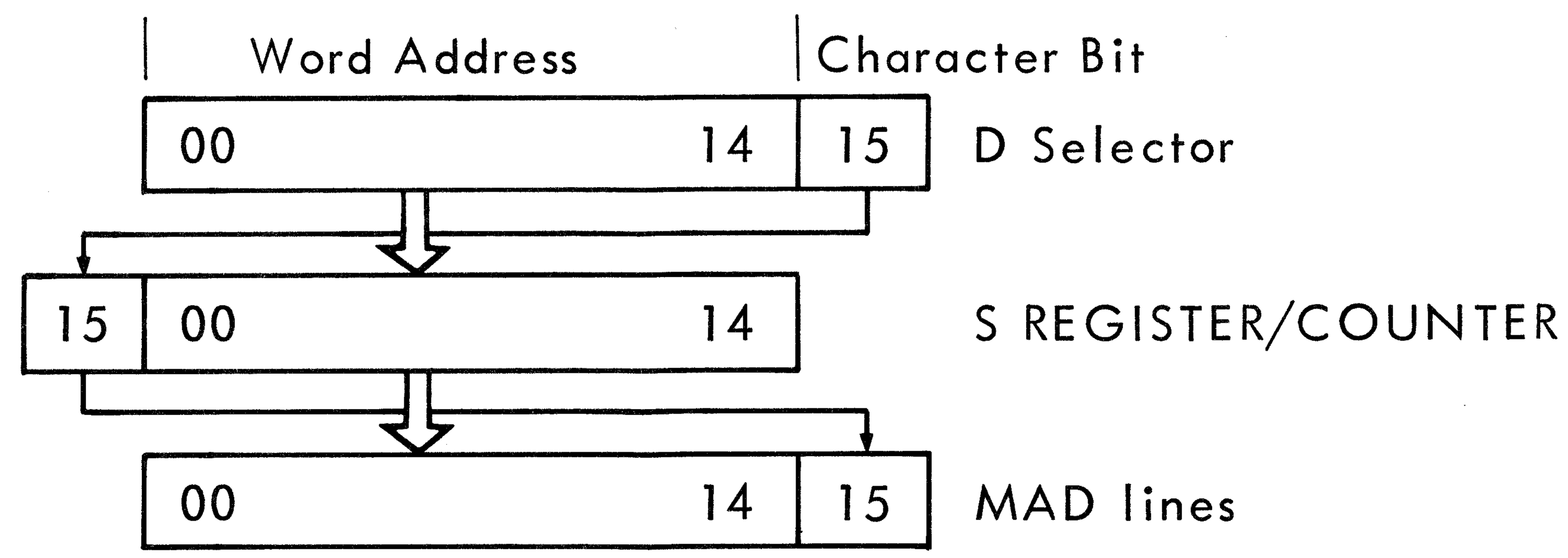
2.70 Q-register bit Q00/Q00N is used for microinstruction addressing in the Flag mode (paragraph 2.28 and Figure 2-8 DD). Q00 is also used as a serial input to the L register during a left-shift L operation.

2.71 S -- Address Register/Counter

The S register/counter (Figure 2-8 KK) is a 16-bit synchronous up/down counter using four type 74S169 circuits. Fourteen bits of the S register/counter are used to store the memory word address, and increment or decrement the address in one-word steps. Loading or counting takes place on the leading edge of the BP pulse, according to the following control from microinstruction bits μS0 and μS1:

μS0, S1	Mnemonic	Function
0 0	SI	Inhibit counting or loading
0 1	SYD	Load S0-16N with D0-15N
1 0	SP2	S plus 2 (count down)
1 1	SM2	S minus 2 (count up)

Since S is loaded with negative (low-level active) data, the count-down mode increments the value of the contents and count-up decrements. The S register/counter is loaded from the D selector, as shown in the following diagram :



D00-14, representing the memory word address, is loaded into the least-significant position of S, while D15, representing the character address, is loaded into the most-significant position. S-register counting then affects a two-character address change (S plus 2, or S minus 2) for a single increment or decrement of the counter. The character indicator from the most-significant position in S is then switched back to the least-significant position on the memory address lines (MAD00-15). The S register/counter works with low-level data, and does not invert.

2.72 A-Bus Selection

The A-Bus selection logic determines which source is to be used as operand-A of the arithmetic operation. Control of the A-bus selection is by microinstruction bits μA0-4 (Table 2-9). The possible sources for operand-A on the A-bus are :

- Scratchpad A0-A15
- Initial program loader IPL
- Program counter P

- Program status word PSW
- Instruction field (R1 or R3)
- Constant 2_{10} (/002) or 16_{10} (/010)
- Control-panel/System
- A-bus equals zero

All operand-A sources onto the A-bus are via active-low, open-collector output circuits (8234 multiplexers, 6200 ROM, 7489 RAM, and one 7403 gate). All these circuits provide a high-level output to the bus when they are not selected. This high is then over-ridden by any active low signal from a selected source.

2.73 The scratchpad (Figure 2-8 MM) is selected directly by μA0 to the chip-select inputs of the scratchpad logic. The specific scratchpad address (A0 to A15) is then selected by μA2-4 controlling the scratchpad-address logic. When the scratchpad is selected, μA1 differentiates between reading and writing the selected address.

2.74 Any A-bus source except the scratchpad is selected by the A-bus control logic (Figure 2-8 LL). The inverted μA0 input to the 1-of-10 decoder chip (type 9301) forces the decoder higher than the seven possible outputs that are connected; this effectively keeps this part of the A-bus selection switched off when the scratchpad is selected. If μA0 is active (high), and μA1-3 are all low, the non-connected output-0 is decoded and no A-bus source is selected; operand-A is then equal to zero. The active-low outputs from the 9301 decoder select either the IPL (output-1) or one of the type 8234 multiplexers. These multiplexers have their A inputs selected by an active low to S1 only, or their B inputs selected by an active low to S0.

2.75 Double-Word Trap. If a Trap occurs during a double-word operation, (P) must be decremented an extra position. The DWIF flip-flop (Figure 2-8 LL) is set for all double-word instructions. If a Trap routine (Figure 1-12) is initiated, the first microinstruction (address /0FF) produces GAEXL which is gated by DWIF to set A-bus bit A00N. This increments the normal microinstruction address by one to select /12E rather than /12F.

Table 2-9 A-Bus Selection

Microinstruction μA0	A1	A2	A3	A4	Control decoder (9301) output active	Mnemonic	Function
							Read scratchpad accumulator :
0	0	0	0	0		ARA0	A0
0	0	0	0	1		ARA1	A1
0	0	0	1	0		ARA2	A2
0	0	0	1	1		ARA15	A15
0	0	1	0	0		ARR1	R1 (Addressed by K5-7,8)
0	0	1	0	1		ARR2	R2 (Addressed by K11-14)
0	0	1	1	0		AR1215	(Addressed by K12-15)
0	0	1	1	1		ARCT	(Addressed by CT counter)
							Write scratchpad accumulator :
0	1	0	0	0		AWA0	A0
0	1	0	0	1		AWA1	A1
0	1	0	1	0		AWA2	A2
0	1	0	1	1		AWA15	A15
0	1	1	0	0		AWR1	R1 (Addressed by K5-7,8)
0	1	1	0	1		AWR2	R2 (Addressed by K11-14)
0	1	1	1	0		AW1215	(Addressed by K12-15)
0	1	1	1	1		AWCT	(Addressed by CT counter)
1	0	0	0	-	--	AZ	A-bus equals zero
1	0	0	1	-	1	AIPL	Read IPL
1	0	1	0	-	2	AEP	Read P
1	0	1	1	-	3	APSW	Read PSW
1	1	0	0	-	4	AEN	Read content of K5-8
1	1	0	1	1	5	ATW0	Constant 2_{10} (/002) onto A-bus
1	1	0	1	0	5	ATEN	Constant 16_{10} (/010) onto A-bus
1	1	1	0	-	6	APUP	Control panel inputs onto A-bus
1	1	1	1	-	7	ASYS	System inputs onto A-bus

2.76 IPL -- Bootstrap

The IPL circuit is a 64-word hardware bootstrap stored in a type 6200 Read Only Memory (ROM) of 256 four-bit words. The ROM contents are regrouped into 16-bit words when they are loaded into main memory by the IPL-microprogram routine (Figures 1-9, 1-10). The IPL/Bootstrap logic (Figure 2-8LL) is addressed by the P-register/counter (bits P07-14) and is accessed through the A-bus. The A-bus-selection command code is µA0,1,2,3 = 1 0 0 1 for the read-IPL routine. Table 2-10 is a listing of a bootstrap after being loaded into memory 00-63. Use of the bootstrap for initial program loading is discussed in Section III. Other bootstrap ROMs may be provided with different CPUs.

2.77 P -- Program Register/Counter

The P register/counter (Figure 2- 8 LL) is a 16-bit synchronous up/down counter using four type 74S169 circuits. The fourteen least-significant positions of P are loaded from the D selector with the 14-bit word address of the program. The P register is also used as an internal counter : in this case, the most-significant bit (which isn't loaded from D) is sent to the flag-selection logic as PM1 (P minus 1). PM1 is used by microinstruction Addressing, Flag mode (paragraph 2.26) to indicate the end of a multiple word instruction.

2.78

Loading or counting of P takes place on the leading edge of the BP pulse (BPP) according to the following control from microinstruction bits µP0 and µP1 :

µP0, P1	Mnemonic	Function
0 0	PI	Inhibit counting or loading
0 1	PYD	Load P0-14 with D0-14
1 0	PM2	P minus 2 (count down)
1 1	PP2	P plus 2 (count up)

The program word-address in P is accessed through the A-bus. The A-bus selection code µA0,A1,A2,A3 = 1010 for read-P. The eight least-significant address bits (P07-14) are also applied directly to the addressing inputs of the IPL ROM, for use when the IPL is selected.

Table 2-10 List of Bootstrap Loaded in Memory

Step	P Reg.	Mem. Cont.		Instruct.		
000				IDENT BEBOOT		
001			*			
002			*			
003			*	DISPLAY THE KEYS AS FOLLOWS:		
004			*			
005			*	BITS MEANING		
006			*	0=1 IPL LOADED FROM ASR, FORMAT 4* 4		
007			*	1=1 DISK, THEN		
008			*	2=1 MOVING HEADS		
009			*	2=0 FIXED HEADS		
010			*	3=1 I/O BUS		
011			*	3=0 MULTIPLEX		
012			*	4 to 7 BOU LINE (4 RIGHTMOST BITS)		
013			*	8=1 MULTI DEVICE CONTROLLER		
014			*	8=0 SINGLE DEVICE CONTROLLER		
015			*	9=1 CDD DISK		
016			*	10 to 15 DEVICE ADDRESS		
017			*			
018			*			
019			*			
020			*	DESCRIPTION		
021			*			
022			*	BEBOOT LOADS ONE RECORD ONTO LOCATION /80 THEN START AT /84		
023			*	THE RECORD IS THE SECTOR # 1 IF DISK, OR 254 CHARACTERS OF THE		
024			*	INPUT DEVICE, LEADING NULL CHARACTERS IGNORED		
025			*			
026			*			
027			*			
028			*	USED REGISTERS :		
029			*			
030			*	A1 BOU LINES, NOT TO BE DESTROYED IF BOOT IS CALLED AGAIN		
031			*	A2 ADDR OF INR INSTRUCTION		
032			*	A3 ADDR OF CIO INSTRUCTION(WHICH IS DESTROYED AND NEEDS TO BE		
033			*	RESTORED IF BOOT IS CALLED AGAIN)		
034			*	A4 ADDR OF SST INSTRUCTION		
035			*	A5 MULTIPLEX: CONTENTS OF 1ST WORD TO BE SENT TO EXT REGISTER		
036			*	IO BUS : CHARACTER COUNT, INITIALIZED AT 254 AND DECREMENTED		
037			*	A6 MULTIPLEX: CONTENTS OF 2ND WORD TO BE SENT TO EXT REGISTER		
038			*	(LOADING ADDR)		
039			*	IO BUS: ADDR OF NEXT CHAR TO BE LOADED, INIT AT /80 AND		
040			*	INCREMENTED		
041			*	A7,A8 WORK REGISTERS		
042			*	A9 WORK REGISTER		
043			*	A10 TO A14 NOT USED		
044			*	A15 CONTAINS THE KEYS'VALUE		
045			*			
046			*			
047			*			
048			*			
049			*			
050			*			
051			*	28/8/73 ARE FIXED:		
052			*	MULTIPLEX DLBLE WORD INITIALIZATION		
053			*	SST INST FOR MULTIPLE DEVICE CONTROLLER		
054			*			
055				EJECT		
056				ADRG	0	
057			*			
058			*			
059			BOOT	EQU	*	
060			*	INITIALIZE REGISTERS		
061	0000	0200	F	LDK	A2,INR	ADDR OF INR INSTRUCTION
062	0002	0300	F	LDK	A3,CIO	ADDR OF CIO INSTRUCTION
063	0004	0400	F	LDK	A4,SST	ADDR OF SST INSTRUCTION
064			*			EXTRACT DEVICE ADDR AND INIT I/O COMMANDS
065	0006	861E		LDR	A6,A15	
066	0008	263F		ANK	A6,/3F	DEVICE ADDR
067	000A	9629		ADRS	A6,A2	INITIALIZE I/O INSTRUCTIONS
068	000C	962D		ADRS	A6,A3	
069	000E	9641		ADS	A6,HIO	
	0010	0000	F			
070			*			EXTRACT CONTROLLER ADDR AND INIT WER INST
071	0012	871E		LDR	A7,A15	
072	0014	3FC8		SLC	A7,8	MULTI OR SINGLE DEVICE CONTROLLER
073	0016	5600	F	RF(6)	INIT20	SINGLE ONE
074	0018	260F		ANK	A6,/F	MULTIPLE ONE
075			INIT20	EQU	*	INITIALIZE MULTIPLEX DBLE WORD:
076	001A	9631		ADRS	A6,A4	SST INSTRUCTION
077	001C	3E41		SLL	A6,1	
078	001E	9641		ADS	A6,WER1	SET UP WER INSTRUCTIONS
	0020	0000	F			
079	0022	9641		ADS	A6,WER2	
	0024	0000	F			

Step	P Reg.	Mem. Cont.		Instruct.		
080			*			LOAD A1 WITH BOU CONTENTS
081	0026	811C		LDR	A1,A7	
082	0028	0550		LDK	A5,80	MULTIPLEX DOUBLEWORD: LOAD 80 CHAR INTO
083			*			LOCATION /80
084	002A	0680		LDK	A6,/80	
085			*			CHECK IF DISK
086	002C	3FE7		SRC	A7,7	
087	002E	5600	F	RF(6)	NODISK	NO
088			*			FIXED HEADS ?
089	0030	3FC1		SLC	A7,1	
090	0032	5600	F	RF(6)	NOSEEK	YES
091	0034	0103		LDK	A1,3	SEEK ZERO
092	0036	41C0	CIO	CIO	A1,1,0	CIO SEEK ZERO
093			NOSEEK	EQU	*	
094	0038	811E		LDR	A1,A15	
095	003A	3966		SRL	A1,6	SECTOR NUMBER
096	003C	213C		ANK	A1,/3C	
097	003E	8520		LDKL	A5,/80CD	1ST WORD OF MULTIPLEX FOR DISK DEVICE
	0040	80CD				
098			NODISK	EQU	*	
099			*			EXECUTE WER, WHATEVER THE CHANNEL IS
100			WER1	EQU	*	
101	0042	7500		WER	A5,0	
102			WER2	EQU	*	
103	0044	7601		WER	A6,1	
104	0046	F031		EXR*	A4	SST
105	0048	F02D		EXR*	A3	
106	004A	5C06		RB(4)	* -4	
107	004C	871E		LDR	A7,A15	
108	004E	3F43		SLL	A7,3	
109	0050	5600	F	RF(6)	SST	MULTIPLEX
110			*			
111			*			IO BUS
112			*			
113	0052	8194		LDR	A9,A5	IF A9=A5 IGNORE LEADING CHAR.
114			INR	EQU	*	
115	0054	4F00		INR	A7,0,0	READ ONE CHAR
116	0056	5C04		RB(4)	* -2	
117	0058	E994		CWR	A9,A5	LEADING CHAR?
118	005A	5400	F	RF(4)	INR10	NO
119	005C	27FF		ANK	A7,/FF	CHECK IF NULL
120	005E	580C		RB(0)	INR	YES, IGNORE
121			*			NO, CHECK IF 4* 4
122			*			
123			INR10	EQU	*	
124	0060	879E		LDR	A15,A15	4* 4
125	0062	5600	F	RF(6)	STORE	NO 8+8
126	0064	3F44		SLL	A7,4	YES, 4* 4
127	0066	809C		LDR	A8,A7	SAVE LEFT BITS
128	0068	F029		EXR*	A2	READ NEXT CHARACTER
129	006A	5C04		RB(4)	* -2	
130	006C	270F		ANK	A7,/F	GET 4 RIGHT MOST BITS
131	006E	9702		ADR	A7,A8	
132			STORE	EQU	*	
133	0070	E739		SCR	A7,A6	STORE CHAR
134	0072	1601		ADK	A6,1	NEXT CHAR ADDR
135	0074	1D01		SUK	A5,1	COUNT DONE ?
136	0076	5924		RB(1)	INR	NO
137			*			YES
138			*			
139	0078	4180	HIO	CIO	A1,0,0	
140			*			
141			STATUS	EQU	*	
142	007A	4FC0	SST	SST	A7,0	
143	007C	5C04		RB(4)	* -2	
144	007E	0F84		AB	/84	
145			*			
146			*			
147			*			
148				END	BOOT	

SYMBOL TABLE

BOOT	0000	A	INR	0054	A	CIO	0036	A	SST	007A	A
HIO	0078	A	INIT20	001A	A	WER1	0042	A	WER2	0044	A
NODISK	0042	A	NOSEEK	0038	A	INR10	0060	A	STORE	0070	A
STATUS	007A	A									

ASS.ERR. 00000

:EOF

2.79 A0-A15 -- Scratchpad

These sixteen 16-bit accumulators comprise 15 working registers (A0-A14) and the stack pointer (A15). The working registers are used as an operand for some instructions. Fifteen registers (A1-A15) are program addressable. The scratchpad (Figure 2-8 MM) is controlled by the A-bus selection commands, µA0-4. The complete command code for scratchpad addressing and function is given in Table 2-9. For the write-scratchpad modes, the contents of the L register (L00-15) are loaded in the selected address (A0-A15). For read-scratchpad modes, the inverted contents of the selected address are gated onto the A-bus.

2.80 The 16-word scratchpad comprises four 64-bit read/write memory chips, type 7489. These circuits provide data inversion (active high input, low output) and open-collector outputs with a high-level off-state when µA0 to the chip-select input is 1 (scratchpad not selected). The scratchpad write-enable input, WSPN, from the start logic (Figure 2-8 RR), is active from the center of clock time T6 to T7, if µA0,A1 = 01.

2.81 The four-bit scratchpad address (SPA0-3) is selected through four type 74151A eight-input multiplexers. The first four sets of inputs are wired to +5 or 0 volts to obtain the direct address code of A0, A1, A2, or A15. The inputs 4, 5, or 6 select different fields of the instruction word to address the scratchpad : the R1/R3 field is provided by buffered K05-08/K05-07 (K08 is given via K0008BUFN if K00 = 1, indicating format-1 instruction); the R2 field is provided by K11-14N; the K12-15 field gives the number of shifts (in complement form). The seventh input selects the CT counter for addressing successive scratchpad registers.

2.82 The five-bit CT counter (Figure 2- 8 MM) is used to count repeat cycles for the sequensor control, or (for P857) to address sequential scratchpad registers during Multiple Load or Store instructions. The counter comprises a type 74161 BCD counter for the four least-significant bits (CT0). A buffer at the output of the counter maintains the scratchpad address while the counter is being incremented. The buffer copies CT at the trailing edge of each AP pulse.

Counter operation is controlled by microinstruction bit μCT and general-field bit GCTLDN, as follows :

μCT (H=1)	GCTLDN (L=1)	Function
0	0	No change
0	1	Load CT1-4 from SPA0-3; CT0 from K11
1	0	Count CT at each T3N

The counter is preset to 16 (CT0-4 = 10000) by KRYN during each instruction Fetch cycle. Other initial counts less than 16 may be loaded into CT1-4 via the scratchpad address selector.

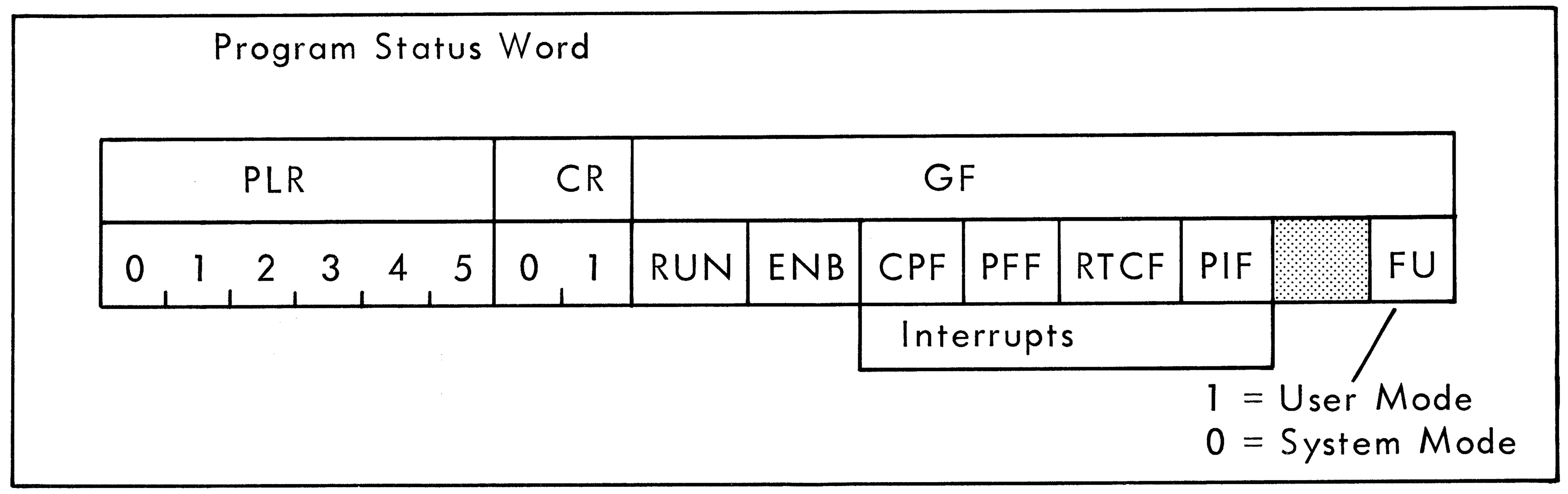
2.83 The number of shifts to be performed are loaded in complement form from the K12-15 field and effectively decremented by the counter operation. The counter is loaded with (R2) for the Double Arithmetic instructions in T1D addressing mode, enabling the least-significant operand word to be processed first. For DAR and DSR instructions in T1D addressing mode, (R2) + 1 addressing is obtained by the CT counter providing the least-significant address bits to the scratchpad. Multiply and Divide instructions count from the preset count of 16. The CTEND signal is generated when the count reaches 31 (CT0-4 = 11111). CTEND is used by the Sequensor to end the repeat cycles.

2.84 PSW -- Program Status Word

The PSW is a 16-bit status word comprising the priority level register (PLR), condition register (CR), and general flip-flops (GF). The complete PSW is saved in the stack, via the A-bus, during the following :

- Interrupt routine
- Trap routine
- Page Fault
- Call Function (CFR)
- Halt (HLT)
- Inhibit Interrupt (INH)
- Reset Internal Interrupt (RIT)
- Enable Interrupt (ENB)
- Link to Monitor (LKM)
- Set Mode (SMD)

During a Return (OPC14) instruction, System Mode (A15), the PLR, the CR, and the ENB and FU-bits of GF are restored from the stack to the PSW; the other bits of GF are not changed by the Return. During a Return instruction, User Mode ($\overline{\text{A15}}$), only CR is restored from the stack.



RESTORED FROM STACK AFTER:	
RTN (A15)	RTN ($\overline{\text{A15}}$)
PLR	PLR
CR	
ENB	(ENB unchanged)
FU	

2.85 PLR -- Priority Level Register

The PLR (Figure 2-8 NN) is a 6-bit register that stores the priority level code of the running program for the Interrupt logic. This register, comprising two type 74175 circuits, is preset to priority level 63 by the system master-clear signal MCLN. The register output, PLR0-5, is used by the Interrupt logic (paragraph 2.97) for comparison with Interrupt Requests. If an Interrupt Request is accepted, the Interrupt Routine is initiated : the contents of the PLR are saved in the stack with the rest of the program status word, and the PLR is loaded with the new program's interrupt level by INTAD0-5 via the C selector.

2.86 CR -- Condition Register

The 2-bit condition register (Figure 2-8 NN) indicates the result of various CPU operations. The condition register flip-flops are updated at each BP clock time (BPQ). The CR input conditions (Table 2-11) are selected by a pair of type 74175A 8-input multiplexers, with the selection controlled by microinstruction bits μCR0-2.

Table 2-11. CR Input Conditions

CR INPUT SELECTION

Selection µCR0,1,2	Mnemonic	Input to CR: CR0	Input to CR: CR1	Function
0 0 0	CRNC	CR0	CR1	No change
0 0 1	CRRTN	BIO06R	BIO07R	CR load during Return
0 1 0	CRIO	Timeout	ACBN	Programmed channel loading
0 1 1	CRFLO	FLOCR0	FLOCR1	Floating Point loading
1 0 0	CRLOG	ROMCR1	ROMCR5	Logic operation loading
1 0 1	CRADD	ROMCR2	ROMCR6	Addition operation loading
1 1 0	CRSUB	ROMCR3	ROMCR7	Subtraction operation loading
1 1 1	CRCMP	ROMCR4	ROMCR8	Compare operation loading

CR INPUT ROM CODE

	0ALUL	0ALUR	ALU00N	A00N	M00N	LOGIC CR1,5	ADD CR2,6	SUB CR3,7	COMP CR4,8	
	0	0	0	0	0	1 0	1 0	1 0	1 0	
	0	0	0	0	1	1 0	1 0	1 0	1 0	
	0	0	0	1	0	1 0	1 0	1 1	0 1	
	0	0	0	1	1	1 0	1 1	1 0	1 0	
	0	0	1	0	0	0 1	1 1	0 1	0 1	
	0	0	1	0	1	0 1	0 1	1 1	1 0	
	0	0	1	1	0	0 1	0 1	0 1	0 1	
	0	0	1	1	1	0 1	0 1	0 1	0 1	
	0	1	0	0	0	1 0	1 0	1 0	1 0	
	0	1	0	0	1	1 0	1 0	1 0	1 0	
	0	1	0	1	0	1 0	1 0	1 1	0 1	
	0	1	0	1	1	1 0	1 1	1 0	1 0	
	0	1	1	0	0	0 1	1 1	0 1	0 1	
	0	1	1	0	1	0 1	0 1	1 1	1 0	
	0	1	1	1	0	0 1	0 1	0 1	0 1	
	0	1	1	1	1	0 1	0 1	0 1	0 1	
	1	0	0	0	0	1 1	1 1	1 1	1 1	not used
	1	0	0	0	1	1 1	1 1	1 1	1 1	not used
	1	0	0	1	0	1 1	1 1	1 1	1 1	not used
	1	0	0	1	1	1 1	1 1	1 1	1 1	not used
	1	0	1	0	0	0 1	1 1	0 1	0 1	
	1	0	1	0	1	0 1	0 1	1 1	1 0	
	1	0	1	1	0	0 1	0 1	0 1	0 1	
	1	0	1	1	1	0 1	0 1	0 1	0 1	

1 : OVF + Enable input	0ALUL	0ALUR	ALU00N	A00N	M00N	LOGIC CR1,5	ADD CR2,6	SUB CR3,7	COMP CR4,8	
	1	1	0	0	0	1 1	1 1	1 1	1 1	not used
	1	1	0	0	1	1 1	1 1	1 1	1 1	not used
	1	1	0	1	0	1 1	1 1	1 1	1 1	not used
	1	1	0	1	1	1 1	1 1	1 1	1 1	not used
	1	1	1	0	0	0 0	1 1	0 0	0 0	
	1	1	1	0	1	0 0	0 0	1 1	1 0	
	1	1	1	1	0	0 0	0 0	1 1	0 1	
	1	1	1	1	1	0 0	0 0	0 0	0 0	
1	-	-	-	-	-	1 1	1 1	1 1	1 1	

Input 0 copies the previous CR contents (no change). Inputs 1-3 select conditions for Return, programmed channel loading, and Floating Point loading.

2.87 Inputs 4-7 are selected for arithmetic operations, and the input conditions are provided by a type-7488A read only memory (ROM). One specific pair of the eight ROM output bits is selected for the type of arithmetic operation being performed (logic, addition, subtraction, or compare). The 2-bit condition codes are selected from one of 32 ROM addresses by : the sign bit of the two ALU operands, A00N and M00N; the sign of the result, ALU00N; and the zero contents of the result for both left and right characters, 0ALUL and 0ALUR.

2.88 An overflow condition, set into the OVF flip-flop, sets CR to 3 (11_2) by switching off the enable input to the ROM to produce an all-Ones output. The overflow condition is tested by two series-connected multiplexers (the 8-input 75141 and the quad 2-input 74157), as follows :

74157 Input	Select A00	74151 In	Select K01	ALU00	A01	Function	
(+5V) 1	0 (11)	0	0	1	1	overflow	Shift
µMLOAD	1 (10)	2	0	0	1	OK	
µMLOAD	0 (11)	1	0	1	0	OK	
(+5V) 1	1 (10)	3	0	0	0	overflow	
GCRDSRN	0 (11)	4	1	1	1	overflow	Divide
OVFMUN	1 (10)	5	1	1	0	OK	
(+5V) 1	0 (11)	6	1	0	1	OK	
GCRDSRN	1 (10)	7	1	0	0	overflow	

OVFMUN = GMULTI.$\overline{\text{ALUS0}}$.M00

The OVF signal fed back to the input-selection enable, prevents changing the overflow condition once an overflow has been set. The overflow flip-flop must be reset by command bit GCRVZ0N. The K01 input is 0 for Op-codes less than 8 to select shift instructions, or 1 for Op-codes of 8 or higher to select Multiply, Divide, or Double instructions. The shift instructions set overflow (via 74151A inputs 0, 3) on the test A01N ⊕ A00N to indicate a changing sign bit. The Divide instructions, specified by command bit GCRDSRN, set overflow (via 74151A inputs 4,7) on the test ALU00N ⊕ A00, indicating that the quotient is greater than one word.

2.89 GF -- General Flip-Flops

These seven general control flip-flops (Figure 2-8 PP), which can be program controlled, are part of the program status word (PSW). All GF bits are stored in the stack along with the rest of the PSW, but only the FU (User Mode) is restored from the stack by the Return instruction.

GF	PSW Bit	Function	Loaded at CLG (set or reset)	Set by	Reset by
RUNF	8	Main CPU status pointer	L08	STARTF.T1	GFRZ0 +(PREQ. RUN.T1)+(FTDEL. $\overline{\text{RUN}}$)+RSLF
ENBF	9	Enable interrupts	L09	GFENBN.$\overline{\text{MCLN}}$	GFSYSN
CPF	10	Operator's Interrupt	L10	(CPINT+V24INT). .T1	MCLN
PFF	11	Power Failure Interrupt	$\overline{\text{M11}}$ resets	PWF ⨍	PWF.RSLF. UNLOCK
RTCF	12	Real Time Clock Interrupt	L12	RTCFZIN from C.Panel	MCLN
PIF	13	Op-Code (Program) Interrupt	L13	GSTOV.STOV from ALU	MCLN
FU	15	User Mode (0=System Mode)	CLPLR.C15	L15.CLG	GFSYSN+MCLN

Program control of the GF bits is by the A-bus command signal AEPSWN (at clock time T8). AEPSWN selects the program status word (with GF) to the A-bus and gates the L-register data back into the GF. The AEPSWN.T8 control signal CLG gates L08-10, 12, 13, 15 to the respective GF bits. GF bit 11 (PFF) is reset at that time if M11 = 0. Bit 11 uses the M-register rather than the L-register because of the required polarity. In addition to the program control of the GF bits, asynchronous set and reset inputs are provided to each flip-flop.

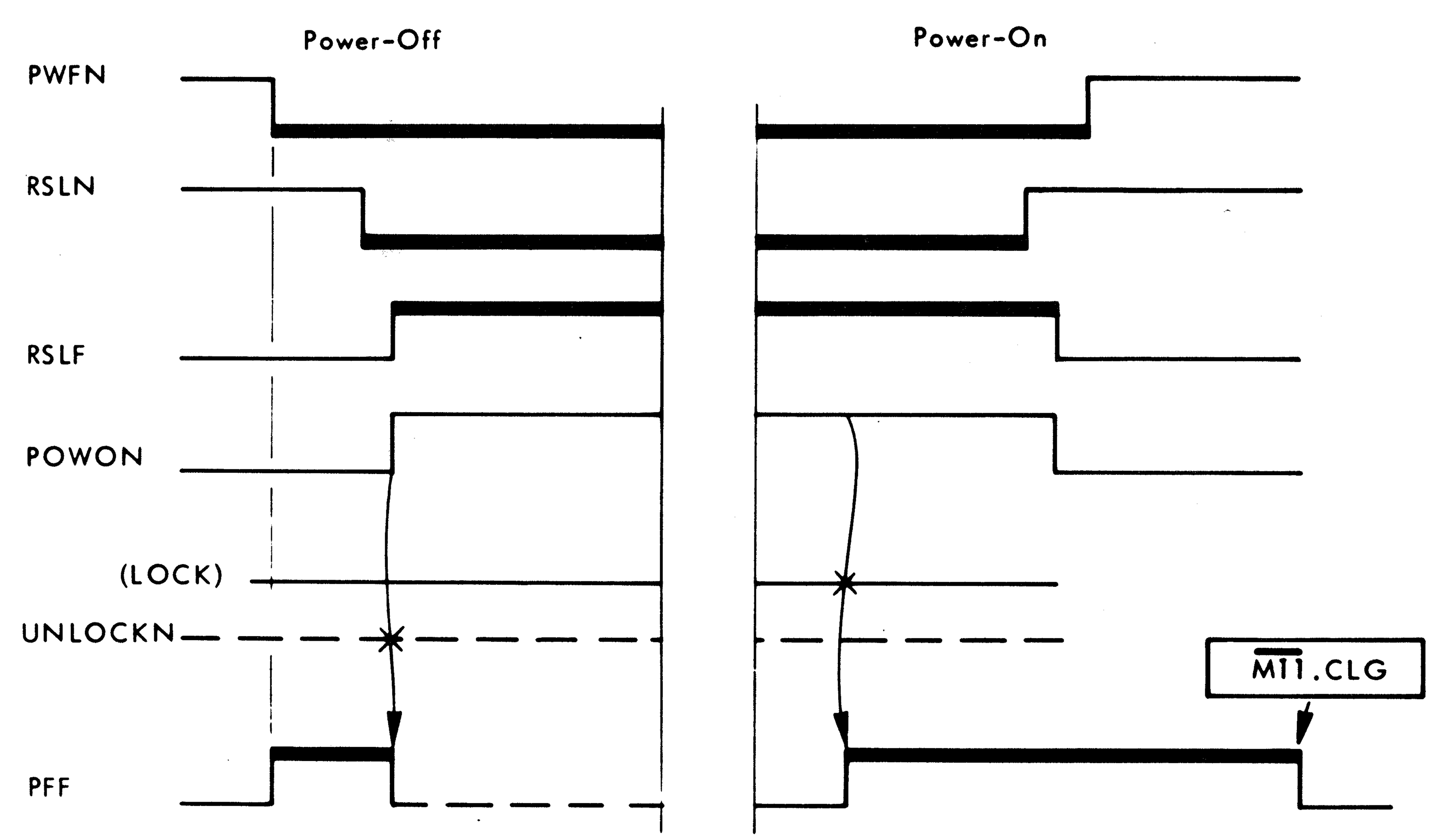
2.90 <u>RUNF.</u> This is the primary machine-state pointer (refer to Figure 1-8). The RUNF flip-flop is set at time T1 by the Start Command from the control panel. The RUNF flip-flop can be reset by any one of four ways :

- command bit GFRZ0;
- a Preset Address Compare from the control panel, at time T1;
- control-panel command Not Run, enabled by the buffered FETCH signal, FTDEL;
- RSLF which is set during the power-off sequence.

2.91 ENBF. This Enable-Interrupt flip-flop is set (with GFENBN) by the Enable Interrupt (ENB) instruction and reset (with GFSYSN) by the Inhibit Interrupt (INH) instruction. ENBF is also set or reset (with GFENBN) by the Return instruction when R2 = 15, and in function with ENB of the restored PSW.

2.92 CPF. This Operator's Interrupt can be set either by the INT switch on the control panel or by the interrupt signal BRGFN generated by the V24/Serial control unit. CPF is reset either by an RIT instruction with bit 0 = 0 (I10 = 0 during the CLG clock) or by the MCLN signal.

2.93 PFF. The Power Failure Interrupt flip-flop is set at the start of a power failure by the leading edge of the PWFN signal from the power supply; PFF is then reset during the power-off sequence if the key switch is set to any unlock position (OFF, ON, ONRTC). The Power Failure Interrupt flip-flop is set during the power-on sequence if the key switch is set to LOCK. PFF is reset by an RIT instruction with bit 11 = 0 (M11 = 0 during the CLG clock).



2.94 RTCF. The Real Time Clock Interrupt flip-flop is set by the RTCFZ1N signal from the control panel. RTCFZ1N is a one-microsecond signal generated once every 20ms by the power supply, and set via the control-panel key switch ONRTC or LOCK positions. The RTCF flip-flop is reset by an RIT instruction with bit 12 = 0 (L12 = 0 during the CLG clock) or by the MCLN signal.

2.95 PIF. The Operation-Code (Program) Interrupt flip-flop is set by a stack overflow condition or by the Link to Monitor (LKM) instruction to switch from User Mode to System Mode. Stack overflow (STOV from the arithmetic unit) sets PIF during memory stack operations (command bit GFSTOV) when the stack pointer decrements to less than 128_{10}. The LKM instruction sets PIF via the load input, with L13 = 1. The PIF flip-flop is reset by an RIT instruction with bit 13 = 0 (L13 = 0 during the CLG clock) or by the MCLN signal.

2.96 FU. The User Mode flip-flop is set during the Return A15 instruction if the previously-interrupted program was in User Mode (CLPLR.C15); the CLPLR signal loads the priority level register from the C-selector during the Return instruction, and C15 indicates the state of FU during the previous interrupt. The User Mode flip-flop is also set by the Set Mode (SMD) instruction (L15 = 1 during the CLG clock). FU is reset, to System Mode, by the Inhibit Interrupt (INH) instruction (with GFSYSN) or by the MCLN signal.

2.97 INTERRUPT LOGIC

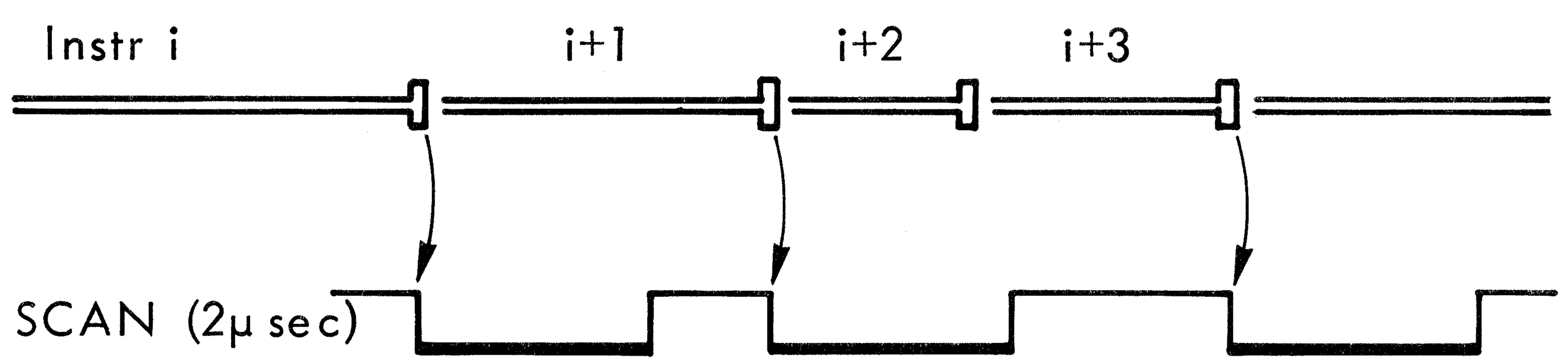
The Interrupt logic (Figure 2-4 and 2-8 SS) generates the Scan External Interrupt signal (SCEIN), tests for internal and external interrupt requests, and initiates the Interrupt routine when a request is received of a higher priority than the running program. Refer to paragraph 1.16 for a description of the complete interrupt system. The Enable Interrupt control flip-flop, ENBF, is located with the general flip-flops (GF), along with the four dedicated internal interrupts : CPINTF, PFF, RTCF, and PIF.

2.98 Internal Interrupts

Eight internal interrupt request lines (IS00-07N) are provided which have the highest system priority. At clock-time AP, all internal interrupt requests are clocked into the two 74175 registers and are then available at the priority encoder 74148. If there are any internal interrupt requests waiting at the priority encoder, the low GS output switches the 74158 multiplexer to select the internal interrupts, and the high EO output (IECE) forces INTAD0,1 low (inactive). The highest-priority interrupt request is indicated by a 3-bit code at the 74148 A-outputs. The interrupt request code is switched through the 74158 multiplexer to the INTAD0-5 lines. The IECGSN signal is sent to the Start logic (Figure 2-8 RR) to enable setting the automatic restart flip-flop during the power-on/power-off sequences.

2.99 External Interrupts

A Scan External Interrupt (SCEIN) signal is generated during the status test at the end of each instruction, or during the Move instruction, if more than two microseconds have passed since the start of the previous scan. In order to validate the interrupt at the end of certain instructions, the microprogram does a 2.070 μsec wait before ending those instructions. (See Figure 1-13, Wait.)



The microinstruction control bits (Figure 2-8 SS) are NAETAN (μSNA0.1) for status test, and GMOVEN for the Move instruction. Either of these inputs puts a high on the D input of the FETAT flip-flop, if there is no pending Interrupt Request (Not IR). At time T3, FETAT is set and the FETATN output triggers the SCEIT single-shot. SCEIT is inverted to provide the 2μsec SCEIN signal to the GP bus.

2.100 The BIECINH flip-flop is triggered by T5 90ns after SCEIT is started. The low BIECINHN output is fed back to the OR'ed input of the single-shot to prevent SCEIT from being re-triggered. On the first T5 pulse after

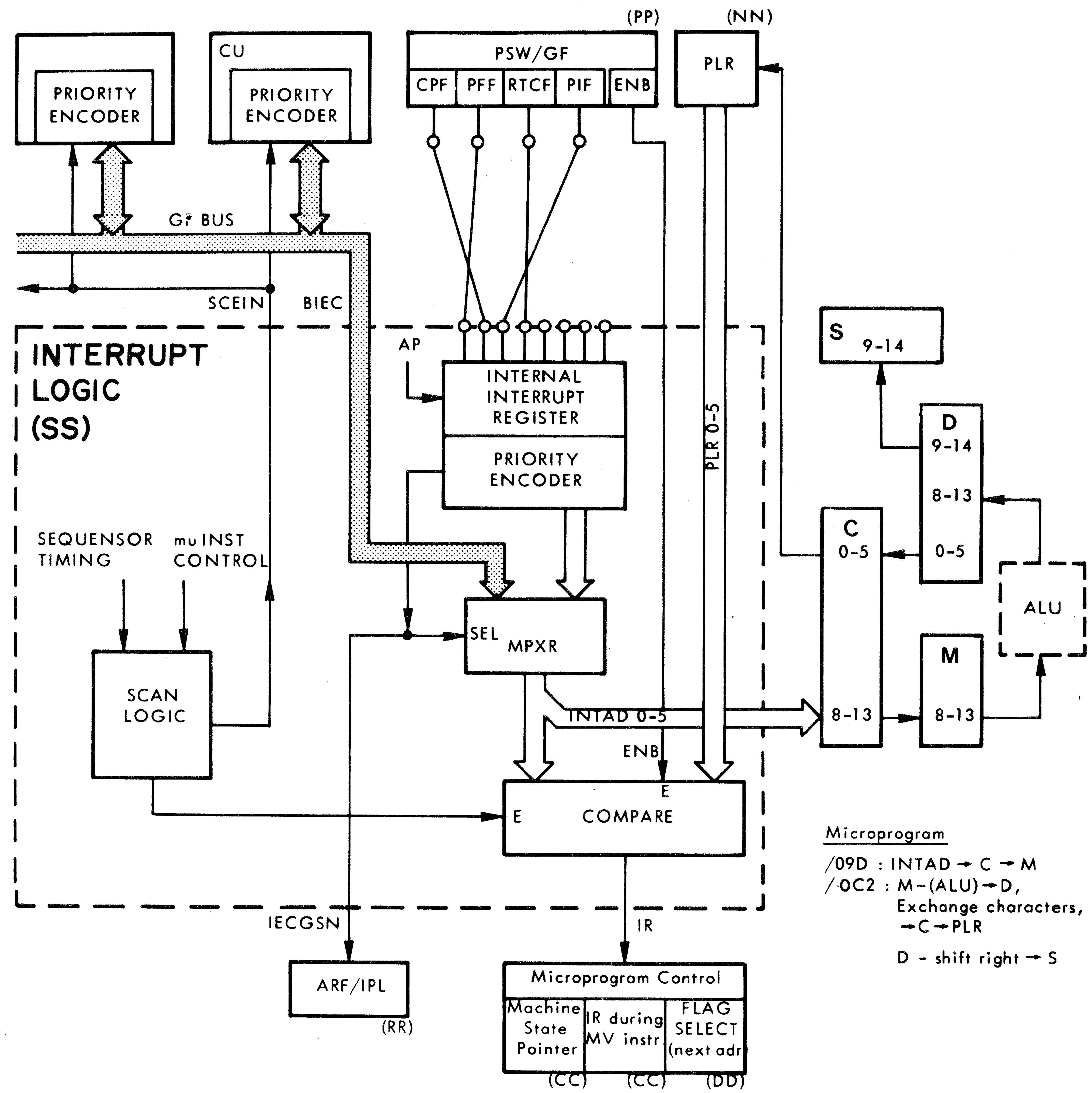


Figure 2-5 Interrupt System

the completion of the 2μsec SCEIT signal, BIECINH is reset, and the SCEIT single-shot is prepared for the next microinstruction-controlled start of a scan signal. The high BIECINH output applied to the most-significant input of the comparator inhibits comparison for 2μsec to wait for stabilization of the BIEC lines (propagation time plus selection of the highest priority level). IR could be sent between T3 and T5, but is never used during status test.

2.101 At the receipt of SCEIN, any CU with a pending interrupt request examines the BIEC lines. If there is no higher priority request, the CU's priority encoder puts the interrupt code on the BIEC0-5 lines. The BIEC code received at the Interrupt logic is switched through the 74158 multiplexer to the INTAD0-5 lines if there is no internal interrupt request.

2.102 Compare Interrupts

The highest-priority interrupt request (internal or external) is available at the INTAD0-5 lines. If the Enable Interrupt flip-flop ENBF is set and the BIECINH delay is completed, the INTAD lines are compared with the priority of the running program (stored in the priority level register PLR). BIECINH and ENBFN to the two most-significant inputs of the comparator must both be low to enable comparison. (The comparator inputs A3/B3 must compare equal before the A2/B2 inputs are tested, etc.; all four A and B inputs must compare before the cascade inputs from the lower-priority comparator are tested.) If the interrupt request on the INTAD lines is of higher priority (lower number) than the current PLR, the A<B output is activated to generate Interrupt Request signal IR.

2.103 The IR signal is sent to the Microprogram Control logic to initiate the Interrupt routine (Figure 2-5). The interrupt address on the INTAD0-5 lines is switched through the C and D selectors during the Interrupt routine to form part of the address in the S register and to reload the PLR with the new priority level.

2.104 SEQUENSOR (CPU CLOCK)

The Sequensor (Figure 2-8 EE) uses a crystal-controlled oscillator to drive twelve shift-register cells which provide the CPU timing signals :

- AP, T1, T2, T3, T4, T5, T6,
- BP, T7, T8, T9, T10.

The basic CPU clock signals are AP and BP. AP represents the beginning of a functional cycle (or microprogram) and is used to load the microinstruction address and read the microinstruction from the ROM control store; CPU logic controls and data paths are established by the command bits of the microinstruction. At each BP pulse time, the CPU logic executes the microprogram, according to the conditions established at AP time, and loads the results into the appropriate CPU registers.

2.105 Operating Modes and Cycles

The Sequensor operated in four different modes (Figure 2-6) controlled by microinstruction bits μSEQ0,1. Within the basic modes, the Sequensor operates in one of six cycles. Five of the operating cycles begin with the AP pulse; the sixth (Repeat cycle) begins and ends with the BP pulse, skipping AP. For internal CPU operations (LOGIC and REPEAT modes), the cycle lengths are controlled by controlling the generation of AP to begin the next cycle (pulses T7-T10 are inserted or deleted after BP as required). For all GP-bus cycles (SEQBUS and SEQBIO modes) and Floating Point execution, the cycle lengths are controlled by controlling the generation of T6 just before BP (pulse T5 may be repeated as a waiting loop until a synchronizing reply signal triggers T6).

2.106 Internal CPU Operations (AP Generation)

Four fixed-length Sequensor cycles are provided for the internal operations LOGIC mode :

- Logic cycle (360ns)
- Scratchpad cycle (405ns)
- Flag cycle (450ns)
- Fetch cycle (540ns)

mu SEQ 0,1	Mnemonic	Function
0 0	LOGIC	Internal CPU Operations
0 1	REPEAT	Short Execution Cycle
1 0	SEQBUS	GP-Bus Cycles
1 1	SEQBIO	I/O Cycles with Bus not released

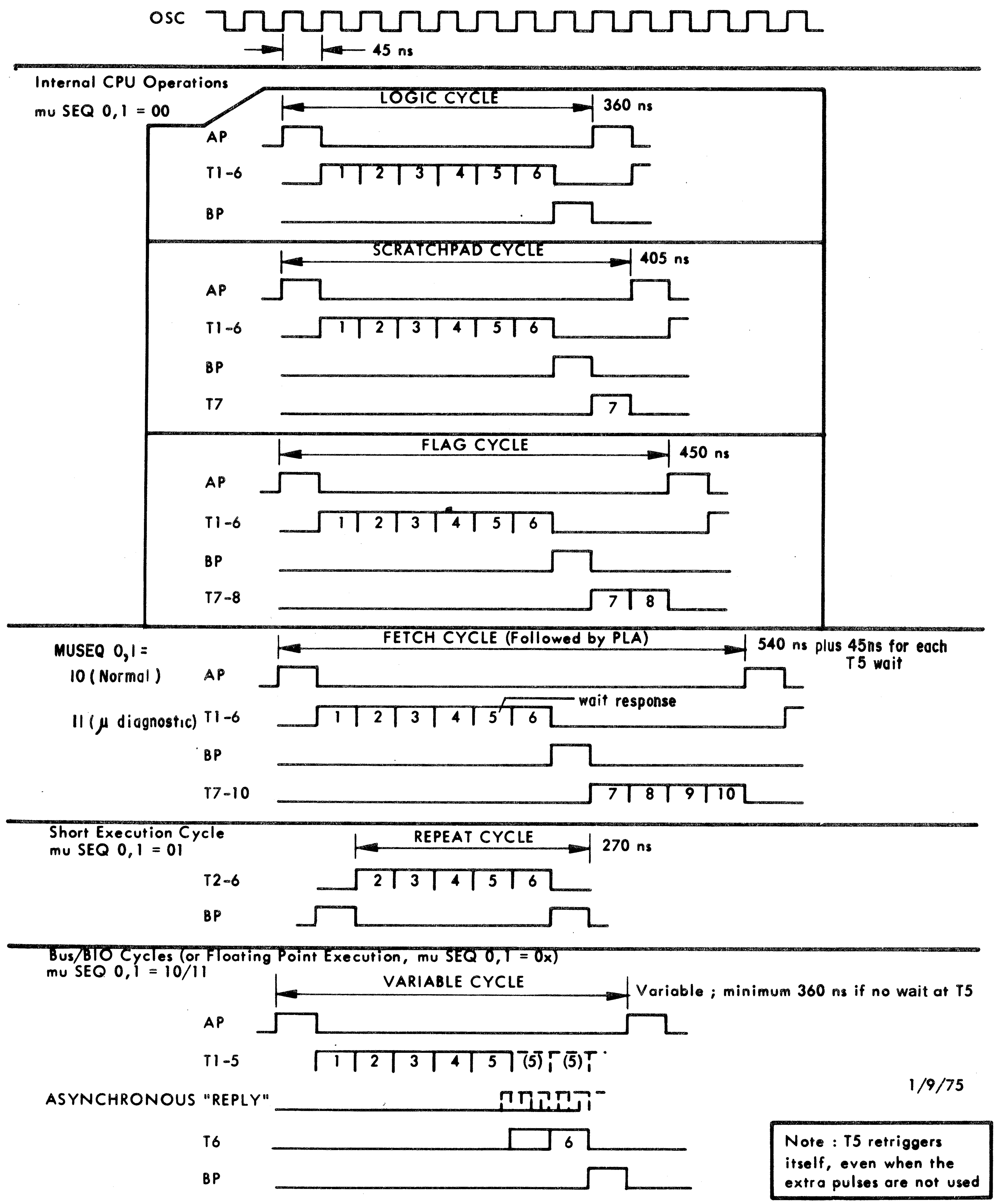


Figure 2-6 Sequensor Operating Cycles

All these cycles begin with AP and count directly through T1, T2, T3, T4, T5, T6, and BP. The time between BP and the end of the next cycle (next AP) is then controlled to provide the required execution time for the operation and to define all Next-Address data. The control is provided by a type 74151A multiplexer (APDN) which selects which clock pulse is to generate AP, as follows :

GFETCH	NAFLAGN	WSPRSLI	APDN Generation	
S2	S1	S0	Input	Function
0	0	0	T8	Flag Cycle (NAFLAG)
0	0	1	T8	
0	1	0	BP	Logic Cycle $(\overline{NAFLAG.WSP})$
0	1	1	T7	Scratchpad Cycle $(\overline{NAFLAG}.WSP)$
1	0	0	T10	Fetch Cycle (GFETCH)
1	0	1	T10	
1	1	0	T10	
1	1	1	T10	

2.107 Logic Cycle. This cycle is used for register-to-register operations, with the result stored into a register other than the scratchpad. Input I3 of the APDN multiplexer is selected so that the AP pulse to begin the next cycle follows the BP pulse.

2.108 Scratchpad Cycle. This cycle is used for register-to-register operations with the result stored in the scratchpad. Data transfer from the L register to the scratchpad requires the extra clock pulse T7 between BP and the end of the cycle.

2.109 Flag Cycle. This cycle is used with a microinstruction that specifies Next-Address Flag mode. Two extra clock pulses (T7 and T8) are required to gate the Flag information onto the control-store address bus.

2.110 Fetch Cycle. The Sequensor fetch cycle is used during an instruction fetch cycle, where the Next Address must be decoded by the instruction-register PLA decoder. The timing required to decode the instruction word and to access the PLA needs four extra clock cycle (T7, T8, T9, T10) following BP.

2.111 Short Execution Cycle (No AP)
The Repeat cycle counts directly through T2, T3, T4, T5, T6, and BP without an AP or T1 clock pulse. This cycle is used for looping while the scratchpad CT counter counts. The microinstruction control inputs, with $\overline{\text{CTEND}}$, hold REPENDI high to gate the BP pulse directly to the T2 input. REPENDI also disables the APDN multiplexer to inhibit the AP pulse. The current microinstruction does not change during the Repeat cycles since AP is inhibited. When the CT counter reaches its full count of 31_{10}, CTEND forces REPENDI low and the APDN multiplexer is enabled to generate AP to begin the next cycle.

2.112 GP-Bus Cycles (T6 Generation)
Variable-length cycles are used for all GP-Bus operations and for Floating Point instruction executions. All these cycles begin with AP and count directly through T1, T2, T3, T4, and T5. The cycle-length parameters then control the generation of the T6 clock. The BP pulse and the succeeding AP pulse directly follow T6. The control of T6 is provided by a type 74151A multiplexer which selects the T6-enabling signal T6DN, as follows :

μSEQ0	FLOACT	BSYDL	T6DN Generation	
S2	S1	S0	Input	Function
0	0	0	1	Internal CPU operations (T6 follows T5)
0	0	1	1	
0	1	0	DONEF	Floating Point instruction execution
0	1	1	0	not used
1	0	0	0	
1	0	1	SYNCT6	Bus and Bus I/O cycles (SEQBUS, SEQBIO)
1	1	0	0	not used
1	1	1	0	

2.113 For the internal CPU operations, T6 directly follows T5. Sequensor control is provided through control of the AP pulse (preceeding paragraphs). The enabling One-bit is selected through the multiplexer by μSEQ0=0 and the floating point command bit FLOACT=0.

2.114 The Floating Point instruction execution is an operation that requires the DONEF response signal from the FPP before the cycle can be terminated. DONEF is selected directly at the I2 input of the T6DN multiplexer by μSEQ0=0, floating-point command bit FLOACT=1, and BSYDL=0. The FLOACT signal is the buffered microinstruction general-field bit GFLOT.

μSEQ	0,1	
	1 0	SEQBUS
	1 1	SEQBIO

2.115 For all GP Bus operations (SEQBUS and SEQBIO), the SYNCT6 input to the multiplexer is selected by μSEQ0=1, BSYDL=1, and FLOACT=0. The μSEQ0 bit = 1 for all Bus operations. BSYDL = 1 95ns after the CPU takes control of the GP Bus. The Sequensor does a waiting loop on the T5 clock until the required reply is received via the 8-input Nand gate SYNCT6. For Bus transfers that don't require memory exchange, the reply signal is provided by the Microprogram Control and T6 may directly follow T5. The reply signal (listed in Table 2-12) must be one of the following:

a. TRMN (SYNMEMN) must be received during a CPU/Memory exchange or during a CPU/External-Register exchange. The TRMN response signal (from Memory or External Register) on the Bus is received and gated by the Bus Controller logic (Figure 2-8 TT) to the Sequensor as input signal SYNMEMN.
b. TPMN delayed 190ns (TPMDLN) is received from the CU during a CU/Memory exchange under CPU control (programmed-channel transfer). The 190ns delay of TPMN is produced by Bus Controller logic for assuring data timing.
c. TIMEOUT generates T6 to end the cycle if no response is received to one of the Master-to-Slave signals TMRN, TMPN, TMEN within

6.48 μsec. This may indicate that the addressed Memory, Programmed Channel, or External Register is not present.

d. BIOEKEY gates T6 without extra T5 waiting loops during a control-panel to CPU transfer. BIOEKEY is generated on the Bus Controller logic (Figure 2- 8 TT) by the microinstruction general-field bits GCRFNUN and GBOKFN, and gated into the Sequensor by GIDLEN.

e. GFETCHN general-field bit gates T6 without extra T5 waiting loops during instruction (K) register loading from the L register, via the Bus BIO lines. GFETCHN is gated by microinstruction control bit μTMRN to generate SYNIMN to the T6 input gate.

f. DONEMN is received from the MMU during an MMU/Memory exchange which is always under CPU control.

g. DONEFN is received from the FPP during an FPP/Memory exchange which is always under CPU control.

h. MFAULTN, delayed 120ns, from the MMU indicates a page fault (PAFN) during a Move Table instruction. The MFAULTN signal sets the PAF flip-flop in the control-store addressing logic (Figure 2- 8 CC). The PAFN signal is delayed 120ns at the Sequensor input (SYNCT6) for control-store address timing.

i. BOFFN from the general-field bits GCRFNUN and GBOFN (Bus Controller logic) gates T6 without extra T5 waiting loops when FPP data is loaded, via the Bus BIO lines, into the CPU during a Fix instruction.

2.116 POWER FAILURE, RESTART, RESETS

The power-on and power-off sequences are shown on Figure 2-7. The logic is shown on Figure 2-8 RR, Start/Resets. While power is off, the Reset Logic (RSLN) signal is held at ground (active low) through a relay on the power supply.

2.117 Power-On Sequence

When the power supply is switched on, or when the mains power is restored after a failure, there is approximately an 850ms delay while the power supply stabilizes (Section 5, Power Sequence Logic) before RSLN goes inactive (high).

Table 2-12 Sequensor T6 Generation

μSEQ0	FLOACT	BSYDL			
S2	S1	S0	Cycle	(text)	Synchronize T6 on :
0	0	x	Internal CPU operations		T5
0	1	0	Internal, Floating Point execution		DONEFN from FPP
1	0	1	GP Bus transfers with Memory exchange		
			CPU/Memory	(a)	TRMN (SYNMEMN) from memory
			CPU/External Register	(a)	TRMN (SYNMEMN) from memory
			Programmed Channel CU/Memory	(b)	TPMN from CU, delayed 190ns
			Addressed Device Missing	(c)	TIMEOUT, generated 6.48μsec after TMRN/TMPN/TMEN sent.
			MMU/Memory	(f)	DONEMN from MMU *
			FPP/Memory	(g)	DONEFN from FPP *
			Page Fault	(h)	MFAULTN from MMU (PAFN), delayed 120ns. *
			GP Bus transfers without Memory exchange		
			Control Panel	(d)	T5, if GBOKFN/GCRFNUN (BIOEKEY), and GIDLEN
			Fetch execution	(e)	T5, if GFETCHN and μTMRN.
			Load from FPP (Fix execution)	(i)	T5, if GBOFN (BOFFN) and GCRFNUN *
			any	(a-i)	TIMEOUT, if required reply not received within 6.48 μsec

* Note - Signals (f,g,h,i) used with 857 only.

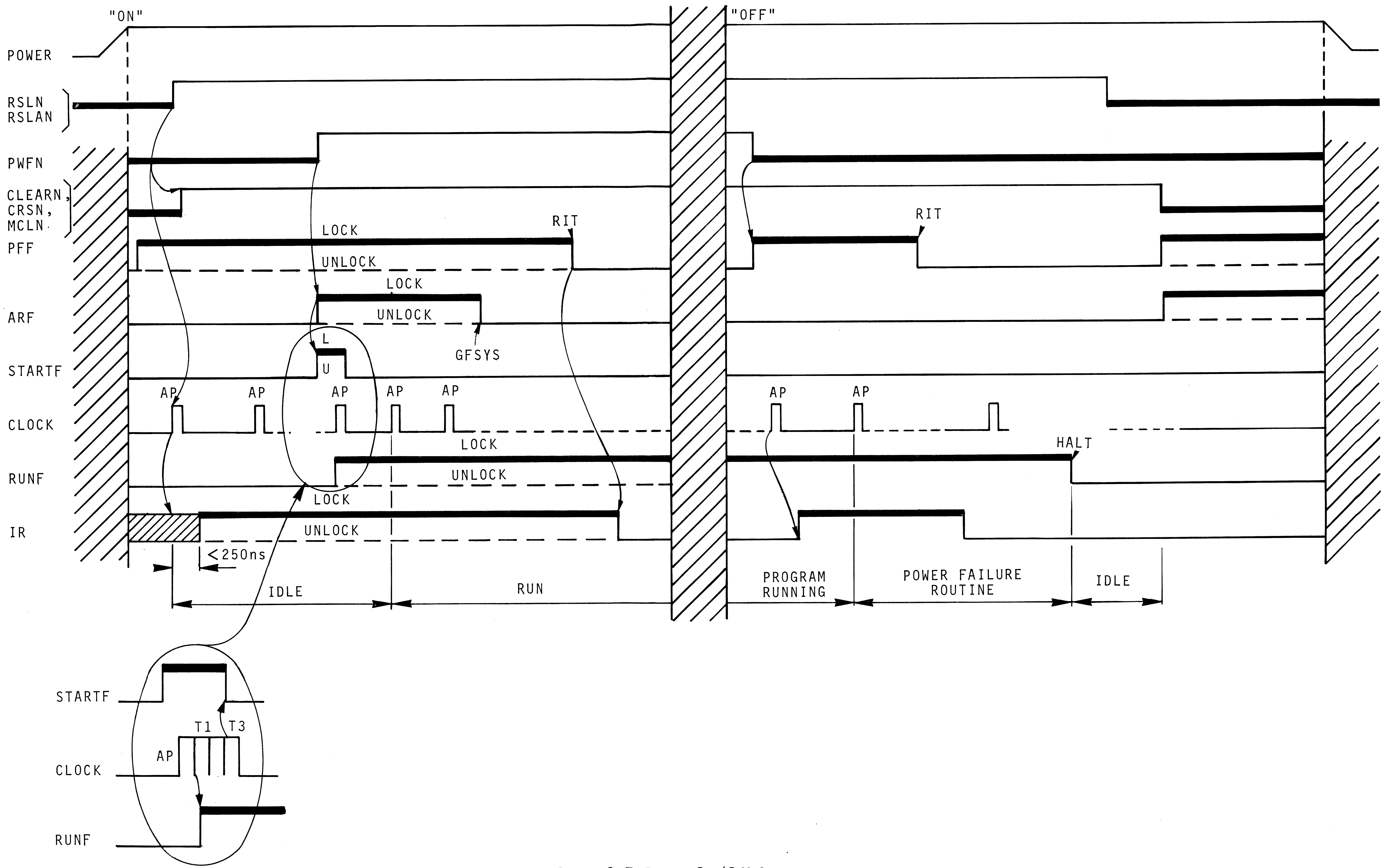


Figure 2-7 Power On/Off Sequence

When RSLN goes high, the OSCN pulse immediately resets RSLF to remove the logic reset condition. The inactive RSLAN/BN/CN signals to the Sequensor enable clock-pulses T1, T3, T5, T6, AP, BP, and the clock begins running.

2.118 Automatic Restart
Automatic restart is performed by the CPU if the control-panel key is at LOCK when the power comes on, therefore the key must be switched to LOCK when switching on or following a power failure while the key is not at LOCK. If the key is at LOCK when the +5 volts comes on to operate the flip-flops (and before RSLN, RSLF drop), PFF is set at the beginning of the power-on sequence (paragraph 2.93, PSW/GF, PFF). Beginning with the first AP, the internal interrupt request PFF is sampled by the Interrupt Logic. A level-0 (highest priority) Interrupt Request (IR) is then generated and the any-internal-interrupt signal IECGSN is active to the ARF flip-flop. The Power Failure (PWFN) signal from the power supply goes inactive (high) 0.5ms after RSLN goes inactive.

2.119 With the key at LOCK, the IECGSN signal sets Automatic Restart flip-flop ARF at the edge of the high-going PWFN signal. The ARFN output sets the STARTF flip-flop directly, via the asynchronous S-input. The RUNF flip-flop (2-8 PP) is set on the first T1 clock after STARTF is set. At the first AP after RUNF sets, the first microinstruction of the Automatic Restart routine (Figure 1-13) is read, and the CPU switches from Idle state to Run state. The second microinstruction of the Automatic Restart routine resets ARF with the GFENBN signal. The PFF flip-flop should be reset during the automatic restart subroutine by an RIT instruction.

2.120 Power-Off Sequence
The Power Failure (PWFN) signal is set by the power supply 10ms after detection of a mains loss, caused by power failure or by the key switched OFF. PWFN is sent via the GP Bus to the CPU and to any other elements on the Bus that must take action during power failure. The leading edge of PWF sets the PFF (GF/interrupt) flip-flop. This level-0 internal interrupt is detected by the Interrupt Logic at the first AP clock following the start of PWFN. The resulting Interrupt Request (IR) can initiate an interrupt subroutine for a power failure.

2.121 PWFN to the Bus Controller logic inhibits setting the OKVAL flip-flop. The Bus Controller thus denies requests by any other Master for Bus access, and the CPU has exclusive control of the Bus. The power failure subroutine requested by IR and PFF should commence with the second AP clock following PWFN. The subroutine should then reset PFF with an RIT instruction and reset RUNF with an HLT instruction. This saving subroutine must not last more than 2ms, and must end with a HLT instruction. The HLT instruction selects Microprogram Control address /177 to place the CPU in the Idle state. The Reset signal (RSLN) from the power supply is activated 3ms after PWFN.

2.122 Resets
The CPU logic is reset at the end of the power-off sequence by the RSLN signal from the power supply. RSLN is held active low while power is off. The first OSCN pulse after RSLN is active sets the RSLF flip-flop. RSLF activates RSLAN, RSLBN, RSLCN, and WSPRSLI, and resets the flip-flops STARTF, WSP, and RUNF. The RSLF-derived signals set or reset the CPU registers and flip-flops to their off states (Table 2-13) and hold them in that condition until the power-on sequence. The RSLN signal also activates the Master Clear signals CLEARN (to the Bus), MCL, and MCLN. Sequensor clock pulses are disabled by RSLF so that the CPU clock is stopped, although the oscillator-generated signal OSC runs continuously.

2.123 Master Clear
The Master Clear signal CPMCN can be generated by the control-panel MC switch if the key is <u>not</u> at LOCK and if the CPU is <u>not</u> in Run mode. The CPMCN signal (Figure 2-8RR) is OR'ed with the RSLF output to activate the signals CLEARN, MCL, and MCLN. CLEARN is sent on the GP Bus as a reset/initialize signal to all System elements. MCL is sent to the V24 CU on the CPU card. MCLN is used by the CPU (Table 2-13) to set the priority level register (PLR) to

Table 2-13 Resets and Clears

Logic-Resets, by RSLN/RSLF only

Logic	Sig.	Action
(RR)	RSLF	RSLAN,BN,CN → 1 STARTF → 0 WSPRSLI → 0
	WSPRSLI	WSP → 0
	RSLAN	ARFN → 0 IPLF → 0
(AA)	RSLBN	K-reg → 0
	RSLAN	KxxBUFN → 0
(BB)	RSLBN	RA0-8 → 0
(CC)	RSLAN	FTDEL → 0 KRY → 0
	RSLBN	PUPN → 0
(DD)	RSLBN	FSIG → 0 FSIGDIV → 0
(EE)	RSLBN	T1,T3,T5 → 0 TC810 → 0
	RSLAN	AP → 0 BP → 0 T6 → 0
	RSLCN	BPQ → 0

Logic	Sig.	Action
(JJ)	RSLCN	Q-reg → 0
(LL)	RSLBN	DWIF → 1
(NN)	RSLBN	OVF → 0 CR0,1 → 0
(PP)	RSLF	RUNF → 0
(SS)	RSLAN	BIECINH → 0 IF00-07 → 0
(TT)	RSLAN	BUSF → 0 DE → 0 FNU → 0 (DD) TMEF → 1 TMPF → 1 WRITFN → 0
	RSLBN	BIOELN → 1 BSYCPU → 0 CHFN → 1 FLOACT → 0 GBCPFN → 1 MADCPUN → 1 MADLCPUN → 1 MUBUSRFN → 1 OKVAL → 0 TMFN → 1 TMMN → 1 TMMU → 0 TMRF → 0

(PP) RSLF.PWF.LOCK set interrupt flip-flop PFF

Resets by Master Clear or RSLN/RSLF

Logic	Sig.	Action
(RR)	CPMCN or RSLAN	MCL → 1
	MCL	CLEARN → 1 MCLN → 1 V24 resets
(NN)	MCLN	PLR → level 63
(PP)	MCLN	ENBF → 1 CPINTF → 0 RTCF → 0 PIF → 0 FU → 0 CPGFZ0N → 1
	CPGFZ0N	CPGF → 0 V24 resets

level 63 and set the Enable Interrupt flip-flop ENBF. The GF flip-flops CPINTF, RTCF, PIF, and RU are all reset, and the control panel interrupt flip-flop CPGF is reset.

Table 2-14 CPU Signal LIST (A-C)

Signal	Logic	Comment
0ALUL,R,N	GG	
0ALURA	NN	
A00N-15N	LL,MM	A-Bus
A00AN	NN	
ABN	AA	
AC,N	NN	BUS(in)
ACAN	V24	
ACBN	NN	
ACHALTN	V24	
ACINR,N	V24	
ACOTRN	V24	
ACSOT,N	V24	
ACSST,N	V24	
ACSTARTN	V24	
AECONSTN	LL	
AEIPLN	LL	
AENBN	LL	
AEPN	LL	
AEPSWN	LL	
AESYSN	LL	
ALU00	FF	
ALU00N-15N	GG	
ALUCO04,08,12,16	GG	
AEPUPN	LL	
ALUCG00N,04N,08N,12N	GG	
ALUCP00N,04N,08N,12N	GG	
ALUS0-3	GG	
ALUS0A-3A	GG	
ALUS0N-3N	FF	
AP,N,DN	EE	
AREDELA,N	V24	
AREDELBN	V24	
ARERCA,CB	V24	
ARF,N,D	RR	
ARFZ0N	RR	
ASR0-7	V24	
BOFFN	TT	Cmd → FPP
BOKFN	TT	
BOKIDLN	EE	
BOMFN	TT	
BIO00N-15N	KK	BUS(in/out)

Signal	Logic	Comment
BIO00AN-15AN	KK	
BIO00R-15R	KK	
BIOEKEY	TT	
BIOELN	TT	
BIOL	TT	
BIEC0-5	SS	input(ext.Int)
BIEC0R-5R	SS	
BIECINH,N	SS	
BP,N	EE	
BPLN	HH	
BPP	EE	
BPQ,N	EE	
BPS	KK	
BRC	AA	
BRGF,N	V24	
BSYCPU,N,AN	TT	
BSYDL	TT	
BSYN,R	TT	BUS(in/out)
BSYZ0N-1N	TT	
BUSF,N,AN	TT	
BUSFDEL,N	TT	
BUSFREN	TT	
BUSR,N	TT	BUS (in)
BUSRDEL	TT	
BUSRPFIN	TT	
C00N-15N	JJ	
CHA	TT	BUS
CHFN	TT	
CHIPER,D	V24	
CHPERZ0N	V24	
CLEARN	RR	BUS
CLEFTZ0	JJ	
CLF0N	V24	
CLG	PP	
CLK	AA	
CLMN	GG	
CLPLR	BB	CMD/G
CLTE0C	V24	
CLV24	V24	
COMVAL	V24	
CPBABS	LL	in from C.Panel
CPBRG	PP	
CPGFN	PP	
CPGFZ0N	PP	

Table 2-14 CPU Signal LIST (C-G)

Signal	Logic	Comment
CPINT,A	PP	in
CPINTF,N	PP	
CPINT4N	PP	
CPINZ1N	PP	
CPLR	LL	
CPMCN	RR	in from C.Panel
CPRR	LL	
CR0-1	NN	
CR0DN	NN	
CR1DN	NN	
CR0XK6	AA	
CR1XK7	AA	
CT0-4	MM	
CT101,102	V24	grounds
CT106,7,9	V24	
CT1082	V24	in for DNOTOP
CT133	V24	in for DREADYN
CT103,N	V24	in
CT104,N	V24	
CT0DN	MM	
CT0Z1N	MM	
CT1CO	MM	
CTBUF1N-4N	MM	
CTEND	MM	
CVN	AA	
CVKCR	AA	
D00-15	HH	
D00N-15N	HH	
D00DN	FF	
DE,N	TT	
DEBSYN	TT	
DEZ0N	TT	
DEZ1BPN	TT	
DES1T2N	TT	
DIVFLAG	FF	
DNOTOP	V24	CT1082 input
DONEF,N	EE	in from FPP
DONEMN	EE	in from MMU
DREADYN	V24	CT133 input
DS0-1	FF	
DS0N-1N	FF	
DWIF,Z0N	LL	
DWIN	AA	

Signal	Logic	Comment
ECHO,N	V24	
ECHOZ1N	V24	
ENBF,N	PP	
ENBFZ1N	PP	
ETATEST	CC	
F0-1,N	V24	
F0Z1N	V24	
F1Z0N	V24	
FACINR,N	V24	
FE,N	V24	
FECHO	V24	
FETATDN,EN	SS	
FETATN	SS	
FETCH	BB	
FHALT,N	V24	
FHALTZ0N,Z1N	V24	
FLAGDIV	DD	
FLAGN	DD	
FLOACT	TT	
FLOCR0-1	NN	in
FNU,N,D	DD	
FPPABS	AA	in from FPP
FSIG,N	DD	
FSIGDVD	DD	
FTBOF	EE	
FTDEL,N	BB,LL	
FTDERU	PP	
FTE0CN	V24	
FU	PP	User mode
FU15R2N	DD	
FUZ0N-Z1N	PP	
GAEXL,N	BB	CMD/G
GB0FN	BB	CMD/G
GB0KN	BB	CMD/G
GB0MN	BB	CMD/G
GBCHN	BB	CMD/G
GBCPFN	TT	
GBCPN	BB	CMD/G
GBEX,N	BB	CMD/G
GBTMEN	BB	CMD/G
GBTMFN	BB	CMD/G
GBTMMN	BB	CMD/G
GBTMPN	BB	CMD/G
GCRDSR,N	BB,FF	CMD/G

Table 2-14 CPU Signal LIST (G-M)

Signal	Logic	Comment
GCRFNU,N	BB,DD	CMD/G
GCRVMLN	BB	CMD/G
GCRVZ0,N	BB,DD	CMD/G
GCSELN	BB	CMD/G
GCT	MM	
GCTLDN	BB	CMD/G
GFENBN	BB	CMD/G
GFETCH,N	BB	CMD/G
GFKYZ0N	BB	CMD/G
GFLOT,N	BB	CMD/G
GFPLRN	BB	CMD/G to CLPR
GFRZ0,N	BB	CMD/G
GFSTOV,N	BB	CMD/G
GFSYSN	BB	CMD/G
GIDLEN	BB	CMD/G
CMOVEN	BB	CMD/G
CMULTI,N	BB,FF	CMD/G
GOSH	DD	
GOTOECH	V24	
GOTOWST	V24	
IEC0-2,N	SS	
IECE	SS	
IECGSN	SS	
IF00N-07N	SS	
IN	V24	
INRECH	V24	
INSTN	--	not used (C.Pan)
INTAD0-5	SS	Interrupt address
INTEPL	SS	
INTGPL	SS	
INTLPL	SS	
INTSERN	V24	
INTVAL	V24	
IPL,N	RR	in from C. Panel
IPLDEL	RR	
IPLF,N	RR	
IR,N	SS	
IS00N-07N	SS	in (Intern. Int's)

Signal	Logic	Comment
K00-15,N	AA	
K0008	AA	
K05-08BUFN	AA	
K08Q15	FF	
K10R2E0	AA	
K10R2N	AA	
K410N	AA	
K415N	AA	
K567	AA	
KFL0	AA	
KRY,N,DN	CC	
KRYZ0N-Z1N	CC	
L00-15	HH	
LOADMN	CC,LL	in from C. Panel
LOADRN	CC,LL	in from C. Panel
LSEL	FF	
M00N-15N	GG	
MAD00-15, 64,128	KK	
MADCPB	TT	
MADCPUN	TT	
MADLCPUN	TT	
MADLDN	TT	
MADLS	TT	
MADMMU,N,A	TT	
MADMUN	TT	
MADS	TT	
MCL,N	RR	
MFAULTN	CC	in from MMU
MMUABS	TT	in from MMU
MOVE	CC	
MOVIRN	CC	
MSN	TT	BUS (in)
MSR	TT	
μA0-4	BB	CMD
μA0N-4N	LL	
μAIN	RR	
μADL0-4	BB	CMD
μBIOL,N	BB,TT	CMD
μBUSR,N	BB,TT	
μBUSRFN	TT	CMD
μC0-1	BB	CMD

Table 2-14 CPU Signal LIST (M-R)

Signal	Logic	Comment
μCON	JJ	
μCR0-2	BB	CMD
μCT	BB	CMD
μGP0-4	BB	CMD to CMD/G
μGPIN	BB	CMD to CMD/G
μMLOAD	BB	CMD
μMSEL	BB	CMD
μNA0N-8N	BB	CMD
μNA7	AA	
MUNA7FUN	AA	
μP0-1	BB	CMD
μPON	LL	
μQ0-1	BB	CMD
μS0-1	BB	CMD
μSON	KK	
μSEQ0-1	BB	CMD
μSEQIN	BB	CMD
μSNA0-1,N	BB	CMD
μTMRN	BB	CMD
μWRITE	BB	CMD
MVREADN	CC	
MVRML	CC	
NACND	AA	
NACNDAD	AA	
NADRT	CC	
NADRTMV	CC	
NAETAN	BB	CMD μSNA0.1
NAETAT	CC	
NAEXPL	CC	
NAFLAGN	CC	
NAFLG	CC	
NAFLGX	CC	
NAPLINH	CC	
NOFLON	AA	
NOJUMPN	AA	
NOVF	FF	
NXTCH,N	V24	
NXTCH0	V24	
ODEVPAR	V24	
OKO	TT	BUS
OKVAL	TT	
OR,ORB	V24	
OROPE	V24	

Signal	Logic	Comment
OSC,N	EE	
OSC90,N	EE	
OSCFLO	EE	
OTINECH	V24	
OTRECHN	V24	
OUT	V24	In/Out FF
OVDIVA,B	NN	
OVF,N,D	NN	
OVFMU,N	NN	
OVSHA,B	NN	
P00-14	LL	
PAFDELN	EE	
PAFN,Z0N	CC	
PARNB	V24	
PC003,07,11N	LL	
PE,PEB	V24	
PFF,N	PP	
PFFZ0N, Z1N	PP	
PIF,N	PP	
PIFZ1N	PP	
PLOADN	LL	
PLA0-1, 0N	AA	
PLAVAL1	AA	
PLAVALN	AA	
PLR0-5	NN	
PMI	LL	
POVFN	FF	
POWON	PP	
PREQN	PP	in from C.P.
PREQTI	PP	
PUPN,D	CC	
PWF,N	TT	in from P. Supply
PWFAN	PP	
Q00	JJ	
Q00N-16N	JJ	
Q15DN	JJ	
R1E0,N	AA	
R1E15	AA	
R2E0,N	AA	
R2E15	AA	
R110,N	V24	

Table 2-14 CPU Signal LIST (R-T)

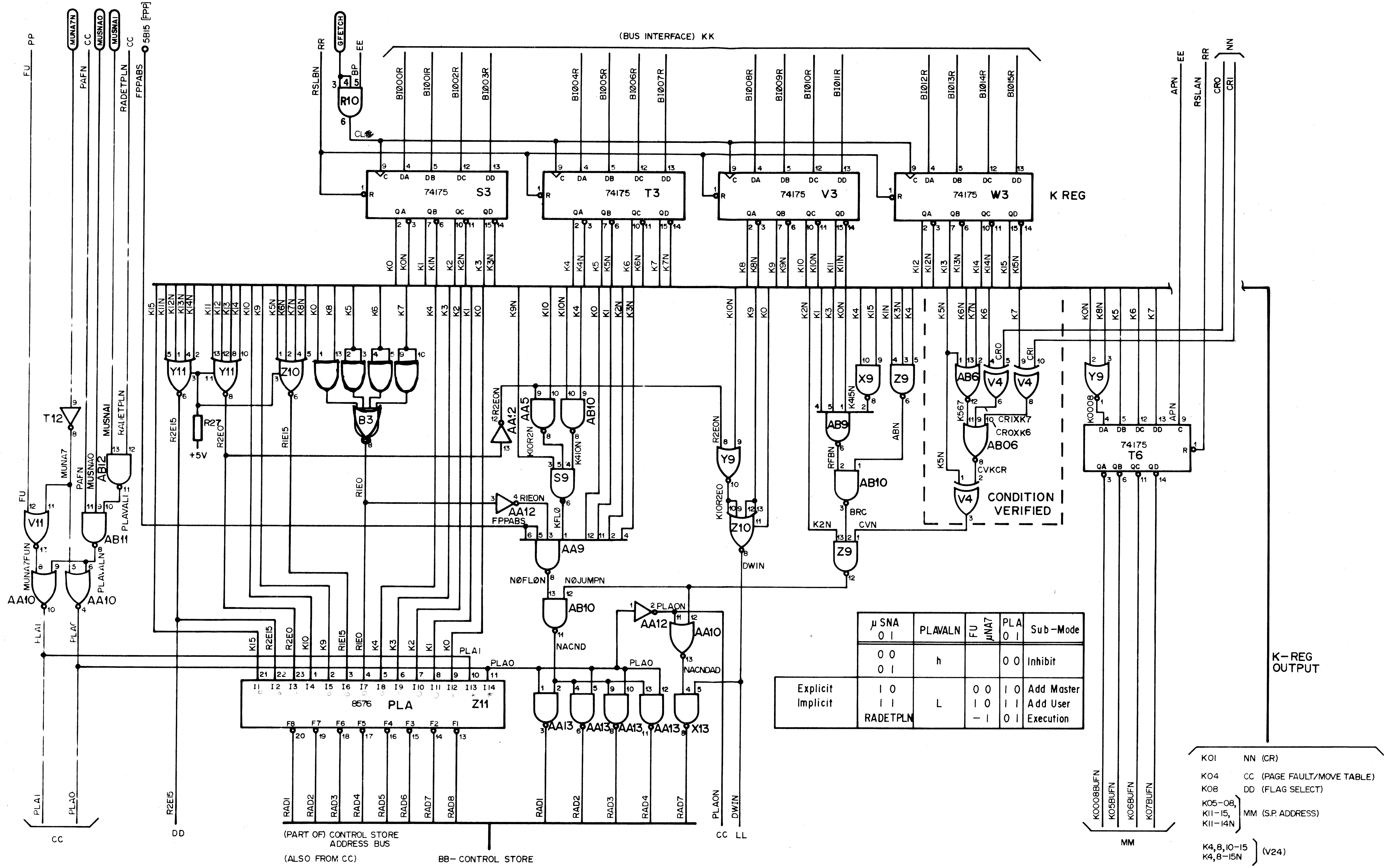
Signal	Logic	Comment
R600,1200	V24	
R2400,4800	V24	
R9600	V24	
RA0-8	BB	
RAD0-8	AA,CC	Control Addr-Bus
RADET6N-T8N	CC	
RADETPLN	CC	
RADL3-4	FF	
RALUA1N-2N	FF	
RATESEL	V24	
RCP0N-3N	LL	in from C.Panel
RD08-15	V24	
RDA,N,1	V24	
RDADEL,N	V24	
RDARC	V24	
RDARN	V24	
READMN	CC,LL	in from C.Panel
READRN	CC,LL	in from C.Panel
READSTN	CC,LL	in from C.Panel
REPENDI	EE	
RFBN	AA	
ROMCR1-8	NN	
ROMCREN	NN	
ROMENB	BB	
RSLAN,BN,CN	RR	
RSL,N	RR	BUS (in from P.S.)
RSLF	RR	
RTCF,AN	PP	
RTCFZ1N	PP	in from P. Sup.
RUNF,N,A	PP	
RUNFZ0N,Z1N	PP	
RUNN	PP	in from C.Panel
RUNT3	RR	
S00-03	KK	
S00N-15N	KK	
SC003,07,11N	KK	
SCEIN	SS	BUS
SCEIT	SS	
SEQINH	V24	
SLOADN	KK	
SPA0-3	MM	
SPYC,A	TT	BUS
SPYCOKN	TT	
SPYCADEL	TT	
SSST	V24	
STAFCL,A	RR	
STAFZ0N	RR	
START,N	RR	in from C. Panel
STARTF	RR	
STOPNB	V24	
STOV	GG	
SYNCT6	EE	
SYNIMN	EE	
SYNMEMN	TT	
T 1-10	EE	
T1,3,5,6,8N	EE	
T2,5D	EE	
T2DREPN	EE	
T4T5N	EE	
T6DN,DEN	EE	
TC810,N	EE	
TDSN	V24	
TEOC	V24	
TESTN	CC	in from C.Panel
TIMEOUT,N	TT	
TMEF	TT	
TMEN	TT	BUS (in/out)
TMER,N	TT	
TMF,N	TT	
TMMOFN	TT	
TMMCY	TT	
TMMN	TT	
TMMU,D	TT	
TMPF	TT	
TMPN	TT	BUS (in/out)
TMPR,N	TT	
TMR,N	TT	BUS (in/out)
TMRF,D	TT	

Table 2-14 CPU Signal LIST (T-W)

Signal	Logic	Comment
TMRR,N	TT	
TOUTZ0N	TT	
TPMDLN	TT	
TPMN	TT	BUS (in/out)
TPMR	TT	
TPMRDEL	TT	
TRMN	TT	BUS (in/out)
TRMR	TT	
TROHLTN	V24	
TROUBLEN	V24	
TYAC	V24	
TYAD0-5	V24	address code
TYARE,N	V24	
UNLOCK,N	PP	in from C. Panel
VALK08N	V24	
VALNA0	CC	
WRITE	TT	BUS
WRITEFN	TT	
WSP,N	RR	
WSPRSLI	RR	

Table 2-15 Microinstruction Address Code

	Hexa	Binary
0	000	000000000
	001	000000001
	002	000000010
	003	000000011
	004	000000100
	005	000000101
	006	000000110
	007	000000111
	008	000001000
	009	000001001
10	00A	000001010
	00B	000001011
	00C	000001100
	00D	000001101
	00E	000001110
	00F	000001111
	010	000010000
	011	000010001
	012	000010010
	013	000010011
20	014	000010100
	015	000010101
	016	000010110
	017	000010111
	018	000011000
	019	000011001
	01A	000011010
	01B	000011011
	01C	000011100
	01D	000011101
30	01E	000011110
	01F	000011111
	020	000100000
	021	000100001
	022	000100010
	023	000100011
	024	000100100
	025	000100101
	026	000100110
	027	000100111
40	028	000101000
	029	000101001
	02A	000101010
	02B	000101011
	02C	000101100
	02D	000101101
	02E	000101110
	02F	000101111
	030	000110000
	031	000110001
50	032	000110010
	033	000110011
	034	000110100
	035	000110101
	036	000110110
	037	000110111
	038	000111000
	039	000111001
	03A	000111010
	03B	000111011
60	03C	000111100
	03D	000111101
	03E	000111110
	03F	000111111
64	040	001000000
	041	001000001
	042	001000010
	043	001000011
	044	001000100
	045	001000101
70	046	001000110
	047	001000111
	048	001001000
	049	001001001
	04A	001001010
	04B	001001011
	04C	001001100
	04D	001001101
	04E	001001110
	04F	001001111
80	050	001010000
	051	001010001
	052	001010010
	053	001010011
	054	001010100
	055	001010101
	056	001010110
	057	001010111
	058	001011000
	059	001011001
90	05A	001011010
	05B	001011011
	05C	001011100
	05D	001011101
	05E	001011110
	05F	001011111
	060	001100000
	061	001100001
	062	001100010
	063	001100011
100	064	001100100
	065	001100101
	066	001100110
	067	001100111
	068	001101000
	069	001101001
	06A	001101010
	06B	001101011
	06C	001101100
	06D	001101101
110	06E	001101110
	06F	001101111
	070	001110000
	071	001110001
	072	001110010
	073	001110011
	074	001110100
	075	001110101
	076	001110110
	077	001110111
120	078	001111000
	079	001111001
	07A	001111010
	07B	001111011
	07C	001111100
	07D	001111101
	07E	001111110
	07F	001111111
128	080	010000000
	081	010000001
130	082	010000010
	083	010000011
	084	010000100
	085	010000101
	086	010000110
	087	010000111
	088	010001000
	089	010001001
	08A	010001010
	08B	010001011
140	08C	010001100
	08D	010001101
	08E	010001110
	08F	010001111
	090	010010000
	091	010010001
	092	010010010
	093	010010011
	094	010010100
	095	010010101
150	096	010010110
	097	010010111
	098	010011000
	099	010011001
	09A	010011010
	09B	010011011
	09C	010011100
	09D	010011101
	09E	010011110
	09F	010011111
160	0A0	010100000
	0A1	010100001
	0A2	010100010
	0A3	010100011
	0A4	010100100
	0A5	010100101
	0A6	010100110
	0A7	010100111
	0A8	010101000
	0A9	010101001
170	0AA	010101010
	0AB	010101011
	0AC	010101100
	0AD	010101101
	0AE	010101110
	0AF	010101111
	0B0	010110000
	0B1	010110001
	0B2	010110010
	0B3	010110011
180	0B4	010110100
	0B5	010110101
	0B6	010110110
	0B7	010110111
	0B8	010111000
	0B9	010111001
	0BA	010111010
	0BB	010111011
	0BC	010111100
	0BD	010111101
190	0BE	010111110
	0BF	010111111
192	0C0	011000000
	0C1	011000001
	0C2	011000010
	0C3	011000011
	0C4	011000100
	0C5	011000101
	0C6	011000110
	0C7	011000111
200	0C8	011001000
	0C9	011001001
	0CA	011001010
	0CB	011001011
	0CC	011001100
	0CD	011001101
	0CE	011001110
	0CF	011001111
	0D0	011010000
	0D1	011010001
210	0D2	011010010
	0D3	011010011
	0D4	011010100
	0D5	011010101
	0D6	011010110
	0D7	011010111
	0D8	011011000
	0D9	011011001
	0DA	011011010
	0DB	011011011
220	0DC	011011100
	0DD	011011101
	0DE	011011110
	0DF	011011111
	0E0	011100000
	0E1	011100001
	0E2	011100010
	0E3	011100011
	0E4	011100100
	0E5	011100101
230	0E6	011100110
	0E7	011100111
	0E8	011101000
	0E9	011101001
	0EA	011101010
	0EB	011101011
	0EC	011101100
	0ED	011101101
	0EE	011101110
	0EF	011101111
240	0F0	011110000
	0F1	011110001
	0F2	011110010
	0F3	011110011
	0F4	011110100
	0F5	011110101
	0F6	011110110
	0F7	011110111
	0F8	011111000
	0F9	011111001
250	0FA	011111010
	0FB	011111011
	0FC	011111100
	0FD	011111101
	0FE	011111110
	0FF	011111111
256	100	100000000
	101	100000001
	102	100000010
	103	100000011
260	104	100000100
	105	100000101
	106	100000110
	107	100000111
	108	100001000
	109	100001001
	10A	100001010
	10B	100001011
	10C	100001100
	10D	100001101
270	10E	100001110
	10F	100001111
	110	100010000
	111	100010001
	112	100010010
	113	100010011
	114	100010100
	115	100010101
	116	100010110
	117	100010111
280	118	100011000
	119	100011001
	11A	100011010
	11B	100011011
	11C	100011100
	11D	100011101
	11E	100011110
	11F	100011111
	120	100100000
	121	100100001
290	122	100100010
	123	100100011
	124	100100100
	125	100100101
	126	100100110
	127	100100111
	128	100101000
	129	100101001
	12A	100101010
	12B	100101011
300	12C	100101100
	12D	100101101
	12E	100101110
	12F	100101111
	130	100110000
	131	100110001
	132	100110010
	133	100110011
	134	100110100
	135	100110101
310	136	100110110
	137	100110111
	138	100111000
	139	100111001
	13A	100111010
	13B	100111011
	13C	100111100
	13D	100111101
	13E	100111110
	13F	100111111
320	140	101000000
	141	101000001
	142	101000010
	143	101000011
	144	101000100
	145	101000101
	146	101000110
	147	101000111
	148	101001000
	149	101001001
330	14A	101001010
	14B	101001011
	14C	101001100
	14D	101001101
	14E	101001110
	14F	101001111
	150	101010000
	151	101010001
	152	101010010
	153	101010011
340	154	101010100
	155	101010101
	156	101010110
	157	101010111
	158	101011000
	159	101011001
	15A	101011010
	15B	101011011
	15C	101011100
	15D	101011101
350	15E	101011110
	15F	101011111
	160	101100000
	161	101100001
	162	101100010
	163	101100011
	164	101100100
	165	101100101
	166	101100110
	167	101100111
360	168	101101000
	169	101101001
	16A	101101010
	16B	101101011
	16C	101101100
	16D	101101101
	16E	101101110
	16F	101101111
	170	101110000
	171	101110001
370	172	101110010
	173	101110011
	174	101110100
	175	101110101
	176	101110110
	177	101110111
	178	101111000
	179	101111001
	17A	101111010
	17B	101111011
380	17C	101111100
	17D	101111101
	17E	101111110
	17F	101111111
384	180	110000000
	181	110000001
	182	110000010
	183	110000011
	184	110000100
	185	110000101
390	186	110000110
	187	110000111
	188	110001000
	189	110001001
	18A	110001010
	18B	110001011
	18C	110001100
	18D	110001101
	18E	110001110
	18F	110001111
400	190	110010000
	191	110010001
	192	110010010
	193	110010011
	194	110010100
	195	110010101
	196	110010110
	197	110010111
	198	110011000
	199	110011001
410	19A	110011010
	19B	110011011
	19C	110011100
	19D	110011101
	19E	110011110
	19F	110011111
	1A0	110100000
	1A1	110100001
	1A2	110100010
	1A3	110100011
420	1A4	110100100
	1A5	110100101
	1A6	110100110
	1A7	110100111
	1A8	110101000
	1A9	110101001
	1AA	110101010
	1AB	110101011
	1AC	110101100
	1AD	110101101
430	1AE	110101110
	1AF	110101111
	1B0	110110000
	1B1	110110001
	1B2	110110010
	1B3	110110011
	1B4	110110100
	1B5	110110101
	1B6	110110110
	1B7	110110111
440	1B8	110111000
	1B9	110111001
	1BA	110111010
	1BB	110111011
	1BC	110111100
	1BD	110111101
	1BE	110111110
	1BF	110111111
448	1C0	111000000
	1C1	111000001
450	1C2	111000010
	1C3	111000011
	1C4	111000100
	1C5	111000101
	1C6	111000110
	1C7	111000111
	1C8	111001000
	1C9	111001001
	1CA	111001010
	1CB	111001011
460	1CC	111001100
	1CD	111001101
	1CE	111001110
	1CF	111001111
	1D0	111010000
	1D1	111010001
	1D2	111010010
	1D3	111010011
	1D4	111010100
	1D5	111010101
470	1D6	111010110
	1D7	111010111
	1D8	111011000
	1D9	111011001
	1DA	111011010
	1DB	111011011
	1DC	111011100
	1DD	111011101
	1DE	111011110
	1DF	111011111
480	1E0	111100000
	1E1	111100001
	1E2	111100010
	1E3	111100011
	1E4	111100100
	1E5	111100101
	1E6	111100110
	1E7	111100111
	1E8	111101000
	1E9	111101001
490	1EA	111101010
	1EB	111101011
	1EC	111101100
	1ED	111101101
	1EE	111101110
	1EF	111101111
	1F0	111110000
	1F1	111110001
	1F2	111110010
	1F3	111110011
500	1F4	111110100
	1F5	111110101
	1F6	111110110
	1F7	111110111
	1F8	111111000
	1F9	111111001
510	1FA	111111010
	1FB	111111011
	1FC	111111100
	1FD	111111101
	1FE	111111110
	1FF	111111111



	µSNA 0 1	PLAVALN	FU µNA7	PLA 0 1	Sub-Mode
	0 0 0 1	h		0 0	Inhibit
Explicit	1 0		0 0	1 0	Add Master
Implicit	1 1	L	1 0	1 1	Add User
	RADETPLN		− 1	0 1	Execution

K01 NN (CR)
K04 CC (PAGE FAULT/MOVE TABLE)
K08 DD (FLAG SELECT)
K05–08, K11–15, K11–14N } MM (S.P. ADDRESS)
K4,8,10–15, K4,8–15N } (V24)

Figure 2-8AA Instruction Word Logic

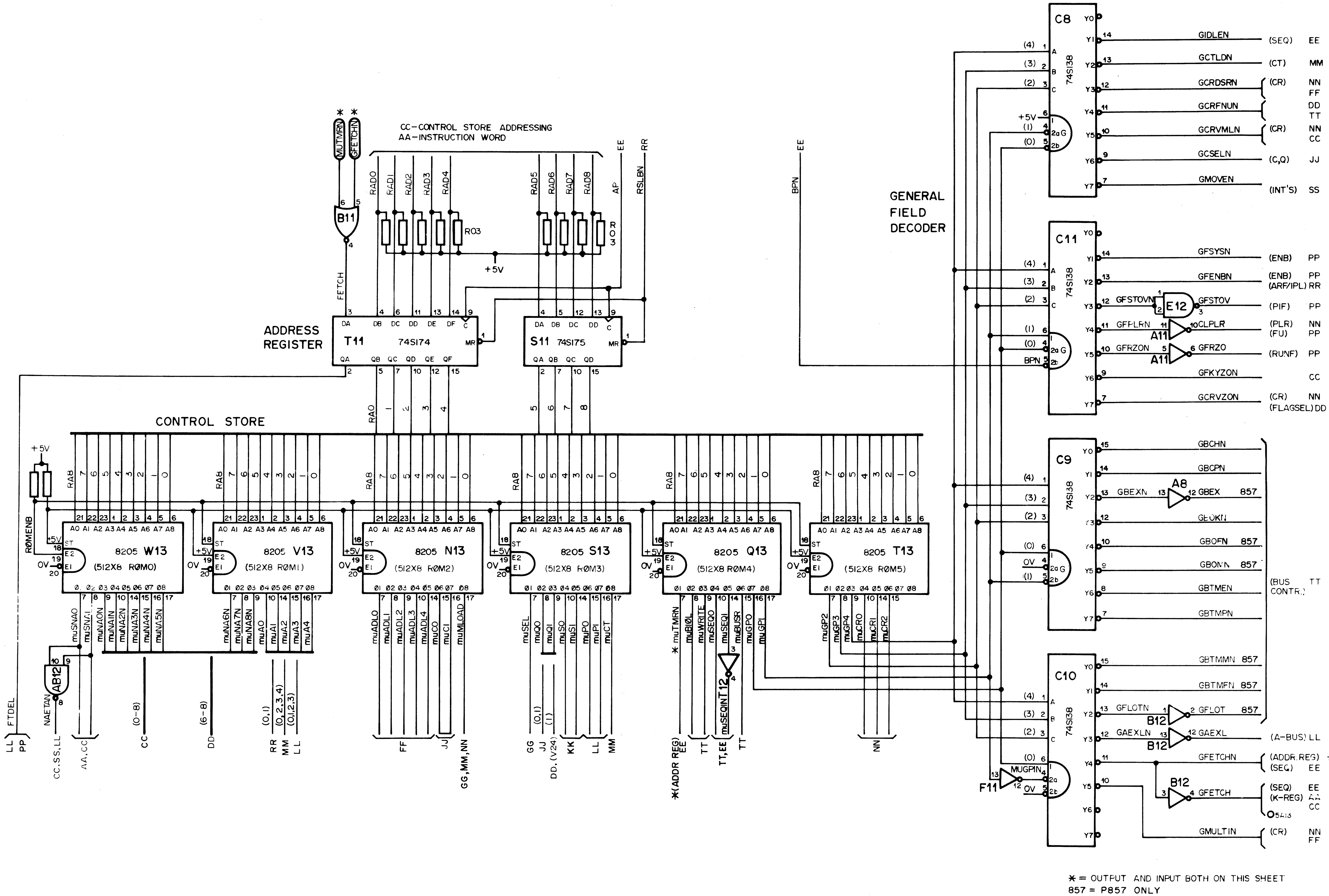


Figure 2-8BB Microcommand Control-Store Logic

µSNA 0 1	Mode	Function
0 0	Explicit	Explicit address is µNA0-8.
0 1	Flag	Flag from execution-pointer tests (selected by µNA6-8) sets RAD8; µNA0-7 to RAD0-7.
1 0	Instruction Word (PLA)	Instruction word (K-reg, decoder) provides address RAD1-8; µNA7 modifies the decoder for fetch or execution.
1 1	Machine-State	The machine-state pointer provides address RAD4-8; µNA0-3 to RAD0-3.
x x	Move Table/Fault: an interrupt or a page fault during a Move Table instruction force a special microinstruction address with bits RAD6-8.	

µSNA0,1 RAD 0 7 8

0 1 µNA0-7

µNA6,7,8N	Flag
0 0 0	Q00
0 0 1	IR
0 1 0	PM1
0 1 1	K08
1 0 0	FNU
1 0 1	NORM
1 1 0	DIV
1 1 1	FU15R2

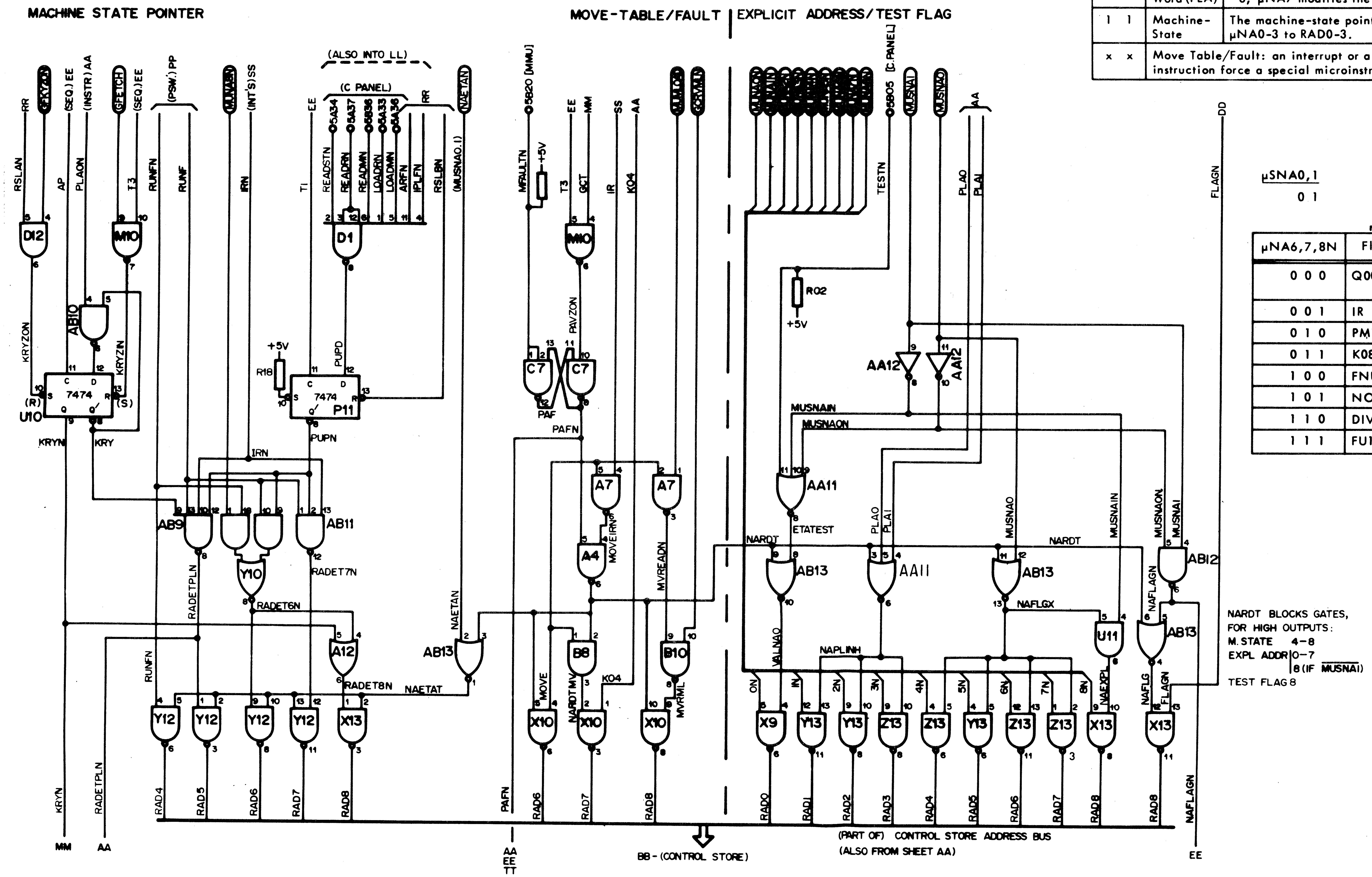
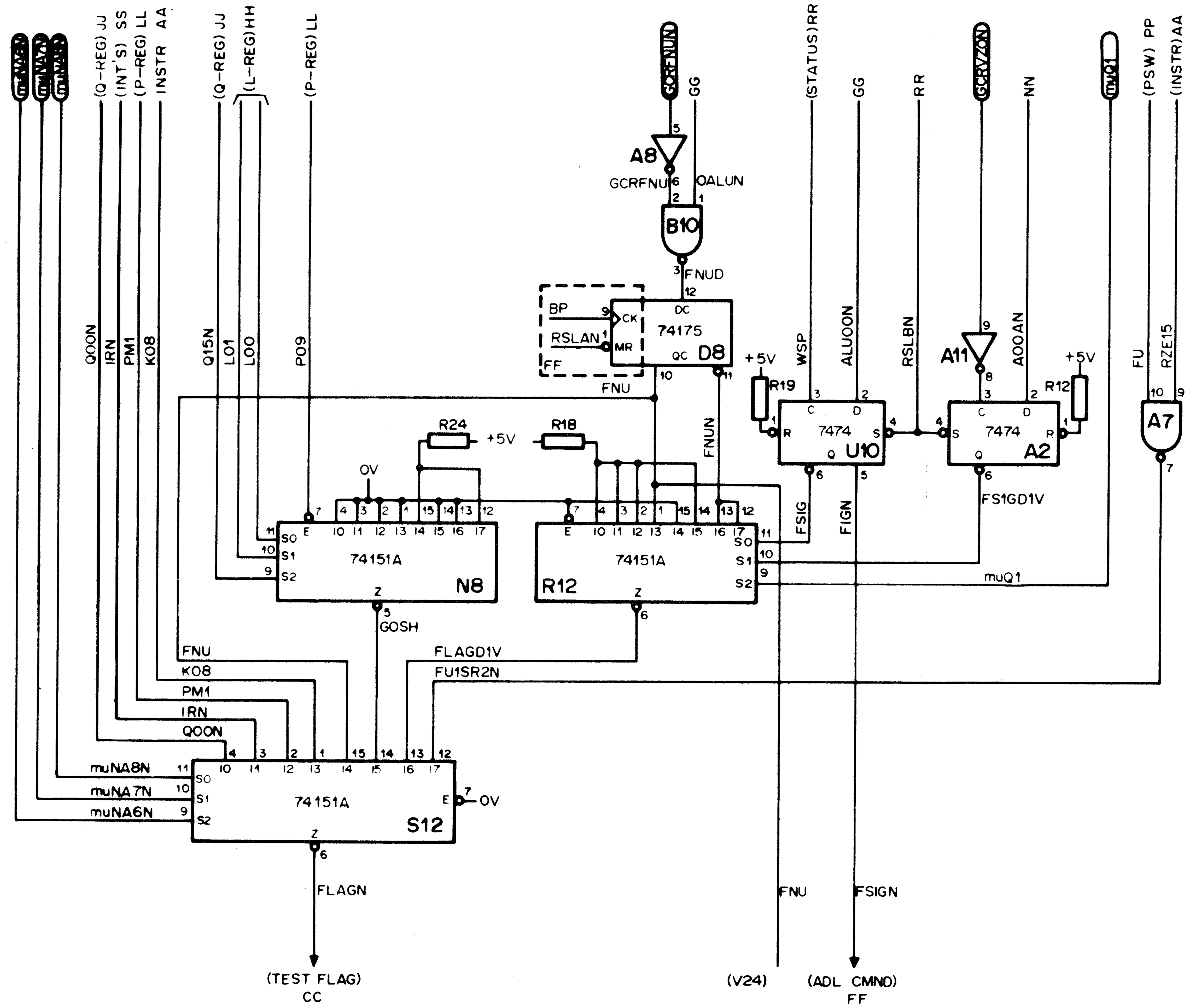


Figure 2-8CC Control Store Addressing



GOSH Select

Q15N S2	L01 S1	L00 S0		
L	0	0	0	0
L	0	1	1	0
L	1	0	2	0
L	1	1	3	0
h	0	0	4	+5
h	0	1	5	0
h	1	0	6	0
h	1	1	7	+5

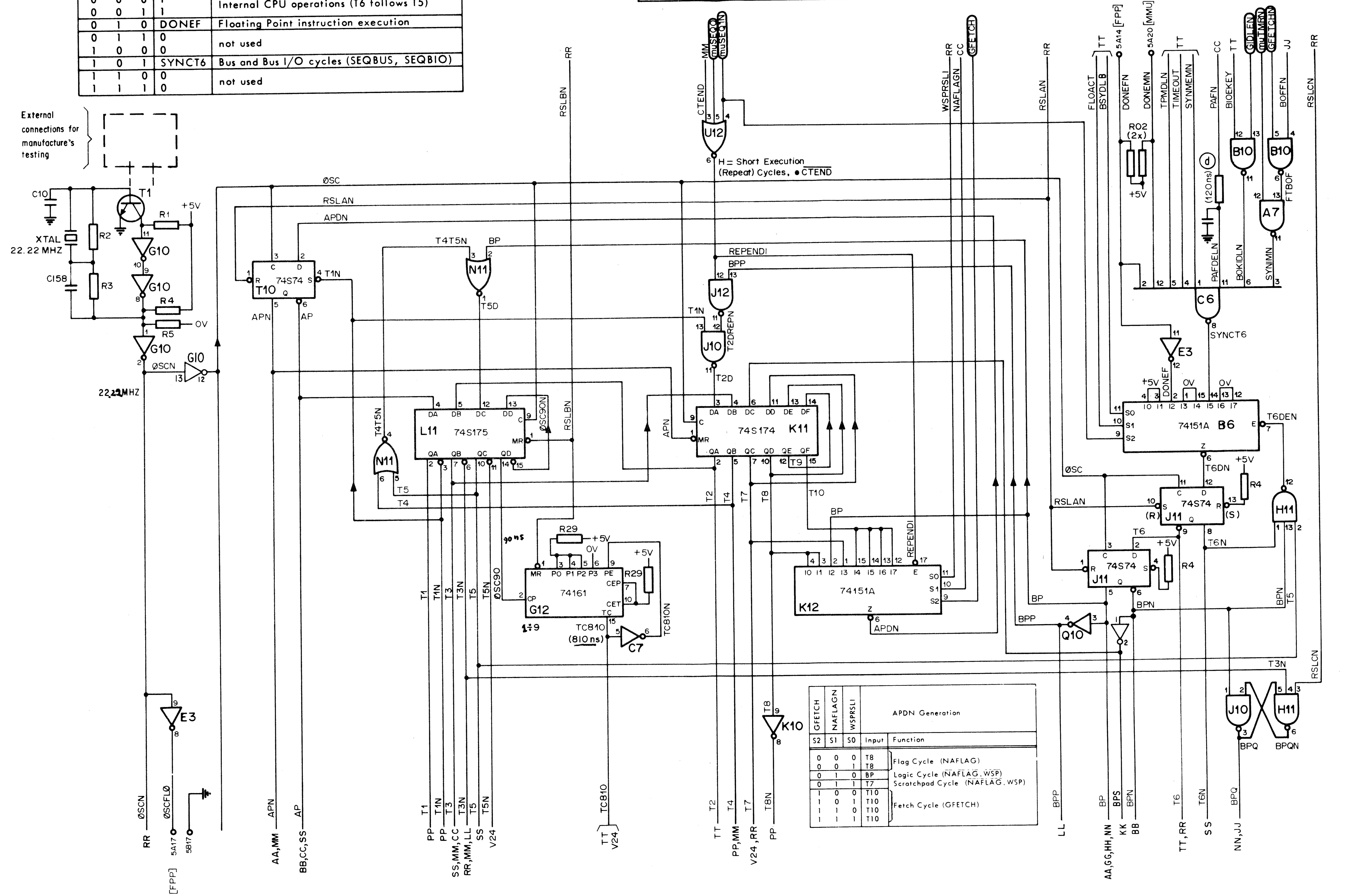
FLAGDIV Select

	μQ1 S2	FSIGDIV S1	FSIG S0		
DIVB	0	0	0	0	+5
	0	0	1	1	+5
	0	1	0	2	+5
	0	1	1	3	FNU --- FSIGDIV.FSIG.FNUN
DIVA	1	0	0	4	0 --- FSIGDIV.FSIG
	1	0	1	5	+5
	1	1	0	6	FNUN --- FSIGFIV.FNU
	1	1	1	7	FNUN

Figure 2-8DD Flag Select

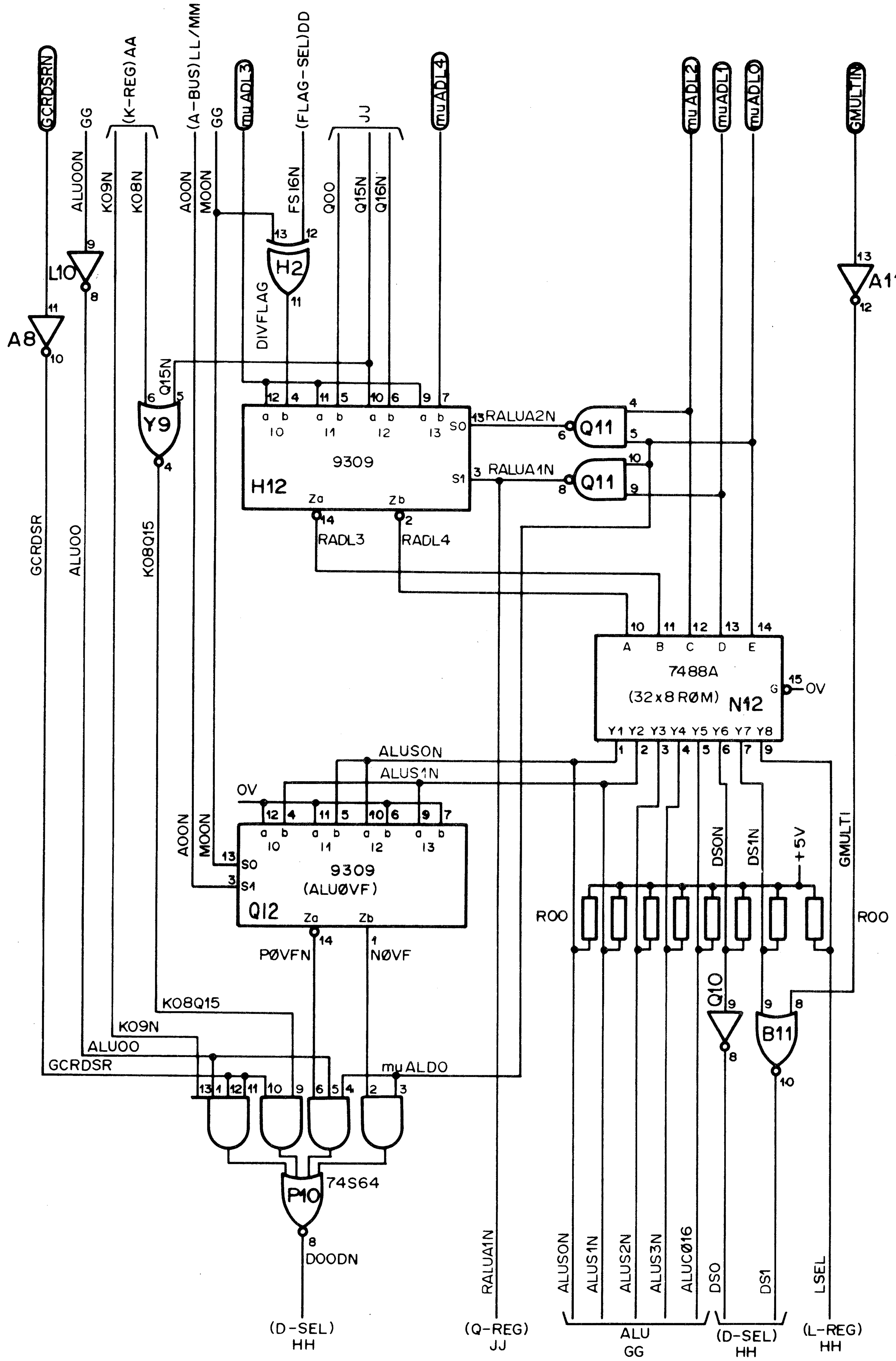
μSEQ0 S2	FLOACT S1	BSYDL S0	Input	Function (T6DN Generation)
0	0	0	1	Internal CPU operations (T6 follows T5)
0	0	1	1	
0	1	0	DONEF	Floating Point instruction execution
0	1	1	0	not used
1	0	0	0	
1	0	1	SYNCT6	Bus and Bus I/O cycles (SEQBUS, SEQBIO)
1	1	0	0	not used
1	1	1	0	

mu SEQ 0,1	Mnemonic	Function
0 0	LOGIC	Internal CPU Operations
0 1	REPEAT	Short Execution Cycle
1 0	SEQBUS	GP-Bus Cycles
1 1	SEQBIO	I/O Cycles with Bus not released



GFETCH S2	NAFLAGN S1	WSPRSLI S0	Input	Function (APDN Generation)
0	0	0	T8	Flag Cycle (NAFLAG)
0	0	1	T8	
0	1	0	BP	Logic Cycle (NAFLAG.WSP)
0	1	1	T7	Scratchpad Cycle (NAFLAG.WSP)
1	0	0	T10	Fetch Cycle (GFETCH)
1	0	1	T10	
1	1	0	T10	
1	1	1	T10	

Figure 2-8EE Sequensor (CPU Clock)



H12 Multiplexer

Multiplexer Select Input		Multiplexer Output to ADL Command ROM	
μADL: 0, 1, 2	RALUA: 1N, 2N	RADL3 (B)	RADL4 (A)
0 0 0	1 1	μADL3N	μADL4N
0 0 1	1 1	μADL3N	μADL4N
0 1 0	1 1	μADL3N	μADL4N
0 1 1	1 1	μADL3N	μADL4N
1 0 0	1 1	μADL3N	μADL4N
1 0 1	1 0	Q15	Q16
1 1 0	0 1	μADL3N	FSIGN
1 1 1	0 0	μADL3N	DIVFLAGN

DIVFLAG = FSIGN ⊕ M00N

M Select (Logic GG)

Enable Clock	Select C or Q	Mnemonic	Data Source	
μMLOAD	μMSEL			
0	-	No Op	Off; previously stored data available at output.	
1	0	MYC	C 00	15
1	1	MYQ	Q 00	15
			M 00	15

D Select (Logic HH)

S1 S0			
DS0, 1			
0 0 :	ALU 00-15N		ALU direct
0 1 :	ALU 08-15N	ALU 00-07N	Exchange ALU characters
1 0 :	DOODN	ALU 00-14N	ALU shift right
1 1 :	BIO 00-15AN		BIO direct
D	00 07	08 15	N

L Select (Logic HH)

LSEL	Data Source		Function
0	D 00-15		D direct
1	D 01-15	Q00	D shifted left
L	00 —— 14	15	

ALU Function Table							(Logic GG)
	ADL-Control ROM						
μADL4	ALUCO16	ALUSO-----3				Operation	
CE	CIN	S3	S2	S1	S0		
0	0	1	0	0	1	A plus B	ARITHMETIC
0	0	1	0	1	1	A or B	
0	0	1	1	0	0	A plus A	
0	1	0	0	0	1	A and B	
0	1	0	0	1	1	Zero	
0	1	0	1	1	0	A minus B	
1	-	0	1	0	1	B inverted	LOGIC
1	-	1	0	0	1	A ⊕ B	
1	-	1	0	1	0	B	
1	-	1	1	1	1	A	
1	-	1	0	0	1	A ⊕ B + 1	

Control signals are all active high.

Functions are for active-low data in and out.

Figure 2-8FF A, D, L Command Logic

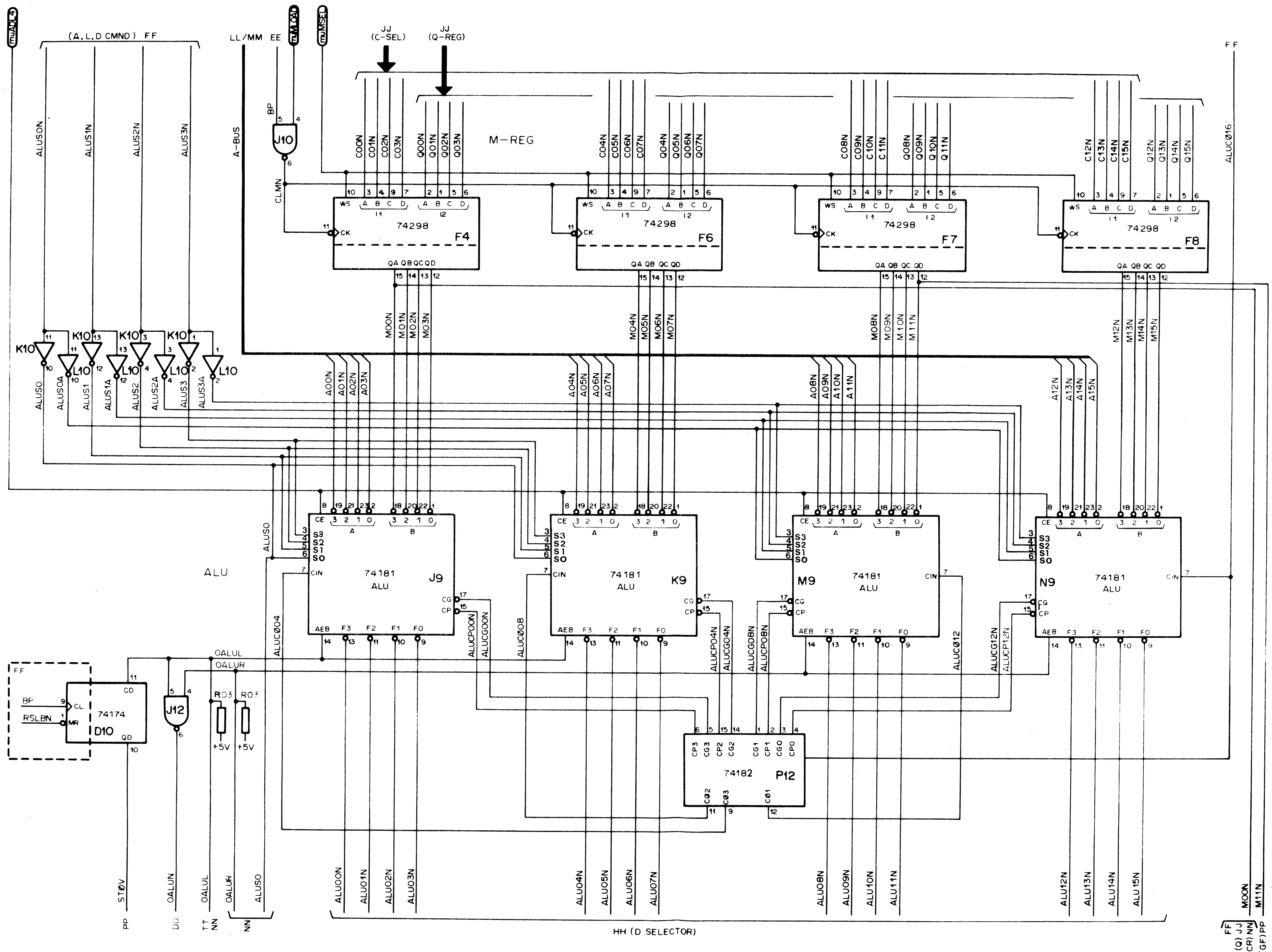


Figure 2-8GG ALU,M Register

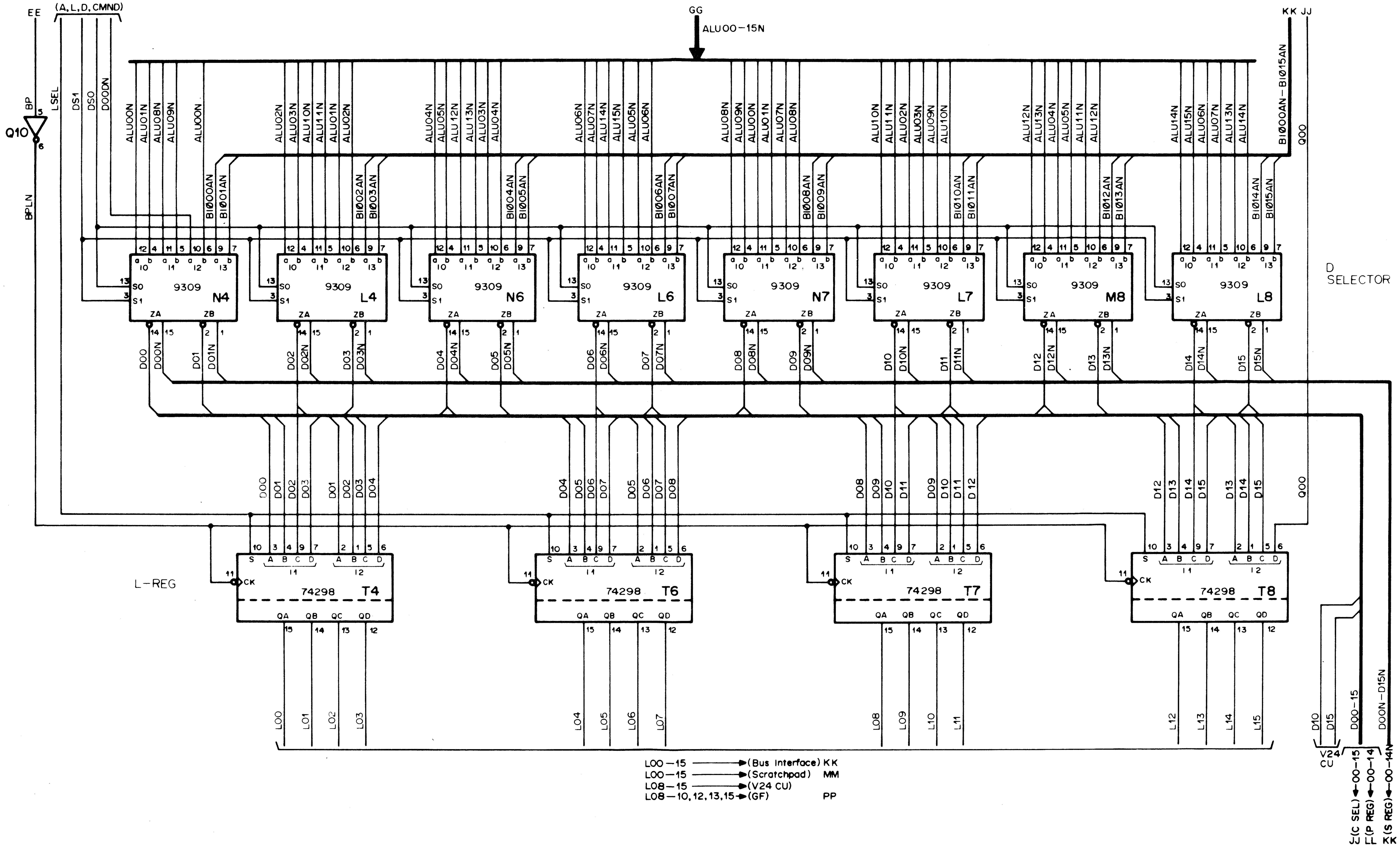
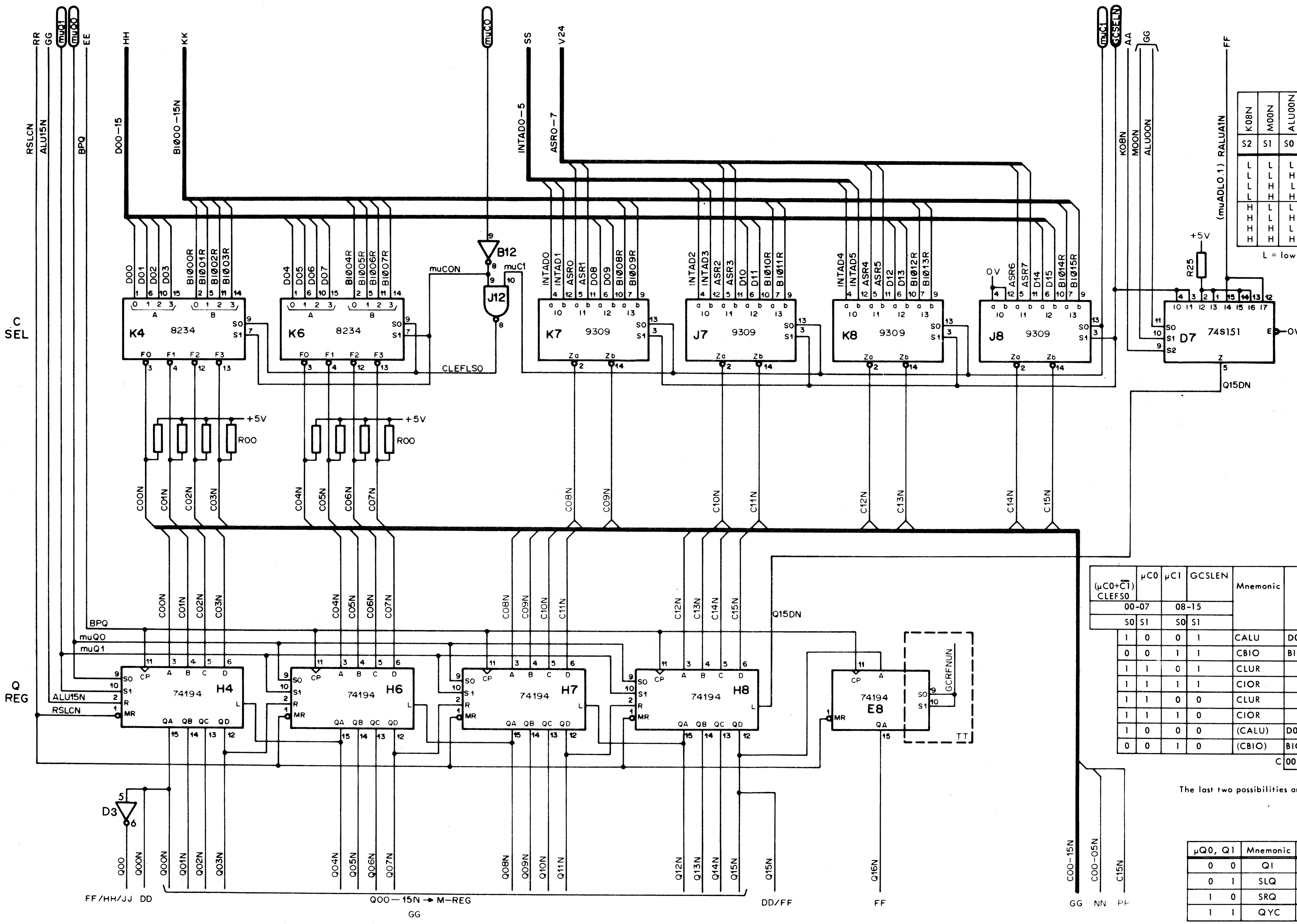


Figure 2-8HH D-Selector, L-Register



K08N S2	M00N S1	ALU00N S0	Q15DN =	
L	L	L	GCSELN	DLC
L	L	H	GCSELN	DLC
L	H	L	H	DLC
L	H	H	H	DLC
H	L	L	(μADL0·1)	Divide
H	L	H	H	Divide
H	H	L	H	Divide
H	H	H	(μADL0·1)	Divide

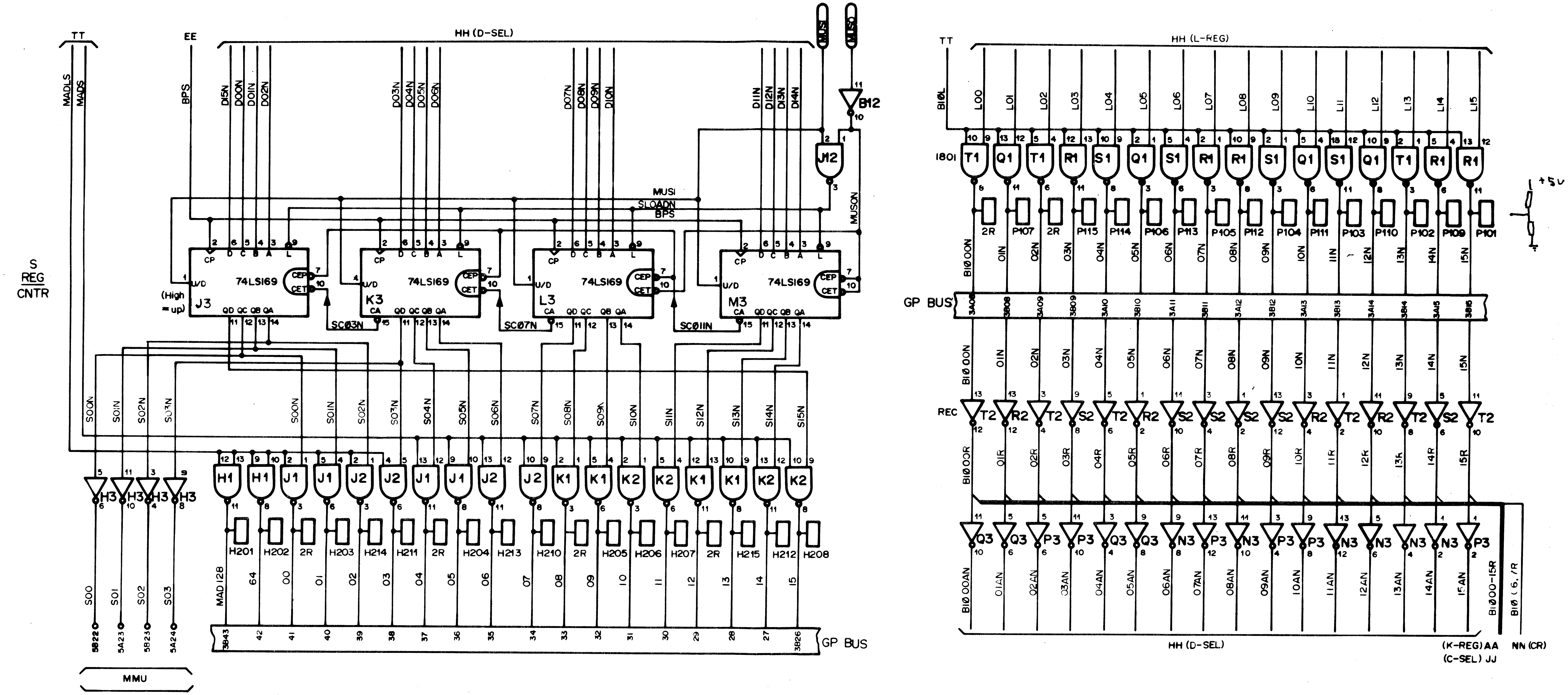
L = low = active

($\mu C0+\overline{C1}$) CLEFS0	μC0	μC1	GCSLEN	Mnemonic	Source	
00-07		08-15				
S0	S1	S0	S1			
1	0	0	1	CALU	D00 15	
0	0	1	1	CBIO	BIO00 15	
1	1	0	1	CLUR		D08 15
1	1	1	1	CIOR		BIO•08 15
1	1	0	0	CLUR		INTAD0 5 0 0
1	1	1	0	CIOR		ASR0 7
1	0	0	0	(CALU)	D00 07	INTAD0 5 0 0
0	0	1	0	(CBIO)	BIO00 07	ASR0 7
				C	00N 07N	08N 15N

The last two possibilities are not used.

μQ0, Q1		Mnemonic	Function
0	0	QI	Hold contents (No Op)
0	1	SLQ	Shift Left Q
1	0	SRQ	Shift Right Q
1	1	QYC	Parallel load Q from C

Figure 2-8JJ C-Selector, Q-Register



µS0, S1	Mnemonic	Function
0 0	SI	Inhibit counting or loading
0 1	SYD	Load S0-15N with D0-15N
1 0	SP2	S plus 2 (count down)
1 1	SM2	S minus 2 (count up)

Figure 2-8KK S-Register/Counter, Bus Interface

µA0,A1,A2,A3	9301 OUTPUT	Selected SOURCE
0 0 0 0	0 A Bus = 0	none
0 0 0 1	1 PLN	IPL
0 0 1 0	2 PN	P
0 0 1 1	3	PSW
0 1 0 0	4 NBN	(K5-8)
0 1 0 1	5 CONSTN	*
0 1 1 0	6 PUPN	
0 1 1 1	7 SYSH	
1 0 0 0	8 not connected	Scratchpad OFF
1 0 0 1	9 not connected	Scratchpad OFF
1 0 1 0	10 not connected	Scratchpad OFF

* µA4=1: 00010=2_{10} (/002)
µA4=0: 10010=18_{10} (/012)

8234 CHIPS

S0	S1	
L	L	B
L	H	B
H	L	A
H	H	High Out (Off)

µP0,P1	Mnemonic	Function
0 0	PI	Inhibit counting or loading
0 1	PYD	Load P0-14 with D0-14
1 0	PM2	P minus 2 (count down)
1 1	PP2	P plus 2 (count up)

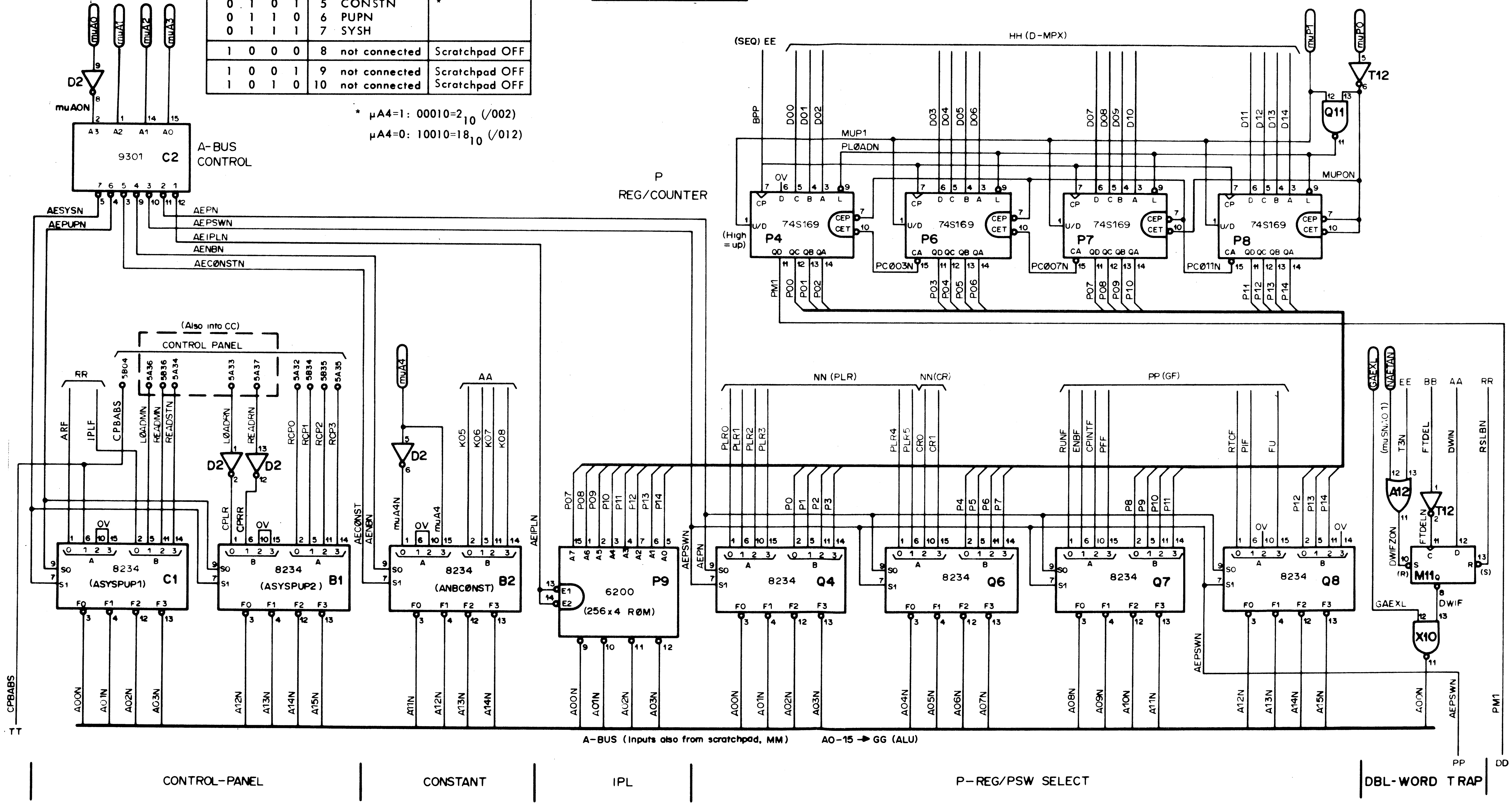
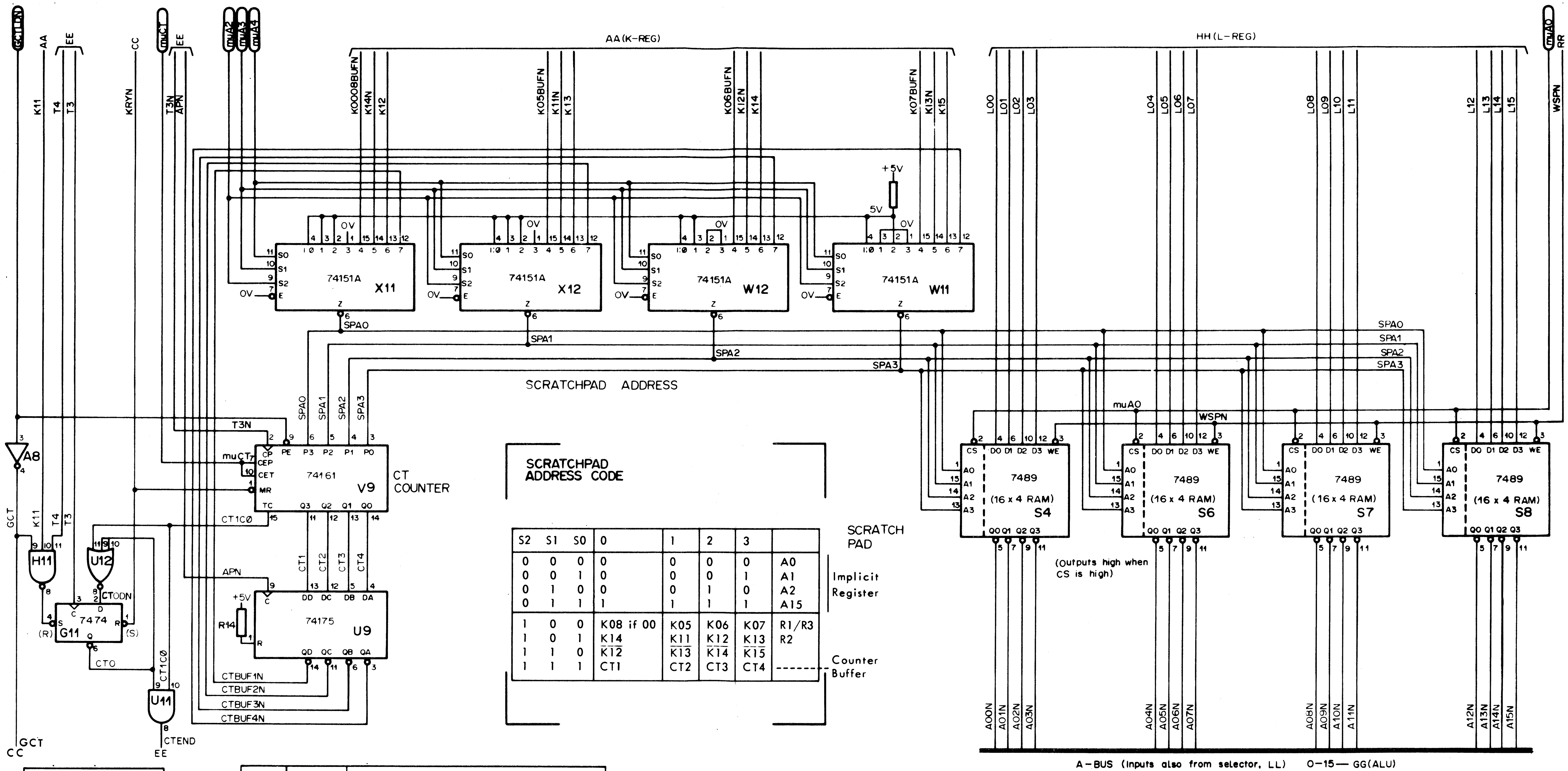


Figure 2-8LL A-Bus Selection, IPL, P-Register/Counter



SCRATCHPAD ADDRESS CODE

S2	S1	S0	0	1	2	3		
0	0	0	0	0	0	0	A0	Implicit Register
0	0	1	0	0	0	1	A1	
0	1	0	0	0	1	0	A2	
0	1	1	1	1	1	1	A15	
1	0	0	K08 if 00	K05	K06	K07	R1/R3	
1	0	1	K14	K11	K12	K13	R2	
1	1	0	$\overline{K12}$	$\overline{K13}$	$\overline{K14}$	$\overline{K15}$		
1	1	1	CT1	CT2	CT3	CT4		Counter Buffer

CT0D = CARRY+CT0:
CT0,1,2,3,4
0 1 1 1 0
0 1 1 1 1 (carry)
1 0 0 0 0
1 0 0 0 1

μCT (H=1)	GCTLDN (L=1)	Function
0	0	No change
0	1	Load CT1-4 from SPA0-3; CT0 from K11
1	0	Count CT at each T3N

Figure 2-8MM Scratchpad

74157 Input	Select A00	74151 In	Select K01	ALU00	A01	Function	
(+5V) 1	0 (11)	0	0	1	1	overflow	Shift
µMLOAD	1 (10)	2	0	0	1	OK	
µMLOAD	0 (11)	1	0	1	0	OK	
(+5V) 1	1 (10)	3	0	0	0	overflow	
GCRDSRN	0 (11)	4	1	1	1	overflow	Divide
OVFMUN	1 (10)	5	1	1	0	OK	
(+5V) 1	0 (11)	6	1	0	1	OK	
GCRDSRN	1 (10)	7	1	0	0	overflow	

OVFMUN = GMULTI. $\overline{\text{ALUS0}}$. M00

CR INPUT ROM CODE

ROM Addressing 0ALUL	0ALUR	ALU00N	A00N	M00N	ROM Output to CR Input Selector LOGIC CR1,5	ADD CR2,6	SUB CR3,7	COMP CR4,8	
0	0	0	0	0	1 0	1 0	1 0	1 0	
0	0	0	0	1	1 0	1 0	1 0	1 0	
0	0	0	1	0	1 0	1 0	1 1	0 1	
0	0	0	1	1	1 0	1 1	1 0	1 0	
0	0	1	0	0	0 1	1 1	0 1	0 1	
0	0	1	0	1	0 1	0 1	1 1	1 0	
0	0	1	1	0	0 1	0 1	0 1	0 1	
0	0	1	1	1	0 1	0 1	0 1	0 1	
0	1	0	0	0	1 0	1 0	1 0	1 0	
0	1	0	0	1	1 0	1 0	1 0	1 0	
0	1	0	1	0	1 0	1 0	1 1	0 1	
0	1	0	1	1	1 0	1 1	1 0	1 0	
0	1	1	0	0	0 1	1 1	0 1	0 1	
0	1	1	0	1	0 1	0 1	1 1	1 0	
0	1	1	1	0	0 1	0 1	0 1	0 1	
0	1	1	1	1	0 1	0 1	0 1	0 1	
1	0	0	0	0	1 1	1 1	1 1	1 1	not used
1	0	0	0	1	1 1	1 1	1 1	1 1	
1	0	0	1	0	1 1	1 1	1 1	1 1	
1	0	0	1	1	1 1	1 1	1 1	1 1	
1	0	1	0	0	0 1	1 1	0 1	0 1	
1	0	1	0	1	0 1	0 1	1 1	1 0	
1	0	1	1	0	0 1	0 1	0 1	0 1	
1	0	1	1	1	0 1	0 1	0 1	0 1	

CR INPUT SELECTION

Selection µCR0,1,2	Mnemonic	Input to CR CR0	CR1	Function
0 0 0	CRNC	CR0	CR1	No change
0 0 1	CRRTN	BIO06R	BIO07R	CR load during Return
0 1 0	CRIO	Timeout	ACBN	Programmed channel loading
0 1 1	CRFLO	FLOCR0	FLOCR1	Floating Point loading
1 0 0	CRLOG	ROMCR1	ROMCR5	Logic operation loading
1 0 1	CRADD	ROMCR2	ROMCR6	Addition operation loading
1 1 0	CRSUB	ROMCR3	ROMCR7	Subtraction operation loading
1 1 1	CRCMP	ROMCR4	ROMCR8	Compare operation loading

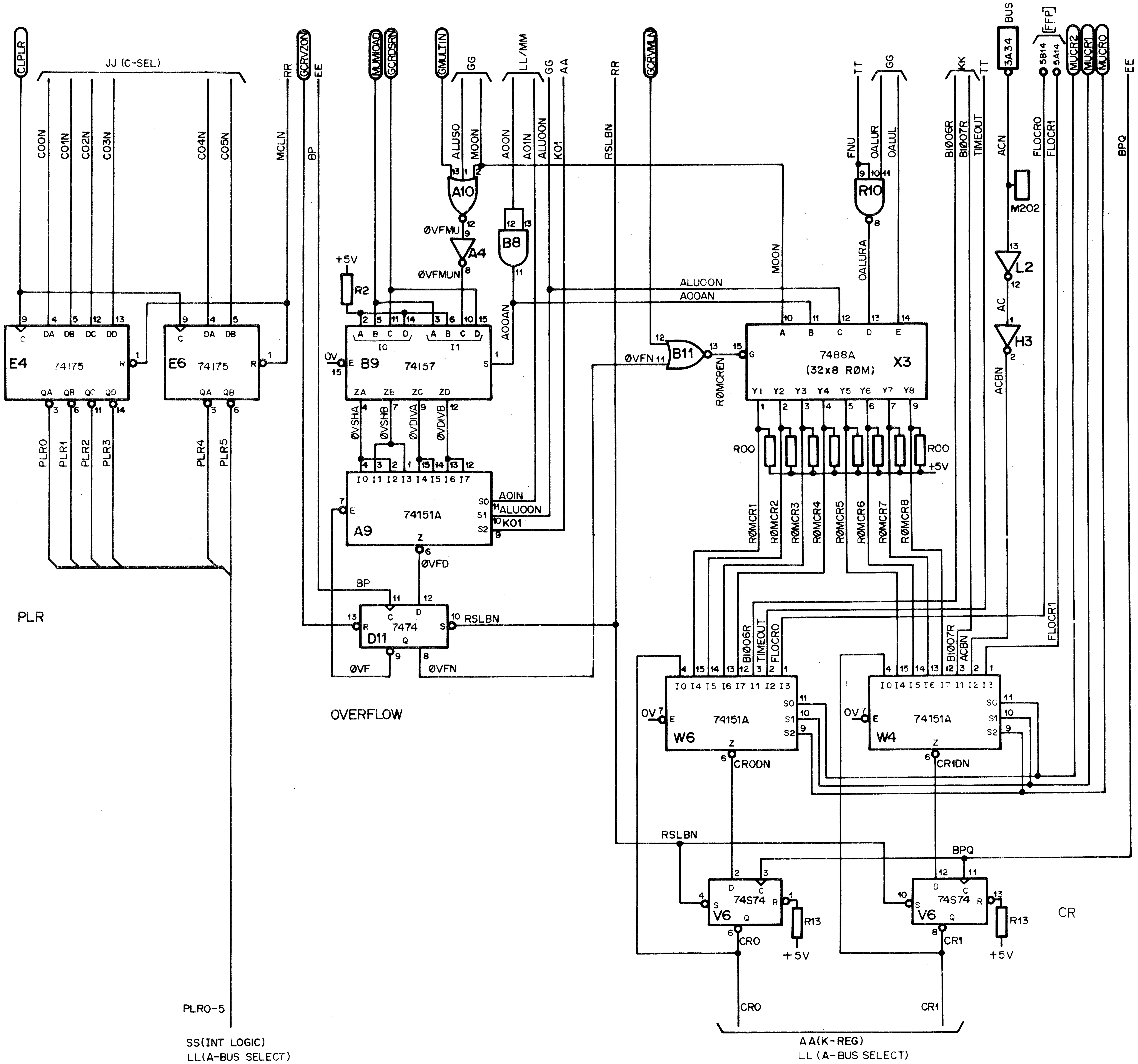
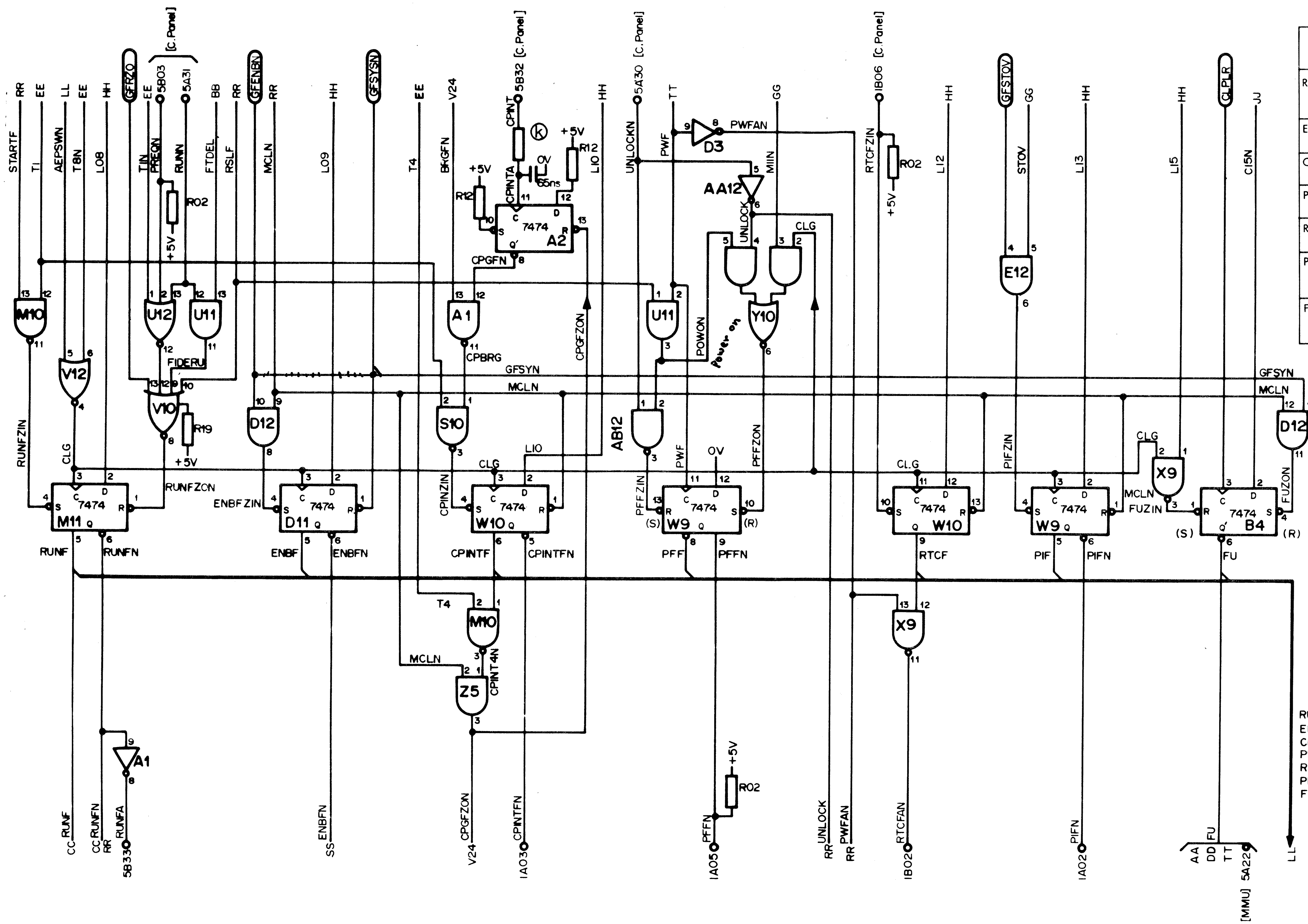


Figure 2-8NN PSW, PLR, CR



GF	PSW Bit	Function	Loaded at CLG (set or reset)	Set by	Reset by
RUNF	8	Main CPU status pointer	L08	STARTF.T1	GFRZO+(PREQ.RUN.T1)+(FTDEL.$\overline{RUN}$)+RSLF
ENBF	9	Enable interrupts	L09	GFENBN.$\overline{MCLN}$	GFSYSN
CPF	10	Operator's Interrupt	L10	(CPINT+V24INT).T1	MCLN
PFF	11	Power Failure Interrupt	$\overline{MI1}$ resets	PWF	PWF.RSLF.UNLOCK
RTCF	12	Real Time Clock Interrupt	L12	RTCFZIN from ALU	MCLN
PIF	13	Op-Code (Program) Interrupt	L13	GSTOV.STOV ~~from C Panel~~ LCM	MCLN
FU	15	User Mode (0=System Mode)	CLPLR.C15	L15.CLG	GFSYSN+MCLN

(k) Delay circuit on sheet uu

Figure 2-8PP PSW -- GF (General Flip-Flops)

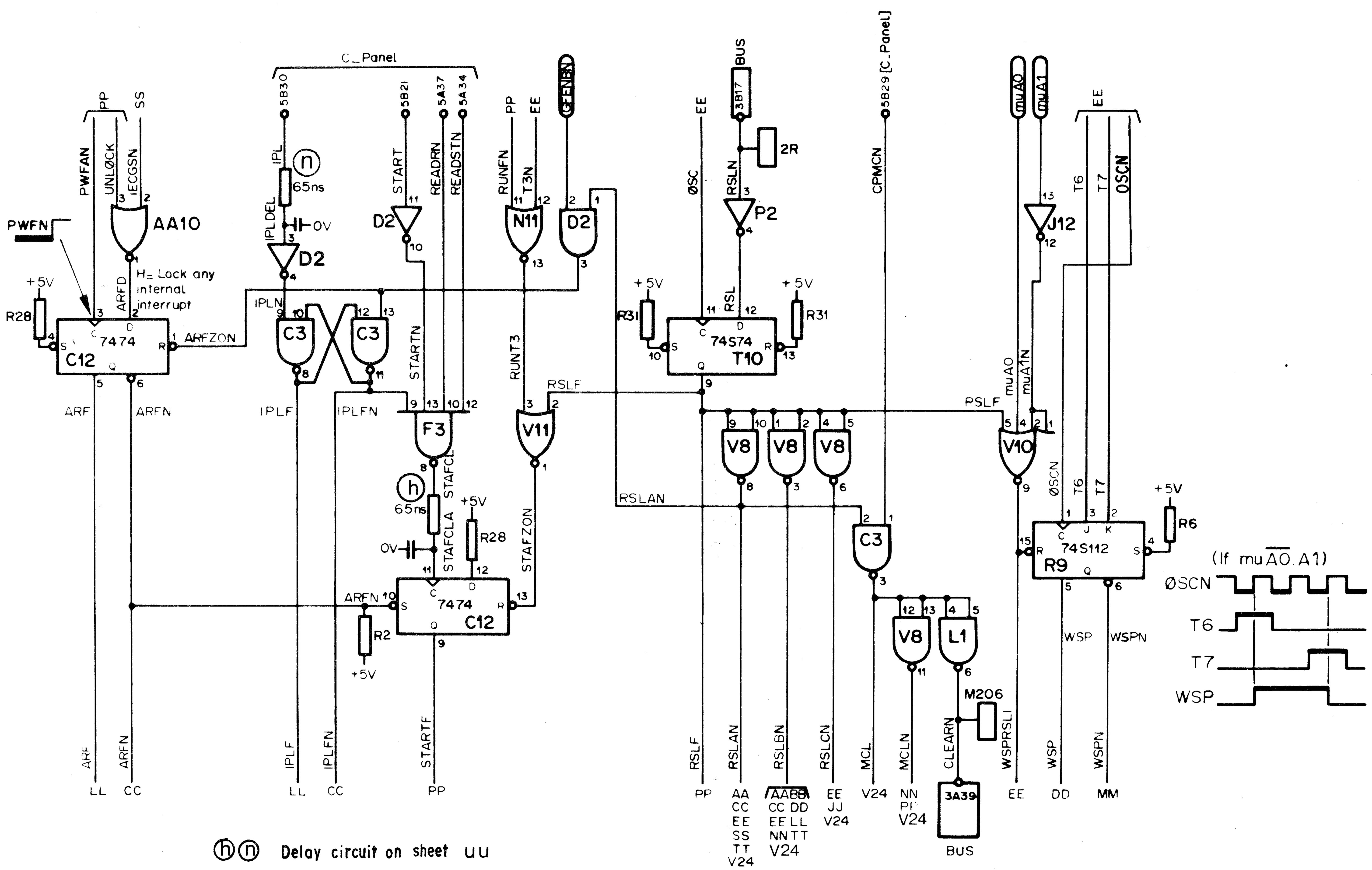
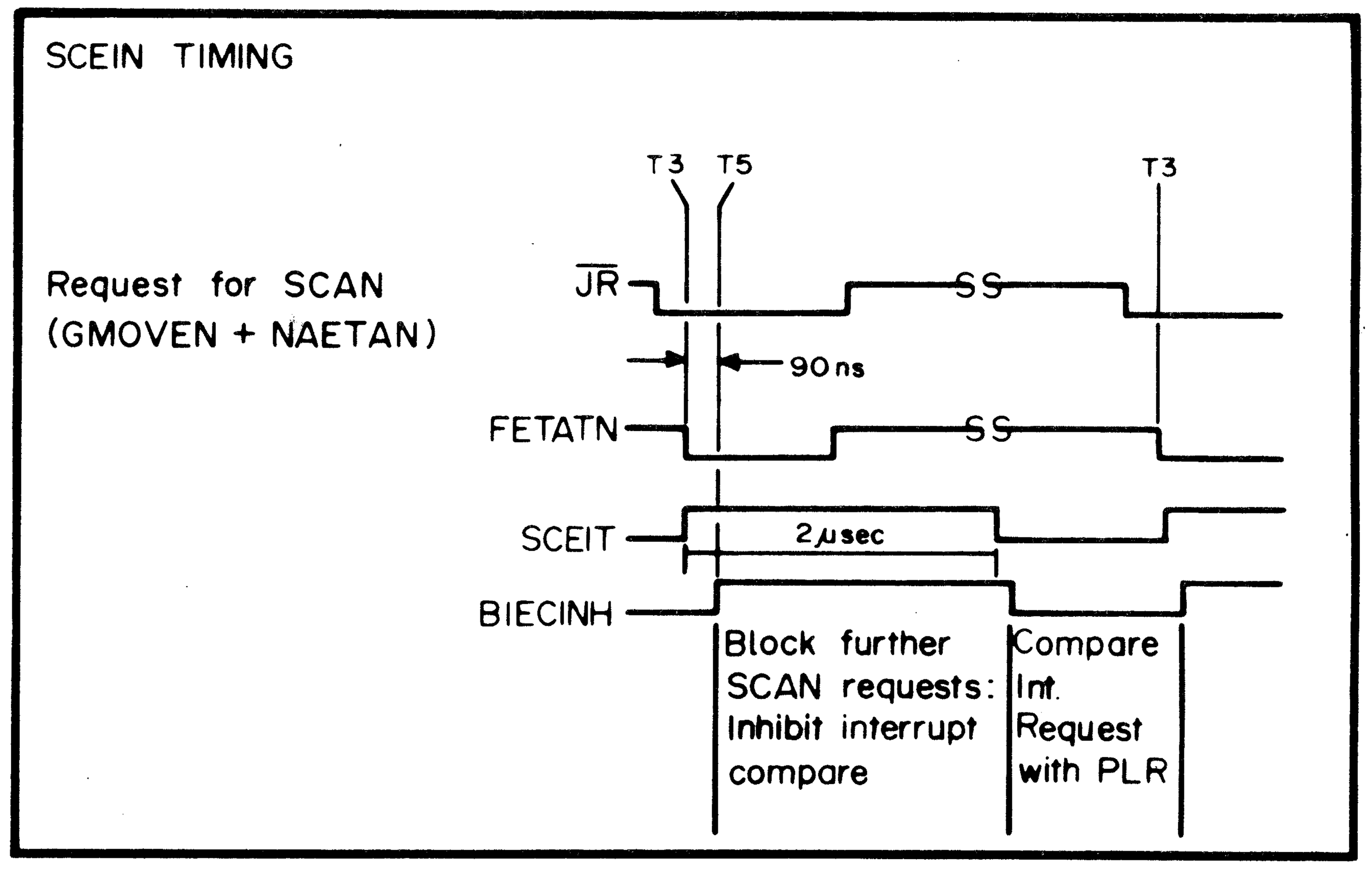


Figure 2-8RR Start, Reset



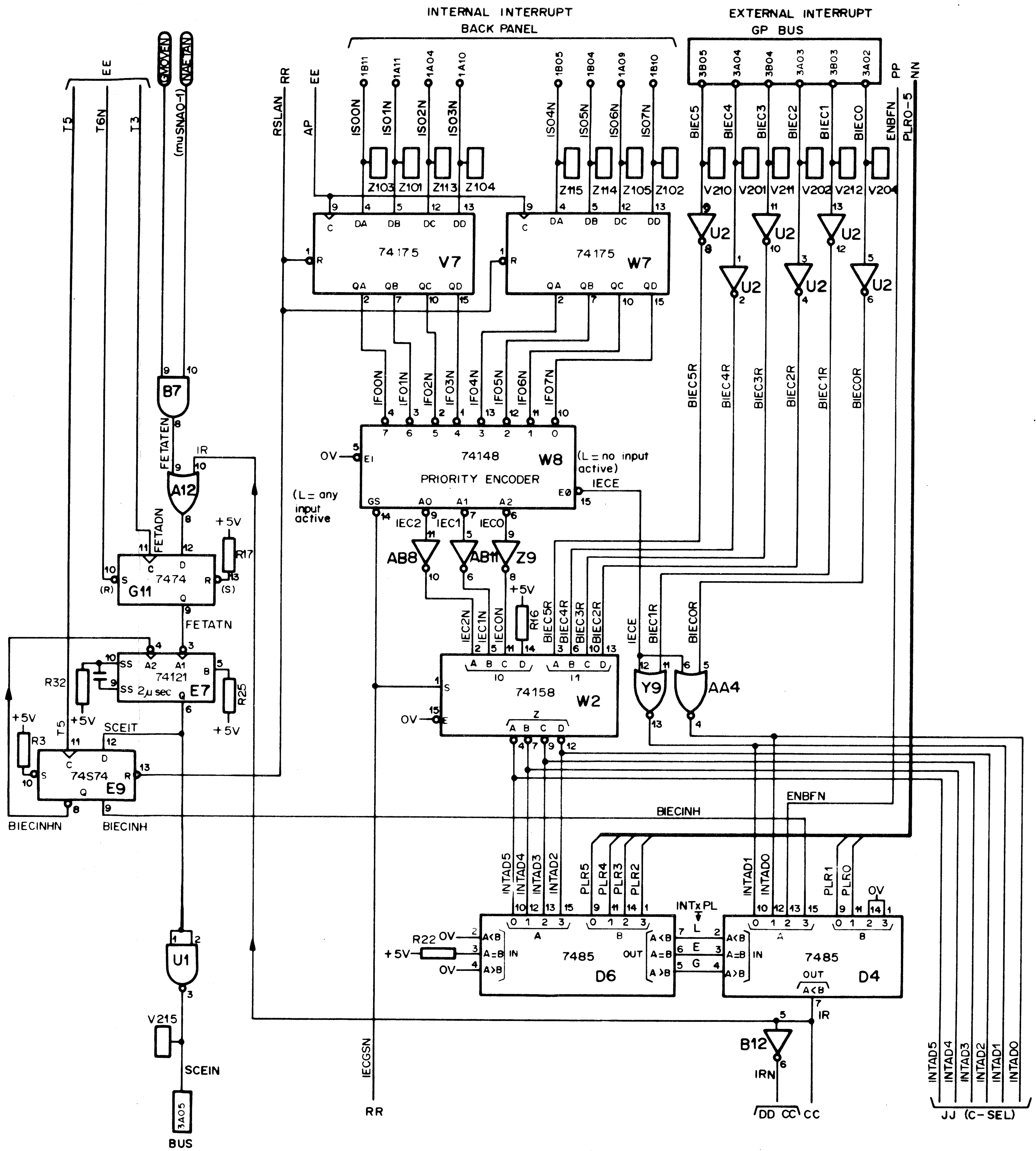
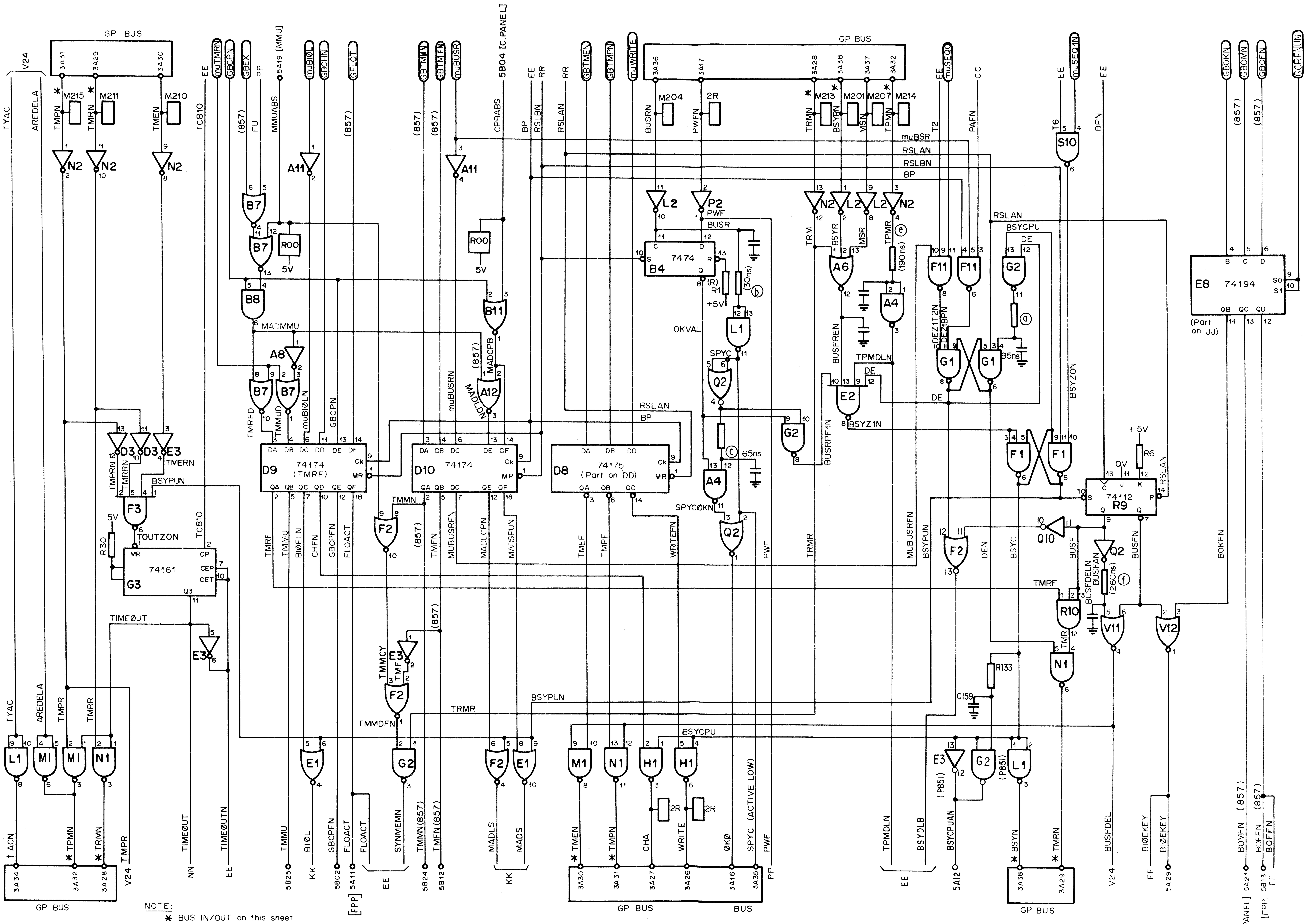
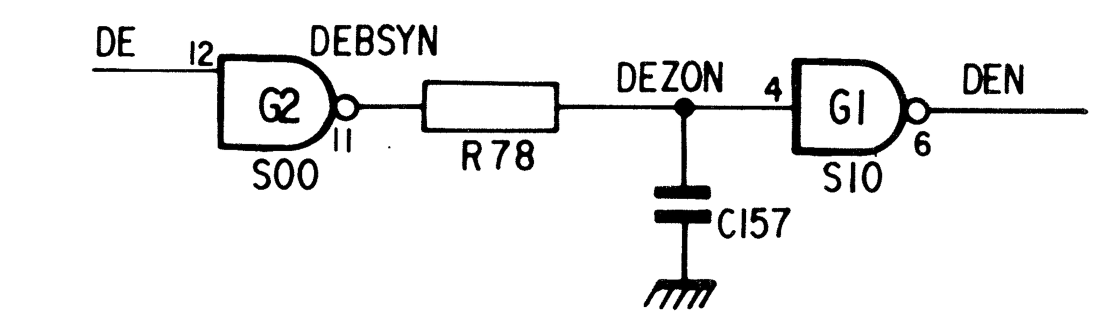
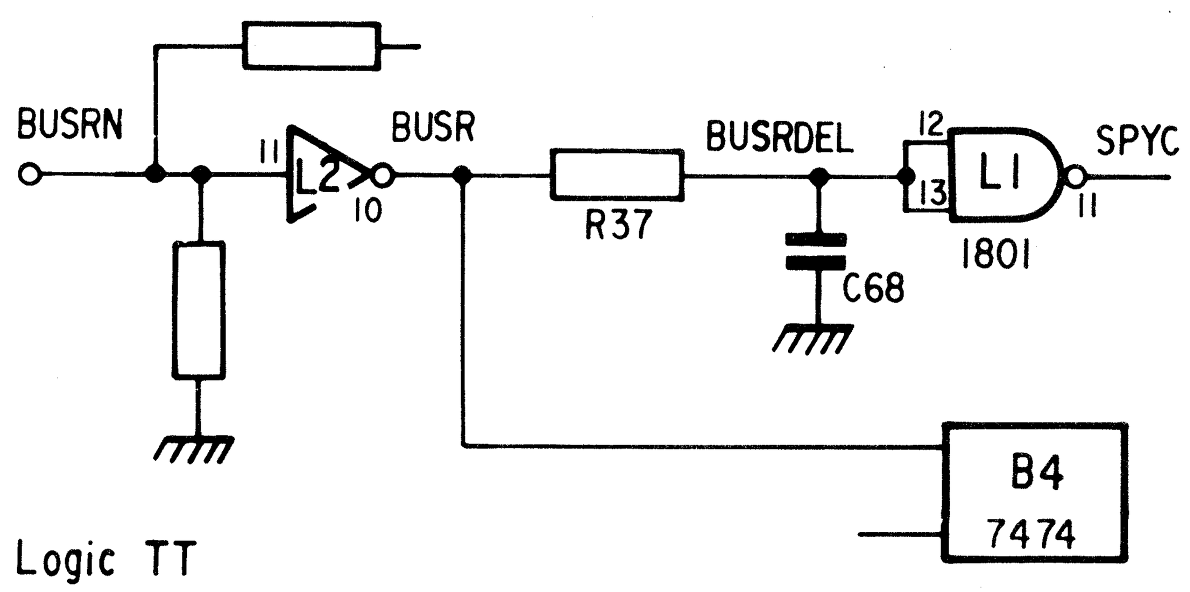
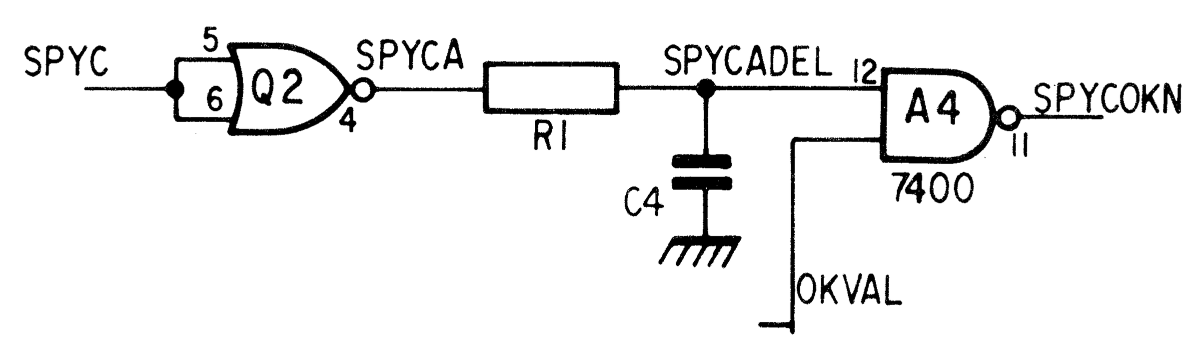
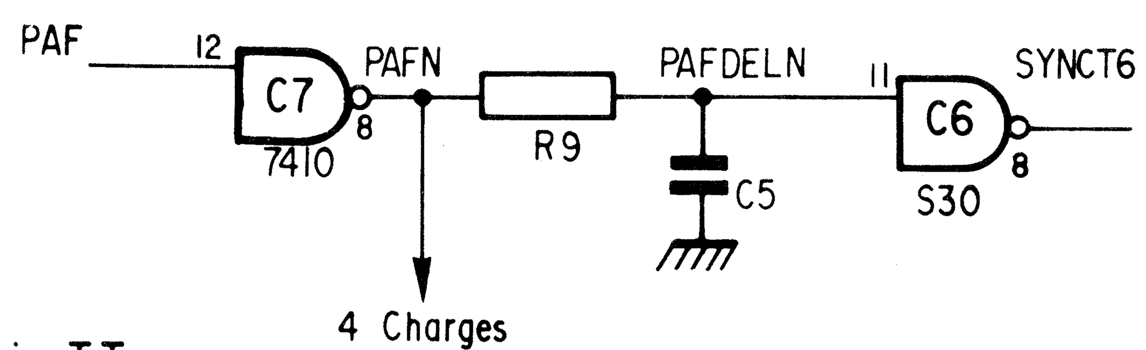
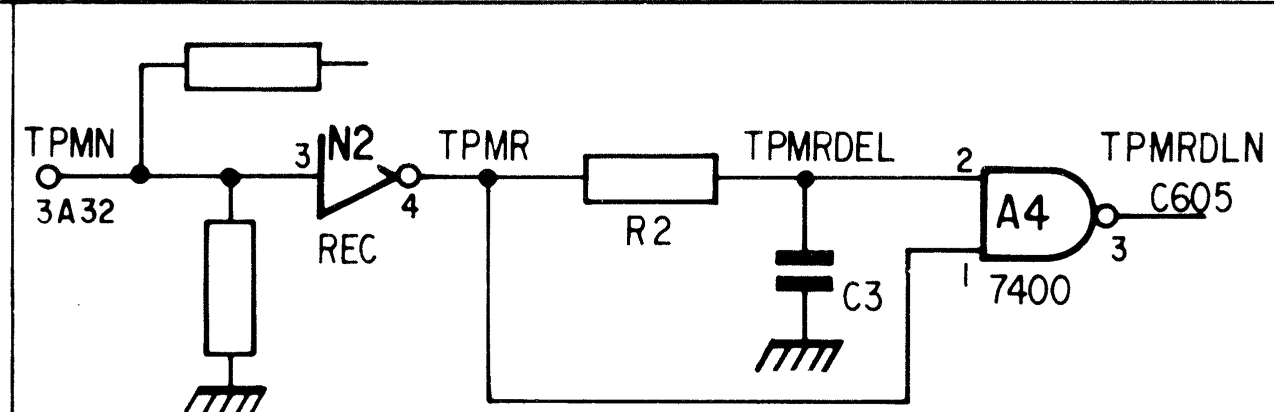


Figure 2-8SS Interrupt Logic



NOTE:
* BUS IN/OUT on this sheet
† BUS IN/OUT (Input on sheet NN)
ⓐⓑⓒⓔⓕ Delay circuit on sheet UU

Figure 2-8TT Bus Controller

SIGNAL	LOGIC SHCEMATIC	DELAY nS nominal	DELAY nS measured	COMPONENTS
ⓐ DEZO	 Logic TT	95nS	100nS	R = 148Ω 1% 1/8W C = 510 pF ±1% 250V 2222 426 45101 micropoco
ⓑ SPYC	 Logic TT	30nS	40nS	R = 100Ω 1% 1/8W C = 200pF ±1% 500V 2222 427 42001 micropoco
ⓒ SPYCOK	 Logic TT	65nS	70nS	R = 110Ω 1% 1/8W C = 430pF ±1% 250V 2222 426 44301 micropoco
ⓓ PAF	 Logic TT	120nS	120nS	R = 148Ω 1% 1/8W C = 620pF 1% 250V 2222 426 46201 micropoco
ⓔ TPMRDEL	 Logic TT	190nS	200nS 190 nS	R = 110Ω 1% 1/8W C = 1,8nF ±1% 2222 424 41802 micropoco

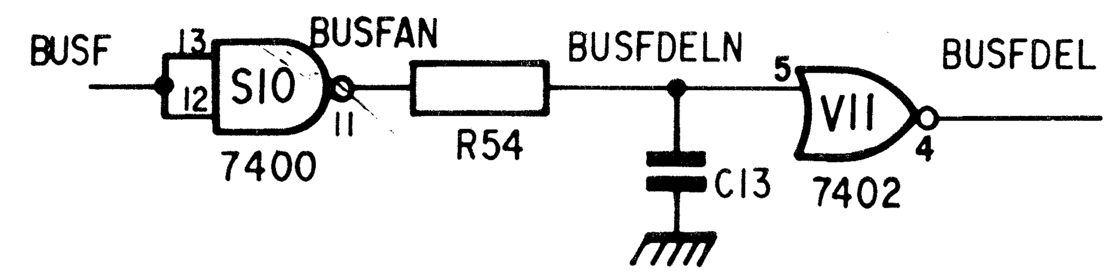
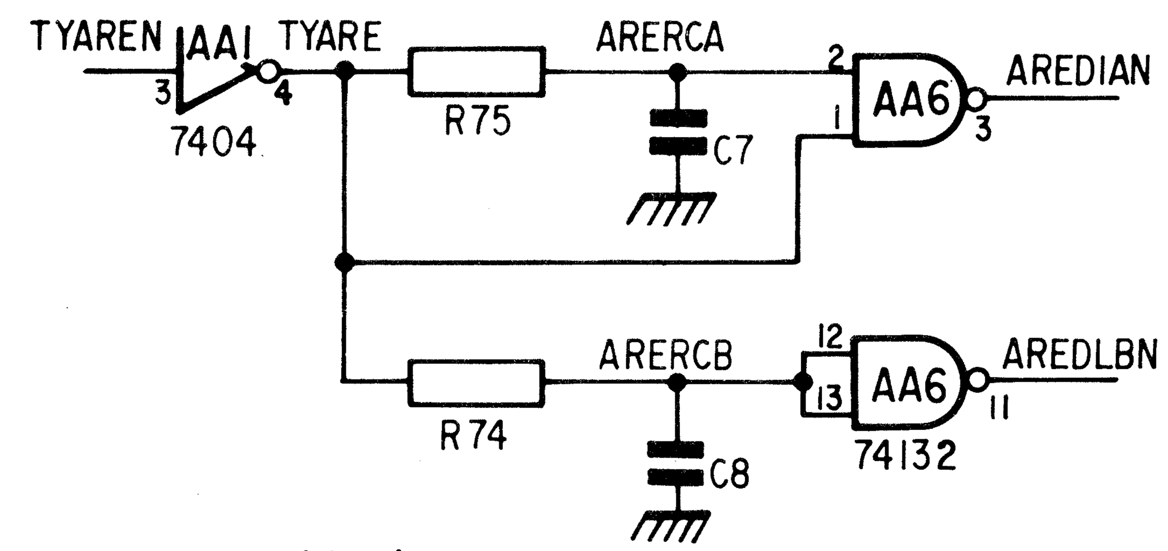
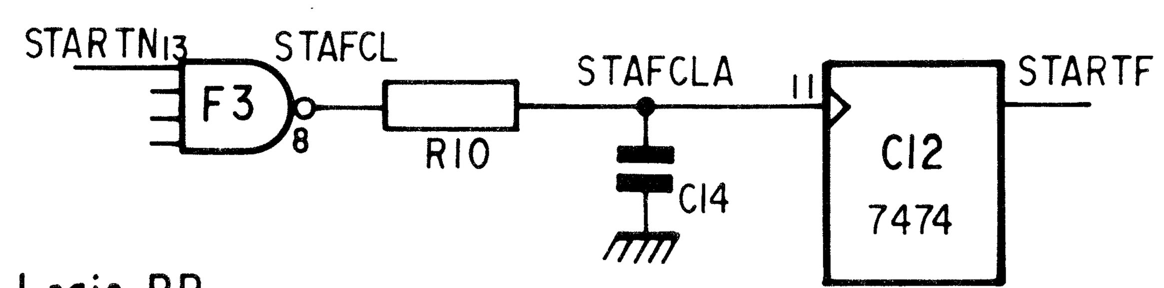
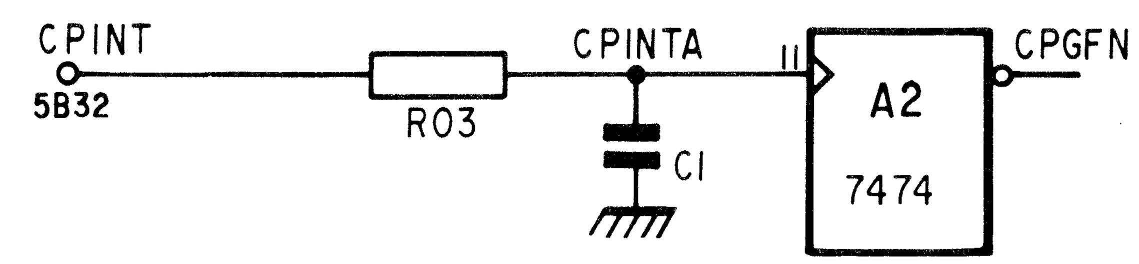
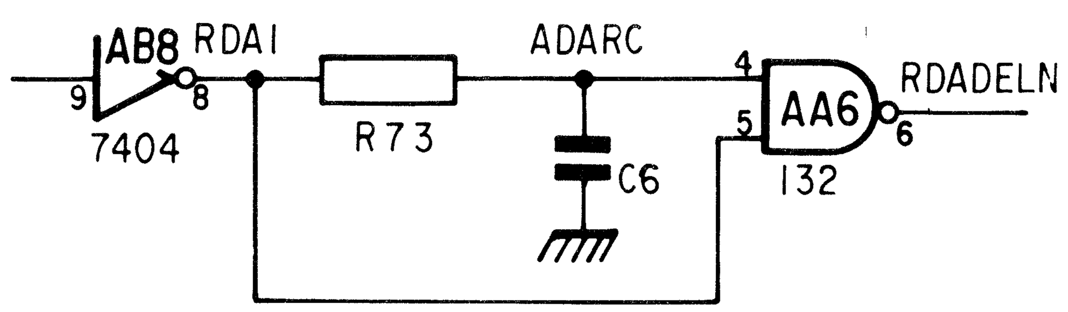
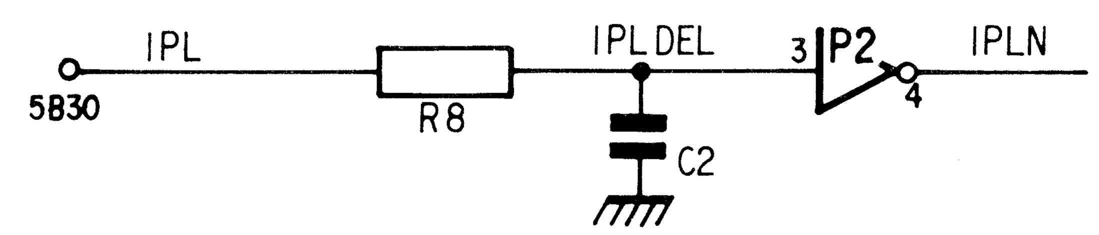
SIGNAL	LOGIC SCHEMATIC	DELAY nS nominal	DELAY nS measured	COMPONENTS
ⓕ BUSFDEL	 Logic TT	260nS	280nS	R = 100Ω 1% 1/8W C = 2nF 1% 2222 424 42002 micropoco
ⓖ DIALB	 Logic 6-4 (V24)	200nS 60nS	240nS 85nS	R = 100Ω 1% 1/8W C = 1,3nF 1% 63V 2222 424 41302 micropoco R = 100Ω 1% 1/8W C = 390pF 1% 250V 2222 426 43901 micropoco
ⓗ STAFCLA	 Logic RR	65nS	75nS	R = 110Ω 1% 1/8W C = 620pF 1% 250V 2222 426 46201 micropoco
ⓚ CPGF	 Logic PP	65	75nS	R = 110Ω 1% 1/8W C = 620pF 1% 250V 2222 426 46201
ⓜ CHIPA	 Logic 6-4 (V24)	200nS	240nS	R = 110Ω 1% 1/8W C = 1,3nF 1% 63V 2222 424 41302 micropoco
ⓝ IPLDEL	 Logic RR	65nS		R = 110Ω 1% 1/8W C = 430pF 1% 25W

Figure 2-8UU Logic Delay Circuit Details

SECTION III

OPERATION AND TESTING

3.1 This section contains all control panel information and system testing information. The testing is divided into two parts : automatic testing via the control panel with the built-in hardware microdiagnostic, and test programs.

3.2 CONTROL PANEL

The P857M is provided with the Extended Control Panel (Figure 3-1). The top half of the panel contains ADDRESS controls; the bottom half, which is identical to the P852M/856M control panels, contains DATA controls and the CPU operating controls. When the computer is running, the BIO lines are displayed on the DATA lamps and the MAD lines are displayed on the ADDRESS lamps. When the computer stops, both the address and the contents of the next instruction are displayed. The ADDRESS half of the Extended Control Panel provides debugging facilities with the following functions :

- Display or load memory in the whole range of addresses (from 0 to 128K).
- Halt on a preset memory address.

All controls and indicators on the panel are described in Table 3-1.

3.3 The P856M is provided with the Standard Control Panel (Figure 3-1). This panel is identical to the P852M panel, with the exception of the TEST position for the key switch. The TEST function operates the microdiagnostic test routines. The P856M Standard Control Panel lacks the ADDRESS controls and the halt-on-preset-address feature.

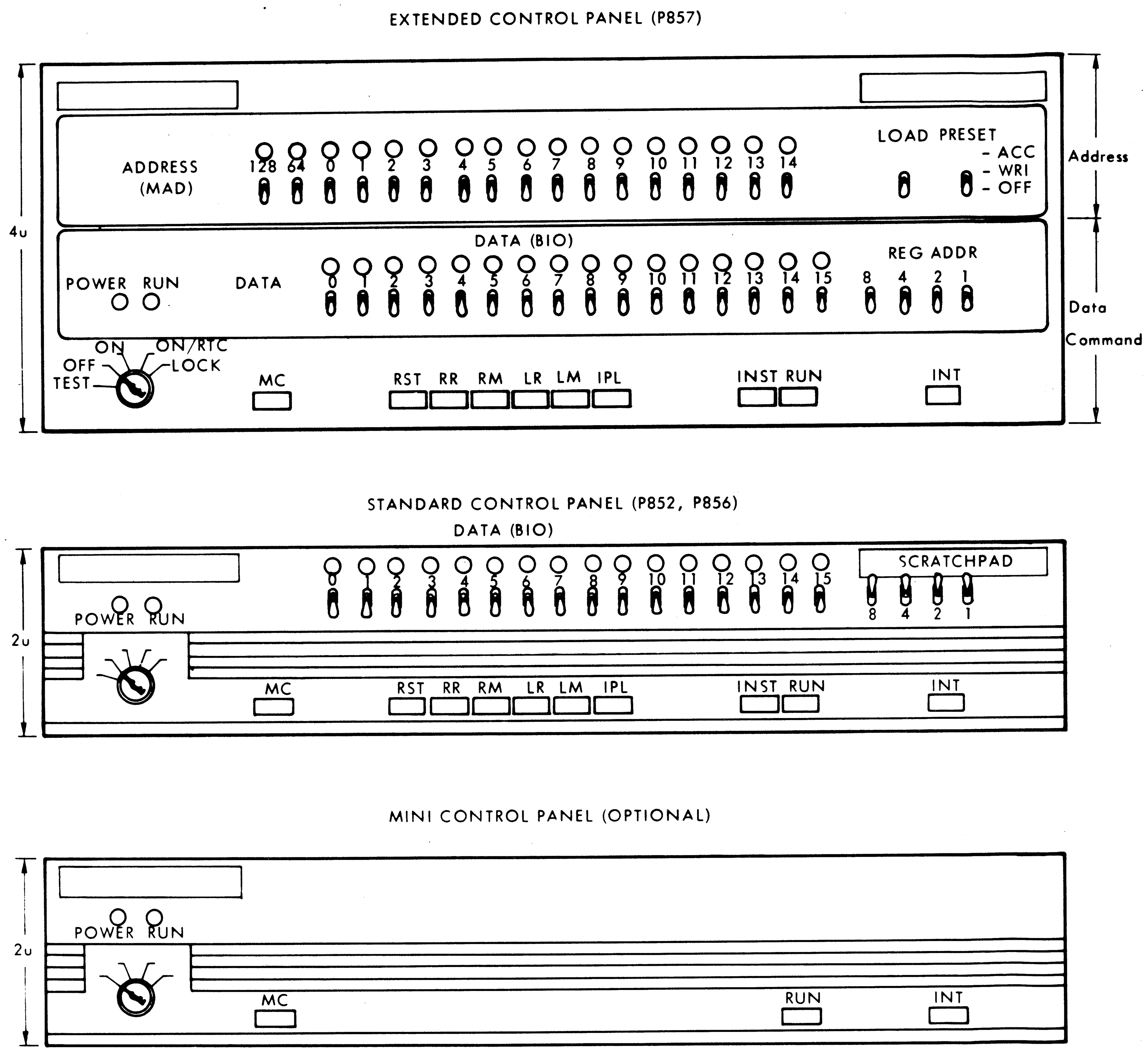


Figure 3-1 Control Panels

3.4 A mini control panel (Figure 3-1) is provided as an option for either the P856M or P857M systems.

3.5 Key and Command Operations

Control panel operations within the CPU operation sequence is shown in Figure 1-8 (Machine-State Pointer, Control Panel Operations). Each of the control-panel sequences shown on the flow diagram is described briefly in Table 3-1. The system is switched on by setting the key switch to ON, ON RTC, or LOCK. The LOCK position disables all control panel command switches except interrupt, which is still operable with the control panel locked. With the control panel switched ON or ON RTC, the CPU is set to the Run state by pressing RUN or IPL. Pressing any of the command switches MC, INST, RST, RR, RM, LR, LM will reset the Run state. When the key switch is set to TEST, the computer is set to Diagnostic mode, and the command switches then have different functions to those listed in Table 3-1.

3.6 Address Operations (Extended CP only)

The ADDRESS half of the control panel operates through an internal address register with incrementing logic. During normal running (PRESET at OFF), the ADDRESS lamps display the Bus MAD lines. When the computer stops, the ADDRESS lamps display the address of the next instruction, which is in the program counter (P), the address register (S), and on the MAD lines.

3.7 For either a read memory (RM) or load memory (LM) operation (Table 3-1), the ADDRESS lamps display the address placed in the control-panel address register by the panel switches. The address is first loaded by setting up the switches and then pressing LOAD ADDR. Thereafter, each time RM or LM is pressed, the address register is incremented and the ADDRESS lamps display the new address. The CPU program counter (P) is not used or affected by either of these operations.

Table 3-1 Control Panel Switches and Lamps

	Key Switch
OFF/ON	Main power switch connected directly to the power supply. The power is switched on (POWER lamp lighted) for positions ON, ON RTC, LOCK, and TEST.
ON	All panel controls are enabled.
ON RTC	All panel controls are enabled, and the Real Time Clock operates (Paragraph 5.6).
LOCK	All control-panel command switches except INT are disabled.
TEST	The automatic microdiagnostic test mode is selected (paragraph 3.16).
	Command Switches
MC	Master Clear : Clears or resets most hardware logic. Activates the GP Bus signal CLEARN, and the CPU signals MCL, MCLN.
RUN	Begins the program. The RUN switch sends the momentary signal START and the flip-flop signal RUNN to the CPU RUNF logic (paragraph 2.90).
INST	Instruction Step : Each time INST is pressed, the CPU performs the one instruction indicated by the program counter (P) and then halts. INST may be used to step the computer through a program (or part of one) instruction-by-instruction. The INST switch resets the control panel RUNN flip-flop, and sends a 113μsec START signal to the RUNF logic.
RST	Read Status. The contents of the program status word are displayed on the DATA lamps (paragraph 1.47).
RR	Read Register. The contents of the scratchpad register (A0-A15) selected by the SCRATCHPAD switches are displayed on the DATA lamps.
RM	Read Memory. The contents of memory are displayed on the DATA lamps. Consecutive words can be read by repeated pressing of the RM button. Std CP: memory address is selected by the program counter (P), and P is incremented with each RM. Ext CP: memory address is selected by the ADDRESS switches. The panel address register is automatically incremented; the program counter (P) is not used or affected.
LR	Load Register. The word code set on the DATA switches is loaded into the scratchpad register (A0-A15) specified by the SCRATCHPAD switches.
LM	Load Memory. The word code set on the DATA switches is loaded into memory. Consecutive words can be loaded by repeated pressing of LM. Std CP: memory address is selected by the program counter (P), and P is incremented with each LM. Ext CP: memory address is selected by the ADDRESS switches. The panel address register is automatically incremented; the program counter (P) is not used or affected.
IPL	Initial Program Loader. An initial bootstrap program located in a hardware read only memory is loaded into memory word locations 00 to 063_{10} (characters 00 to $7E_{hex}$).
INT	Interrupt. This button generates a level-1 Interrupt Request for the Operator's interrupt. The same interrupt can be set by the I/O console via the integral serial control unit. The interrupt may be used by the operator, for example, to change the running program with information supplied by the operator.
	Data
DATA	The 16 DATA switches are used to set a data word onto the Bus BIO lines during load register (LR) and load memory (LM) operations. For all computer operations, the DATA lamps display the contents of the Bus BIO lines. When a running computer stops, the DATA lamps display the contents of the next instruction. For RR and LR operations, the DATA lamps display the contents of the scratchpad register (A0-A15) selected by

	SCRATCHPAD. For RM and LM operations, the DATA lamps display the contents of the memory address selected by the panel address register (Ext CP) or program counter (Std CP)
	Address Section (Extended CP only)
ADDRESS	The ADDRESS switches are used to select an initial memory address for read memory (RM) and load memory (LM) operations. For all computer operations except RM and LM, the ADDRESS lamps display the contents of the Bus MAD lines, via the panel address register. When a running computer stops, the ADDRESS lamps display the address of the next instruction. For RM and LM operations, the ADDRESS lamps display the contents of the panel address register, which is loaded from the ADDRESS switches and incremented by the RM and LM operations. No control is provided for bit 15 (character selector) because the panel accesses only memory word addresses.
LOAD ADDR	When this button is pressed, the code set on the ADDRESS switches is immediately loaded into the panel address register. This address is incremented by successive RM or LM operations; the address register is reloaded from the MAD lines for any other operation.
PRESET	This switch is used to select a Stop On Address mode. The stop will occur when the MAD-line address, via the panel address register, compares with the code set on the ADDRESS switches. OFF Normal operation. Do not stop on address. ACC Stop On Address, Access. Stop when any memory operation accesses the address set on the ADDRESS switches. WRITE Stop On Address, Write. Stop when any memory operation writes at the location set on the ADDRESS switches.
	Scratchpad Registers
SCRATCHPAD	These four switches select one of the scratchpad registers (A0-A15) to be accessed by the read register (RR) or load register (LR) operation.

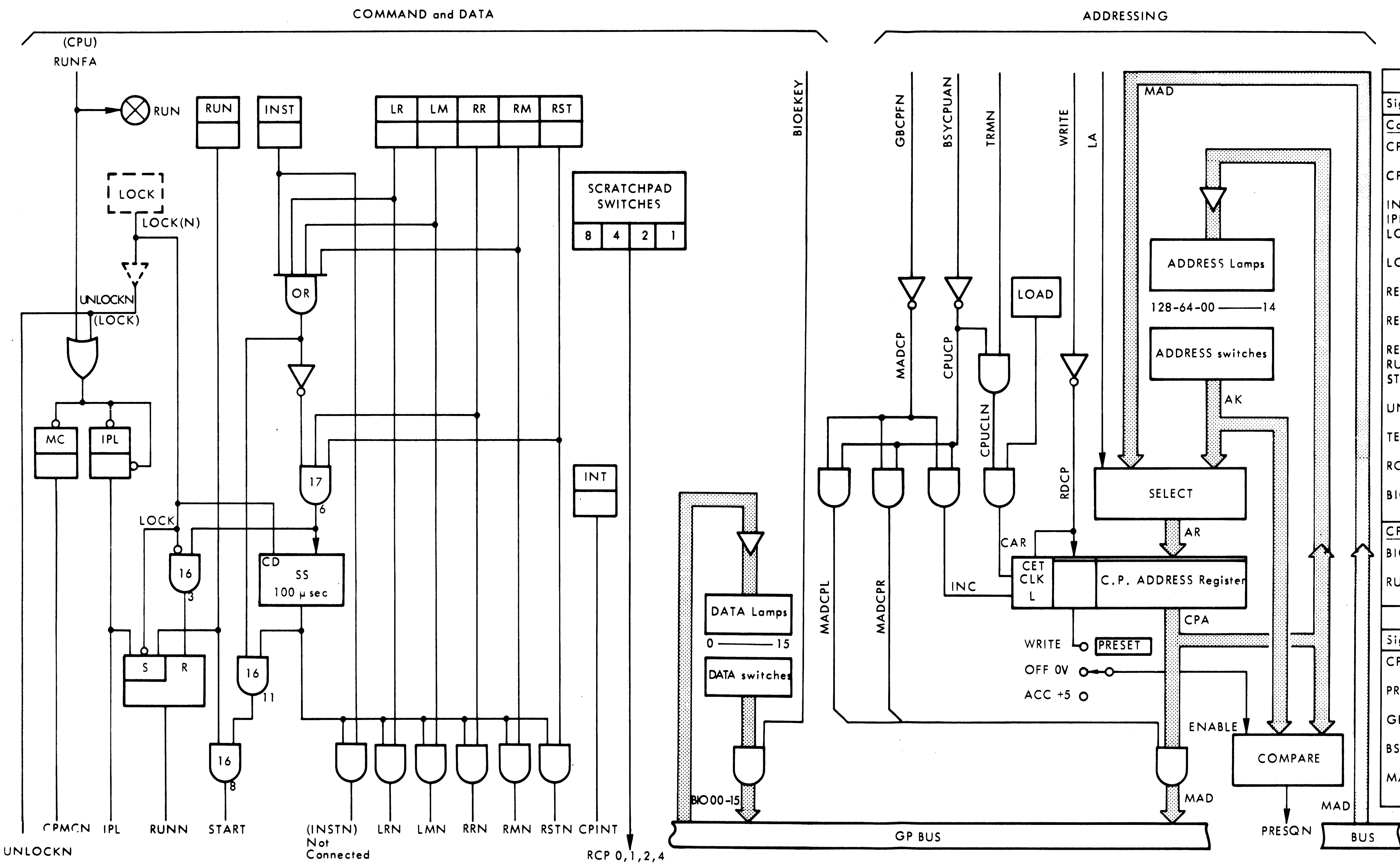
3.8 Halt on preset address is performed by setting the PRESET switch to ACC or to WRITE. The control-panel address register is loaded with the desired address set on the ADDRESS switches and LOAD ADDR pressed. In access mode (PRESET to ACC), the computer will halt whenever the memory address on the MAD lines is the same as the one set on the ADDRESS switches. In write mode (PRESET to WRITE), the computer will halt when the memory address on MAD compares with the ADDRESS switches during a memory write operation.

3.9 Interface Signals

The control panel interface signals are listed in Table 3-2, along with a brief description of each signal. All interface signals are also shown on the control panel block diagram, Figure 3-2. The control panel connector is included in the Wiring and Cabling part of Section III.

3.10 Control Panel Logic

A block diagram and a detailed schematic diagram of the control panel are provided in Figures 3-2 and 3-3.



Note 1 : With the key set to LOCK : the MC and IPL buttons are blocked ; INST, LR, LM, RR, RM, RST are blocked by the SS which is held reset ; and RUN is blocked from activating START by the reset SS.

Note 2 : Run state or Lock inhibit the MC and IPL buttons.

Figure 3-2 Control Panel Block Diagram

Table 3-2 Control Panel Interface

Data/Command Part of Control Panel	
Signal	Purpose, To/From
Control Panel to CPU:	
CPINT	C.Panel Interrupt, sets Operator's Interrupt.
CPMCN	C.Panel Master Clear, causes CLEARN and CPU MCL.
INSTN	Instruction Step.
IPL	Initial Program Loader request.
LOADMN	Load Memory; (DATA) to ADDRESS in memory.
LOADRN	Load Register; (DATA) to SCRATCHPAD register.
READMN	Read Memory; (memory) ADDRESS to DATA.
READRN	Read Register; (register) SCRATCHPAD to DATA.
READSTN	Read Status; (CPU PSW) to DATA.
RUNN	Run mode signal.
START	Start operation, sent by RUN switch.
UNLOCKN	C.Panel unlocked (ON, ONRTC) or locked (LOCK).
TESTN	Key at TEST selects microdiagnostic test mode.
RCPON-3N	Scratchpad (A0-A15) register address code.
BIO00-15	GP Bus data lines (input and output).
CPU to Control Panel:	
BIOEKEY	Gate C.Panel DATA switches onto Bus BIO lines.
RUNFA	Run mode, lights RUN and enables MC and IPL switches.

Address Part of Control Panel	
Signal	Purpose
CPBABS	The B (Address) part of the panel is absent.
PREQN	Preset-Address compare has occurred, and is on MAD lines.
GBCPFN	From CPU: gate ADDRESS to MAD lines.
BSYCPUAN	CPU is master; C.Panel must test the GBCPFN signal.
MAD00-14	GP Bus memory address lines (input and output).

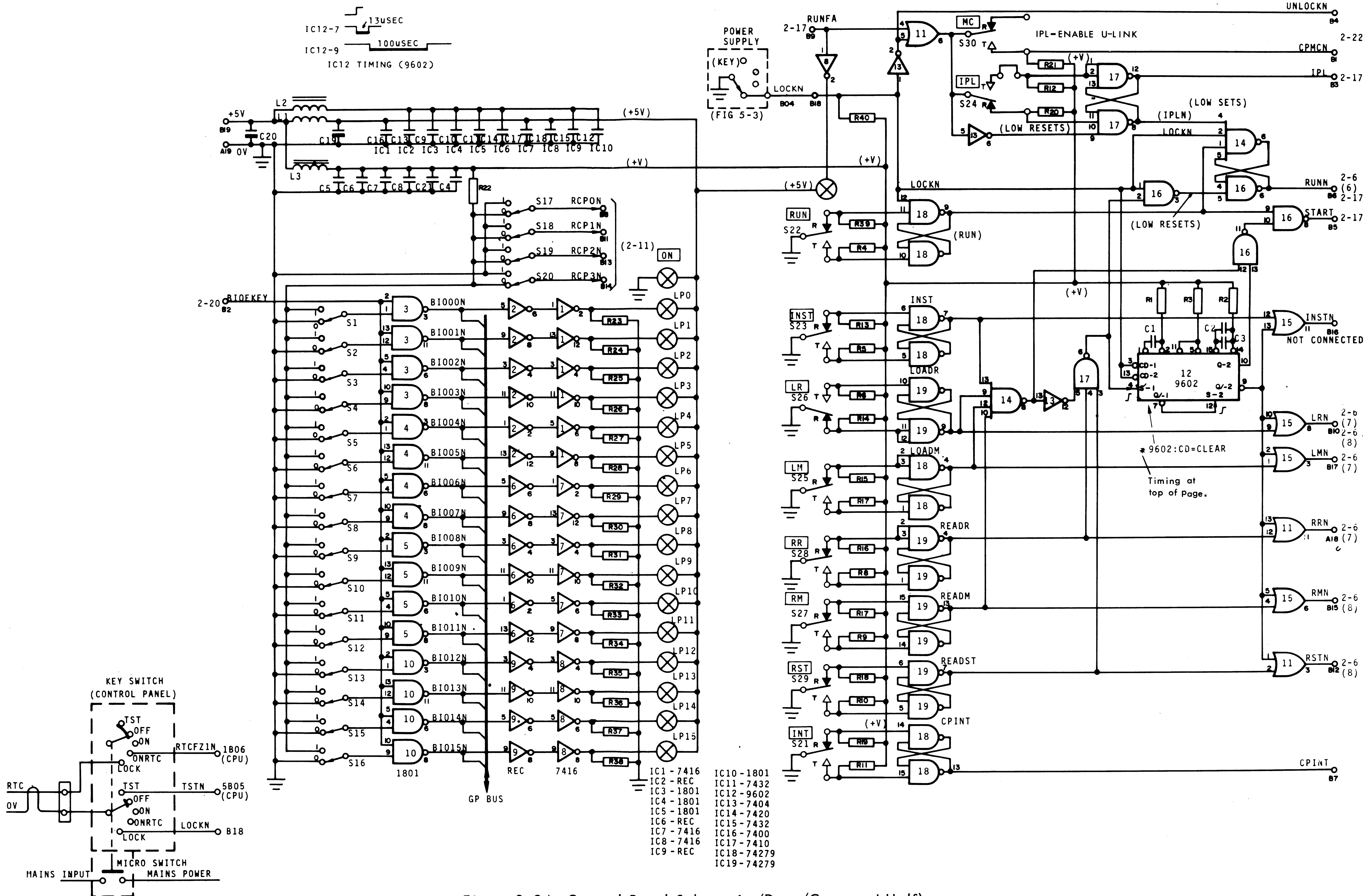


Figure 3-3A Control Panel Schematic (Data/Command Half)

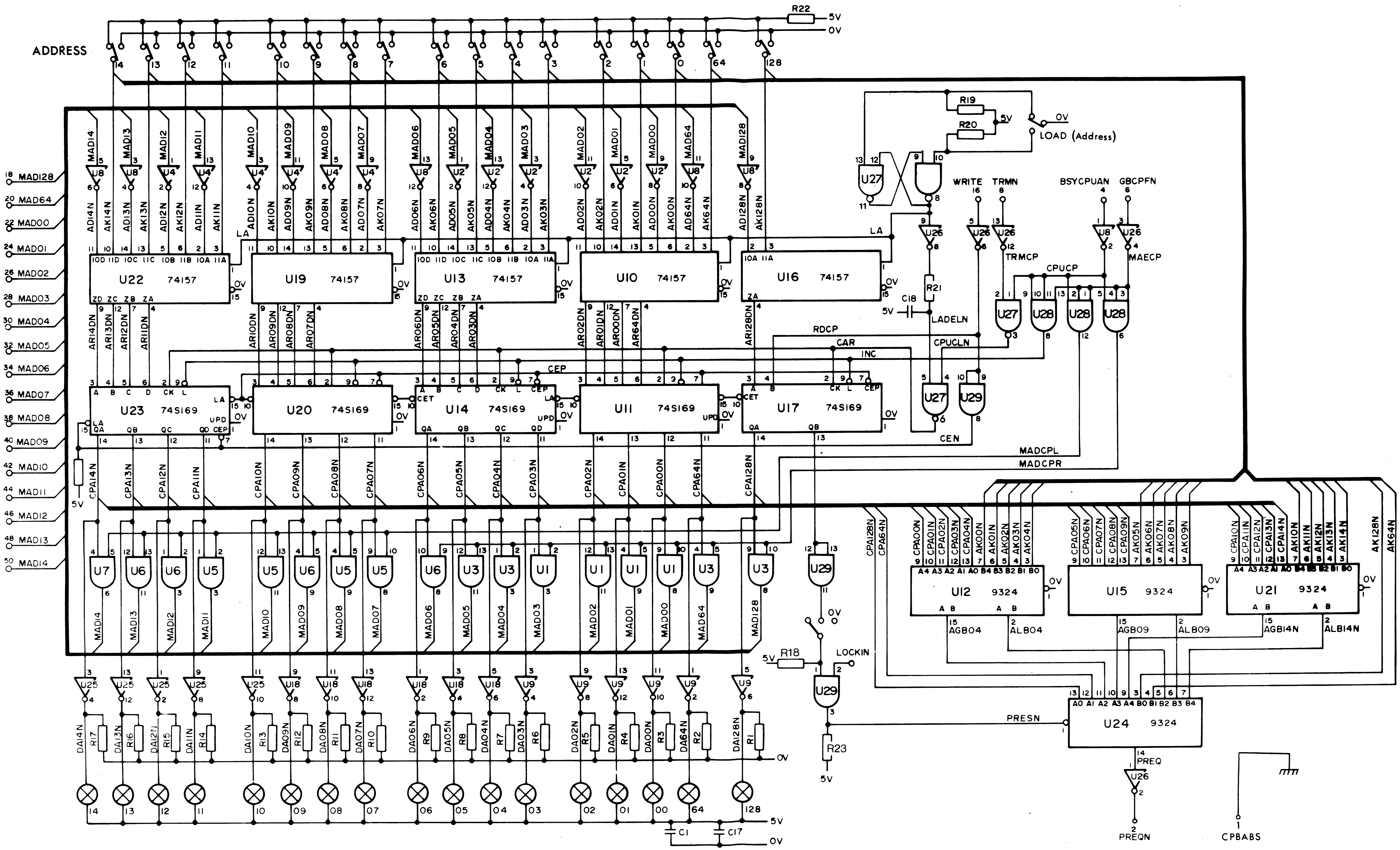


Figure 3-3B Control Panel Schematic (Address Half)

3.11 Key Switch. The Key Switch is shown on the power supply schematic, Figure 5-3. One contact of the switch controls the mains input power to the power supply. A second contact produces the enabling signal RTCFZ1N for the real time clock logic. A third contact of the Key Switch produces the test signal TSTN and the control-panel enable/inhibit signal LOCKN. The test signal is described in the Diagnostic Testing part of Section III. The Lock signal is used to disable all the control-panel command switches except INT. The following table is a review of the Key Switch functions.

Key Switch	Power	Real Time Clock	UNLOCKN Signal	Conditioning for :
OFF	off	--	--	RUN, INST,
ON	on	--	low	LR, LM, RR, MCL,
ON RTC	on	on	low	RM, RST, IPL
LOCK	on	on	h	

3.12 Scratchpad. The four Scratchpad switches are used to address a scratchpad register (A0-A15). The switches produce the true-logic (+5V = logic 1; 0V = logic 0) signals RCP0-4.

3.13 Data. The DATA switches are used to set the 16 bits of a data word to be placed on the GP-Bus BIO lines. The DATA switches are gated onto the BIO lines by CPU signal BIOEKEY. BIOEKEY is produces by BUSFN and BOKFN in the Bus Controller logic (paragraph 2.10).

3.14 INITIAL PROGRAM LOADING

A hardware bootstrap is provided which consists of a 256x4 bit ROM and an IPL-microprogram routine. This hardware routine organizes the bootstrap-ROM contents into 64 sixteen-bit words and loads them into memory locations 00-63. The system then has a bootstrap loaded and is able to accept a software IPL followed by a program. A basic procedure is given here for loading a program from paper tape into the system. This procedure assumes that the tape has been assembled with the software IPL, followed by the program.

Program Load Routine

- Load IPL/Program tape into paper-tape reader.
- Set DATA switches for I/O Bus and paper tape reader :

0 0 0 1	0 0 0 0	0 0 1 0	0 0 0 0

(The codes are shown at the top of Table 2-10, Bootstrap Listing).

- Press INST, MC, IPL
  a. Hardware bootstrap organized by hardware IPL routine and loaded into memory 0-63 (/00 — /7E).
  b. The bootstrap routine is performed to load the software IPL low-core part into memory, starting at /80.
  c. The low-core IPL is performed (with access into the bootstrap) to load the IPL high-core part into memory, at the high end.
  d. The high-core IPL is performed to load the objuct program into memory, starting at /00 and overwriting the bootstrap and the low-core IPL. Interrupt start addresses are loaded at this time in locations /00 — /7E.

3.15 Bootstrap Test

The hardware bootstrap can be loaded into memory for testing (comparing with table 2-10) without being overwritten by the software IPL. This is done by performing the Program Load Routine without a device connected (Paper-tape reader off, disc off, etc.), as follows :

- Set DATA switches for I/O Bus and reader.
- Press INST, MC, IPL.
  a. Hardware bootstrap organized by hardware IPL routine and loaded into memory 00-63.
  b. CPU stops after bootstrap is loaded.
- Load highest memory address into P-register (DATA + reg=0).
- Press INST which increments P to 000.
- Begin pressing RM, and copy bootstrap from DATA lamps.

3.16 MICRODIAGNOSTIC TESTS

Diagnostic test routines are included in the CPU microinstruction control ROM. Approximately 100 words of the CPU control store are allocated for microdiagnostic routines. The complete microdiagnostic tests about 70% of the CPU hardware (Figure 3-4). These microdiagnostics are performed, with the system off line, by setting the control-panel key switch to TEST. The microdiagnostics are arranged sequentually from very basic data-path and register tests to complex instruction operations (Figure 3-5). The different methods of running the tests, in the suggested order, are :

- Automatic Go/No-Go Mode. The instruction-tests DLA and RB are executed and a final display is given. The memory and CPU/CU test is executed and a final display is given. Correct results in this mode indicate correct machine operation.
- Test-By-Test Mode. The operator steps sequentually through logic tests T1, T2, T3, the memory and CPU/CU tests M, and the instruction tests R, and obtains a display indication at each stage. The Chained Tests mode provides further exercise of the same sequence of tests.
- Basic Tests. The operator steps sequentually through basic tests of the control panel, L register, M register, and Q register, exercising all DATA switches and lamps for each test. This checks the basic registers and data paths.

3.17 Test Procedure, Basic Tests

- Key switch to TEST. (Loops through <u>control-panel test</u>.)
- Operate each DATA switch.
- oo Corresponding DATA light should light.
- Press LR. (Loops through <u>L-Register test</u>.)
- Operate each DATA switch.
- oo Corresponding DATA light should light.
- Press LR. (Loops through <u>M-Register test</u>.)
- Operate each DATA switch.
- oo Corresponding DATA light should light.
- Press LR. (Loops through <u>Q-Register test</u>.)

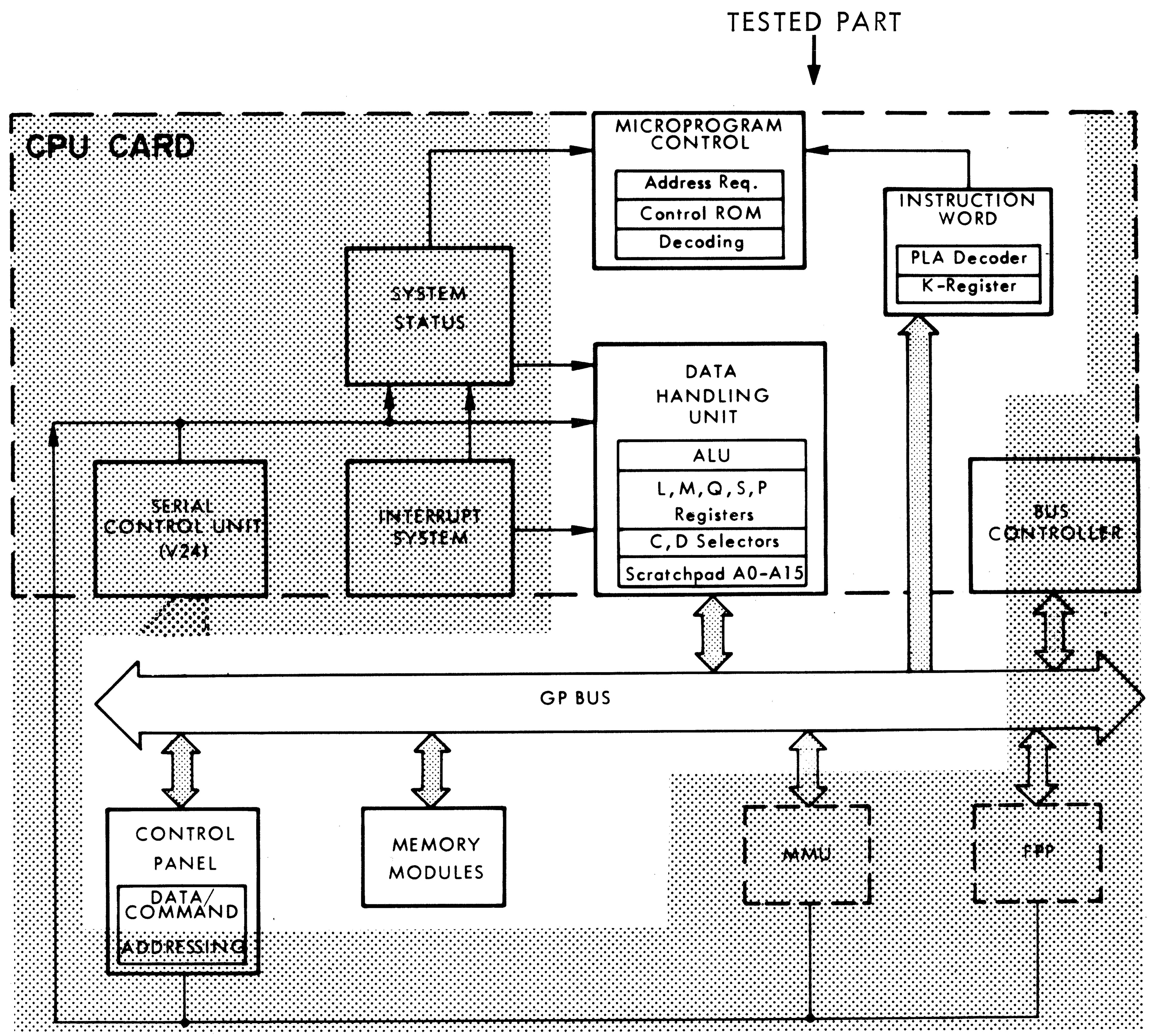


Figure 3-4 CPU Logic Tested by Microdiagnostics

• Operate each DATA switch.

oo Corresponding DATA light should light.

The test is now looping through the Q-Register test. To continue to the next series of tests, set the DATA switches as required (refer to Automatic Mode or Test-by-Test Mode) and press LR or RUN.

3.18 Test Procedure, Automatic Mode (Go/No-Go)

• Key switch to TEST. (Loops through control-panel test.)
• DATA-switch 0 to 0 position. (Enables final display.)
• RUN. (Instruction test performs DLA and RB and then loops at end-of-test display).

oo Wait for display 000000000000●000. If error, do Test-by-Test mode to diagnose fault. If correct, do Memory and CU test.

• Set address of existing CU device on DATA switches 02-07.
• Press LM. (Executes memory tests 1 and 2, and CPU/CU test and then loops at end-of-test display.)

oo Wait for display 0000000000000000. If error, do Test-by-Test mode to diagnose fault.

3.19 Test Procedure, Test-by-Test Mode

• Set address of existing CU device on DATA switches 02-07.
• Key switch to TEST. (Loops through control-panel test.)
• DATA-0 to 0 position. (Enables display at each stage.)
• DATA-15 to 1. (Selects test T1 first.)
• Press LR four times. (Steps through the basic tests, performs logic test T1, and loops at end-of-test display.)

oo Wait for display 000000000000000●.

• Press LR. (Executes test T2, loops at end-of-test display.)

oo Wait for display 00000000000000●0.

• Press LR. (Executes test T3, loops at end-of-test display.)

oo Wait for display 0000000000000●00.

• Press LR or RUN. (Executes instruction test R, loops at end-of-test display.)

oo Wait for display 000000000000●000.

• Press LR or LM. (Executes memory and CPU/CU test M, loops at end-of-test display.)

oo Wait for display 0000000000000000.

Chained tests performes the same tests in the same sequence as above, but without halt on display after each stage.

• Key switch to TEST.
• Address of existing CU device on DATA 02-07.
• DATA-0 to 1 (chained tests, no display loop).
• DATA-15 to 1 (selects test T1 first).
• Press LR four times (steps through basic tests, then loops through the five operating tests without stopping.)
• Set DATA-0 to 0 to stop the chained tests.

oo Wait for one of the five displays.

• Press LR to step through the tests and obtain the display for each.
• Set DATA-0 to 1 to restart the chained-tests loop.

To restart the tests at the beginning, do :

• DATA-0 to 1.
• Key switch away from TEST (OFF is best), then back to TEST.

To Start a test out of sequence (for example, to start test T3 after an error indication for test T2) :

• Key to TEST; CU address on DATA 02-07; DATA-0 to 0.
• Set DATA 08-15 to select desired test :

  00000001 (/01) for logic test T1.
  00000010 (/02) for logic test T2.
  00000100 (/04) for logic test T3.
  00001000 (/08) for instruction test R.
  00010000 (/10) for memory, CU test M.

3.20 Analysing Tested Functions

3.21 Basic Tests. In the control-panel test (Figure 3-6), the operator verifies the operation of all control-panel DATA switches and lamps and the BIO

Note: 0 = lamp on
● = lamp off

data path to and from the GP Bus. In the L-register test, the data path through the CPU D-selector and L-register is added to the control-panel test. The M-register test adds the C-selector, M-register, and ALU to the data-path loop. The Q-register test adds the Q-register to the data-path loop.

3.22 These tests exercise all bit positions of the control-panel switches and lamps and the CPU L, M, and Q registers as well as the data path. Not all the functions of the registers, the C and D selectors, or the ALU are tested.

3.23 Q-Register Test. This test is not only a part of the basic tests, but also the start and the error output of the following logic, memory, and instruction tests. While looping through the Q-register test, the DATA switches content are loaded into the CPU registers to select the type of Test-by-Test operation and the number of the next test. (Test selection and incrementing is shown in Figure 3-7.) If a test has been performed and an error occured, the test branches back to the Q-test loop and the DATA lamps display the setting of the switches rather than the correct display.

3.24 Logic Test T1. The tested functions (Figure 3-8) are :

- Q-reg. shifted left
- A-Bus selection
- Constant Two
- Qo
- ALU=0
- A OR B
- A + B
- $\overline{B}$

3.25 Logic Test T2. The tested functions (Figure 3-8) are :

- Q-reg. shifted right
- Constant Ten
- ALUZERO
- P-register
- P - 2
- A - B
- A + B
- crossed A

3.26 Logic Test T3. The tested functions (Figure 3-8) are :

- A-operand shifted right
- 4 X A
- Reading and writing scratchpad, address incrementation by CT counter (particular addresses A=A2 and A=A15).

3.27 Instruction Tests R. The tested functions (Figure 3-10) are :

- The K-register is loaded with the instruction code for the DLA instruction.
- Scratchpad registers A1 and A2 are loaded with known values.
- Branch to the DLA microprogram and execute DLA instruction.
- Return to Test R via CPU Idle loop and verify A1 and A2. (The CR is verified by the next instruction.)
- The K-register is loaded with the instruction code for the RB instruction.
- Address the RB microprogram with the PLA, and execute RB instruction.
- Return to Test R via CPU Idle loop and verify P-register.
- A failure in CR positioning or in CV (condition verified) circuit, or PLA addressing function should cause wrong value of P counter.

3.28 Memory Test M1. The tested functions (Figure 3-9) are :

- Bit-15 is set to 1 in all addresses of a 16k block (4k for P856).
- The block is read and verified.
- The pattern is shifted left 1 position (to bit 14) and the block is read and verified.
- Shifting the pattern and reading the block is continued until all bit positions have been read and verified.

3.29 Memory Test M2. The tested functions (Figure 3-9) are:

- Every word of a 16k memory block (4k for P856) is written with its own address value.
- Each word is read and verified.

3.30 CPU/CU Dialogue Test M3 (part of Memory-test loop). The tested functions (Figure 3-9) are:

- An existing CU device address (from DATA switches 02-07) and TMP are sent on the GP-Bus MAD lines.
- The CU response TPM is verified by means of condition register (CR $\neq$ 3).

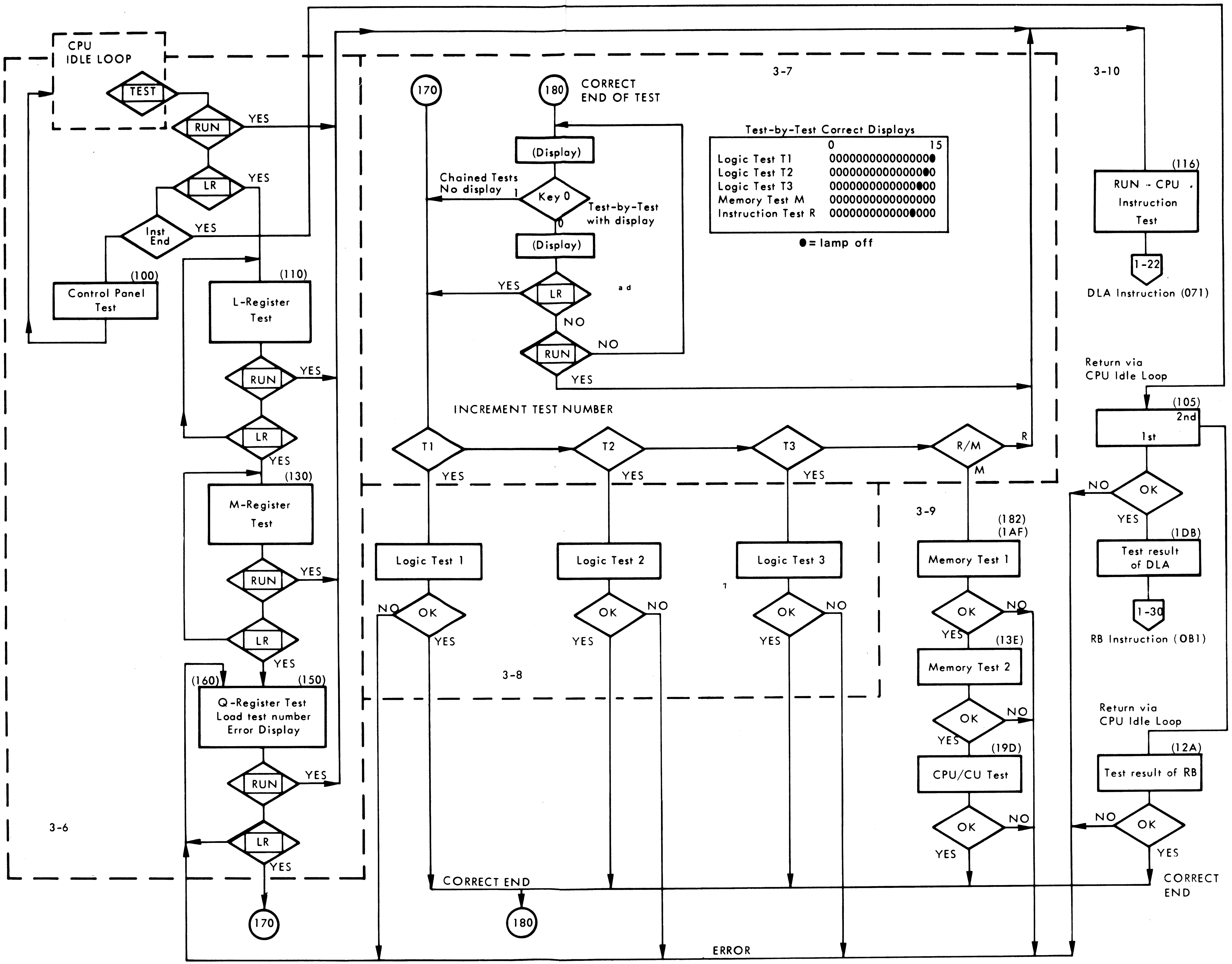
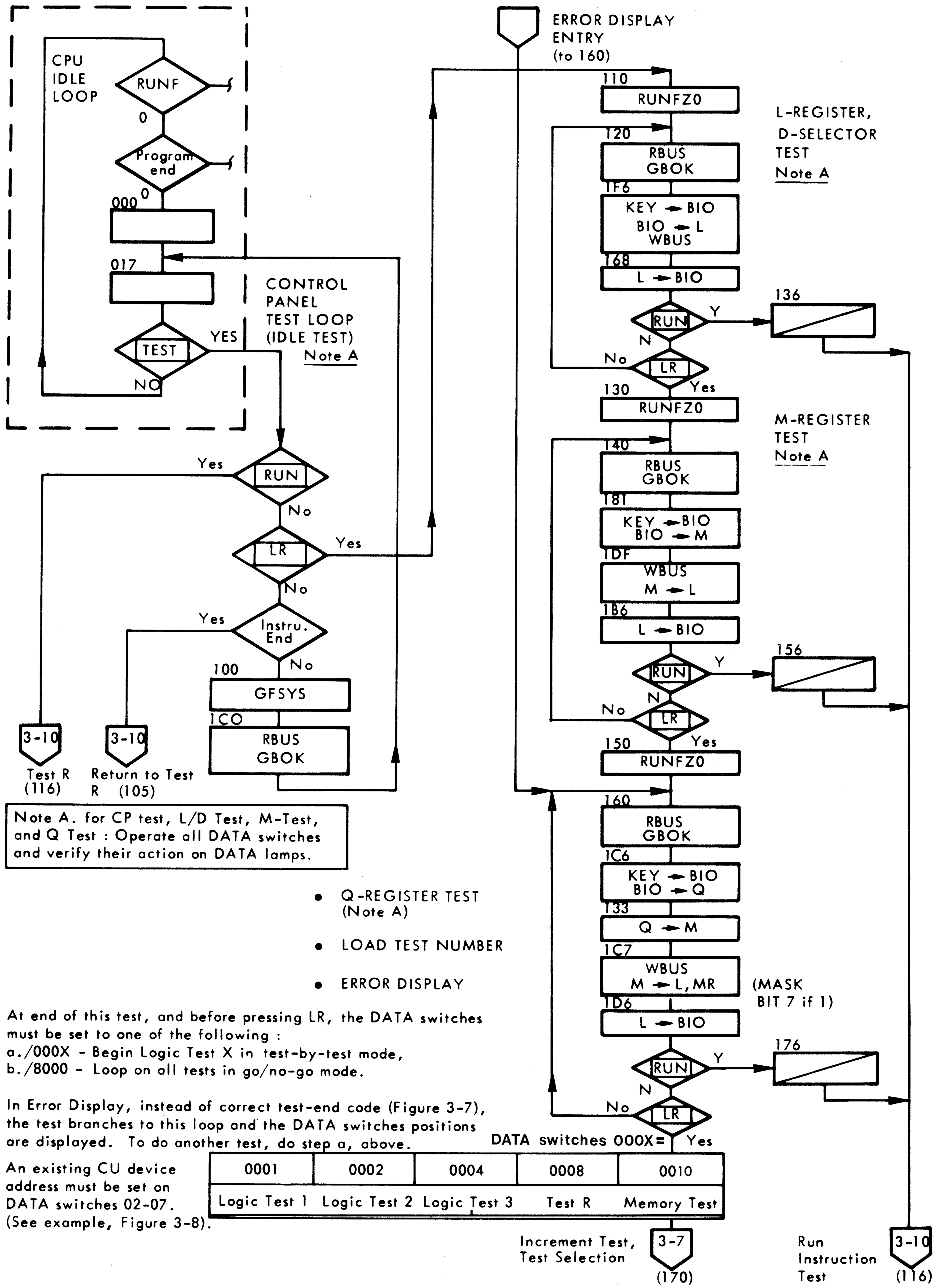


Figure 3-5 Microdiagnostics Block Diagram



0001	0002	0004	0008	0010
Logic Test 1	Logic Test 2	Logic Test 3	Test R	Memory Test

Figure 3-6 Start and Basic Tests

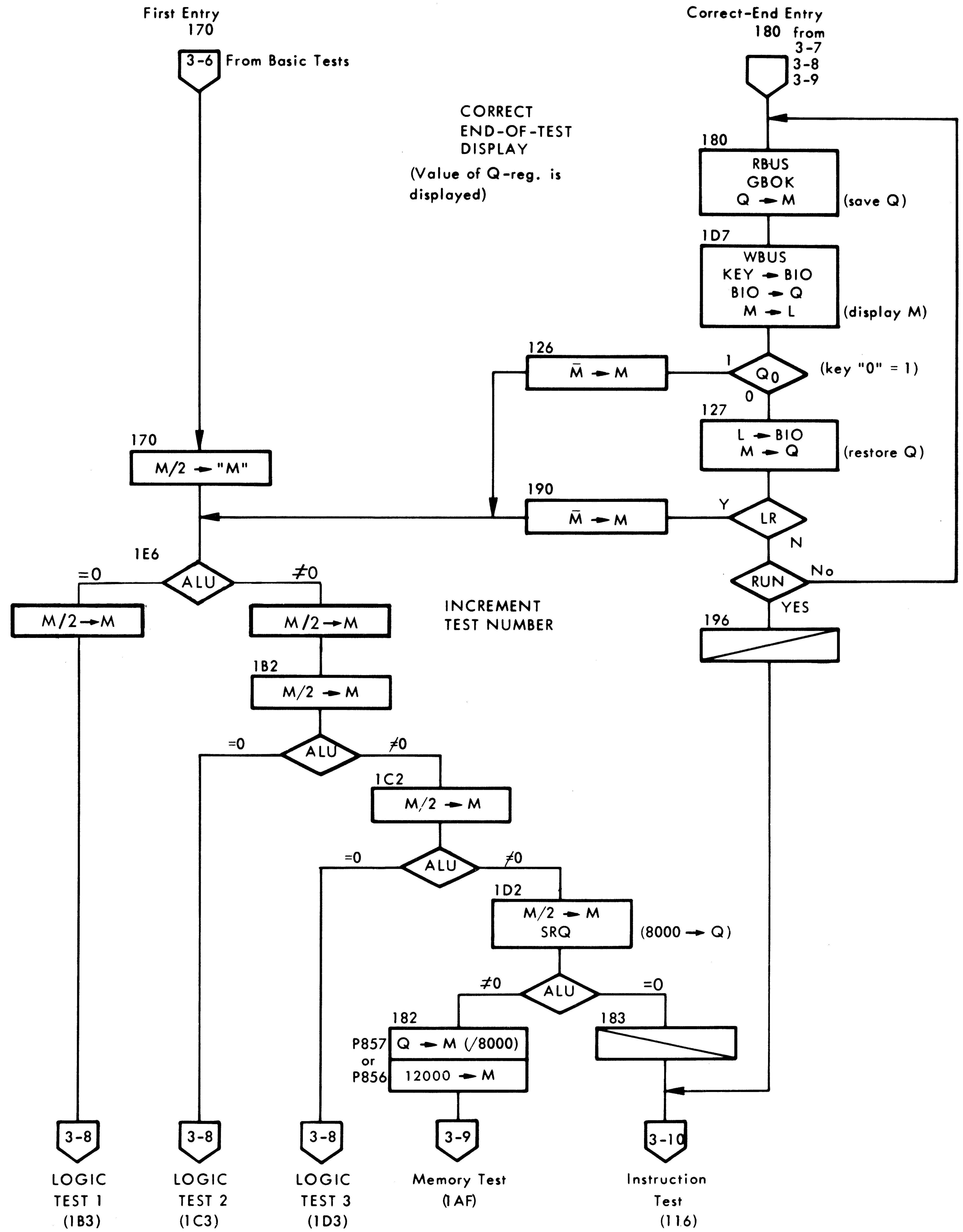


Figure 3-7 Increment Test, Test-End Display

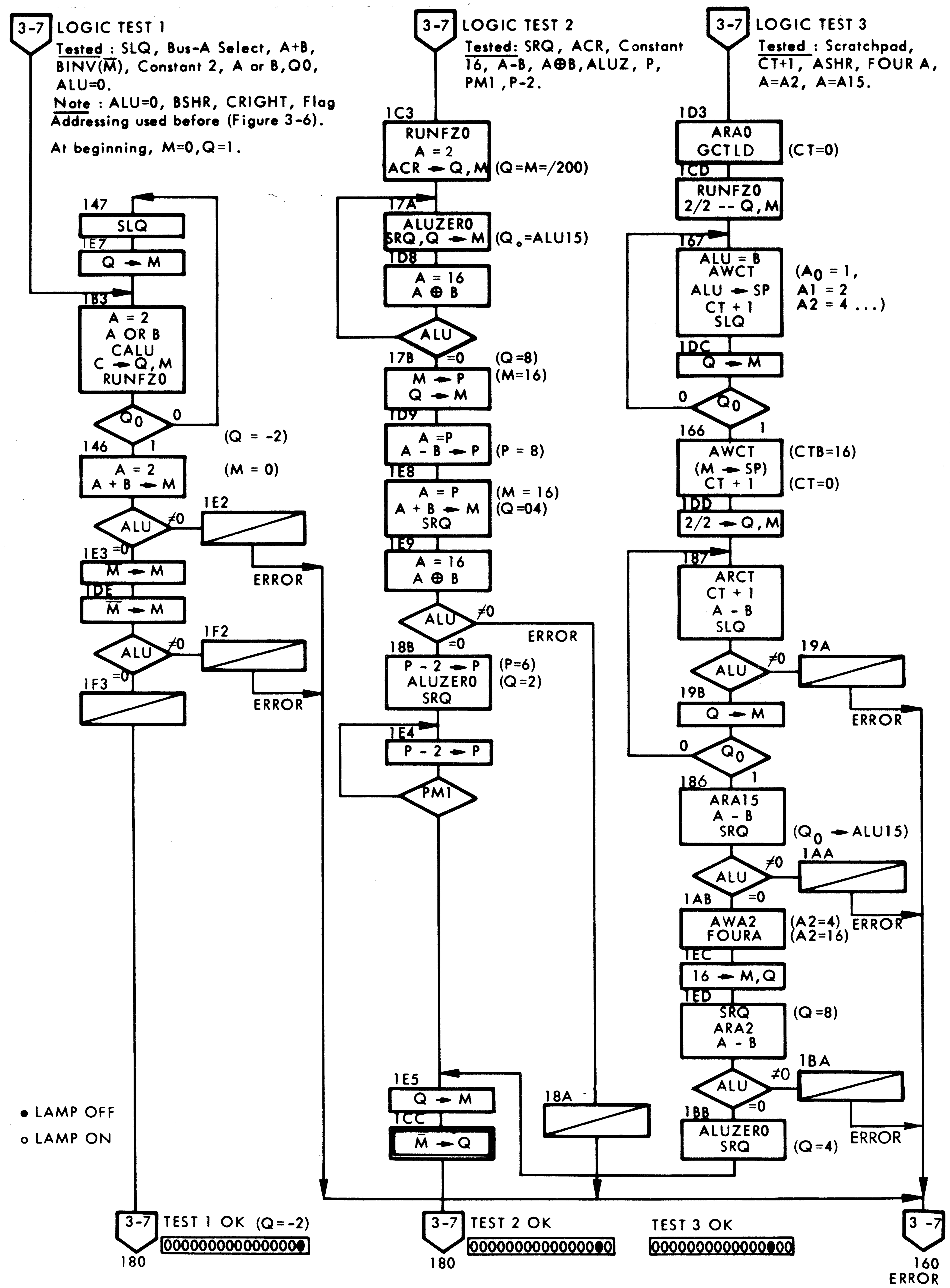


Figure 3-8 Logic Tests (T1,T2,T3)

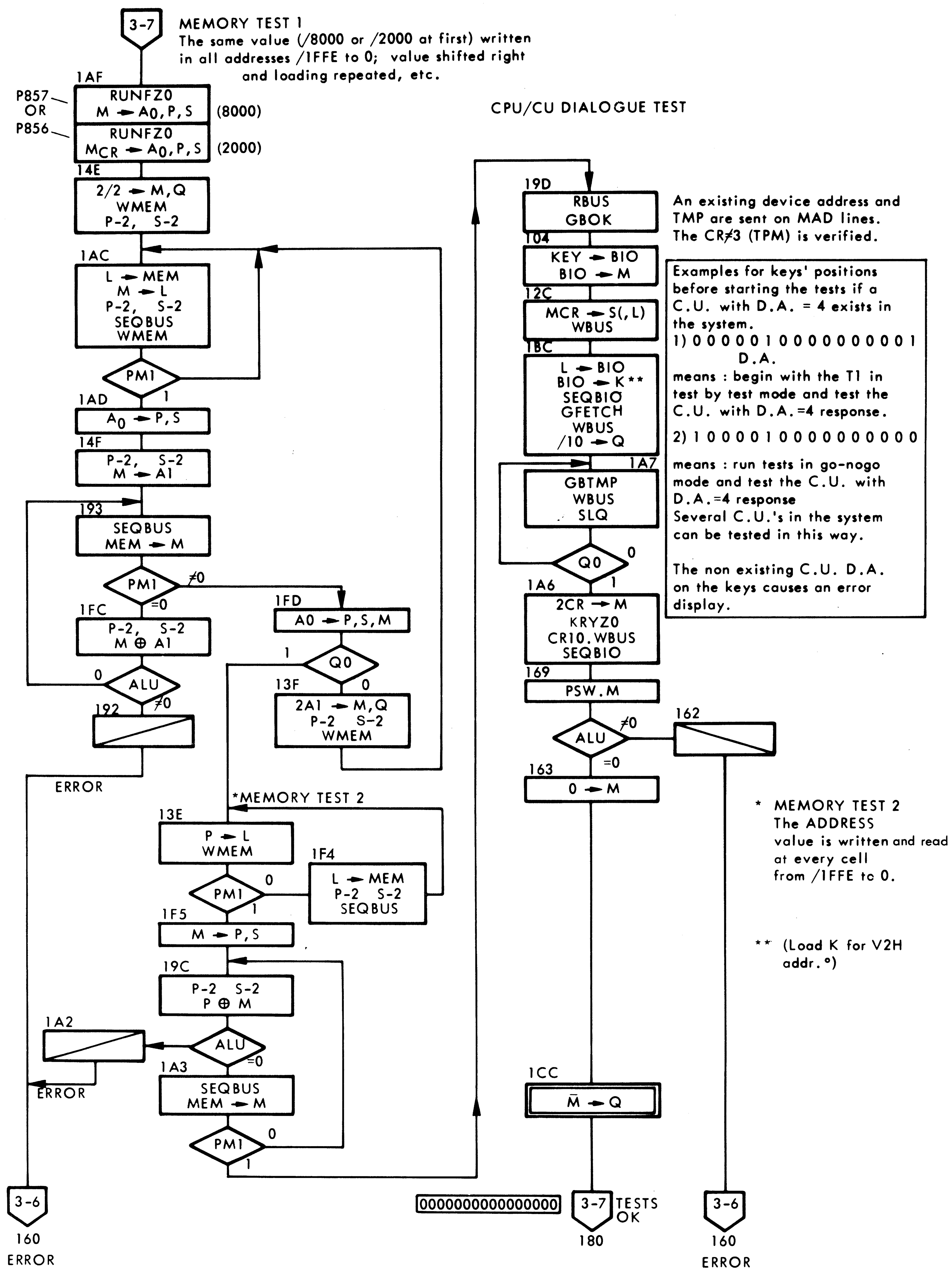


Figure 3-9 Memory and CPU/CU Dialogue Tests (Test M)

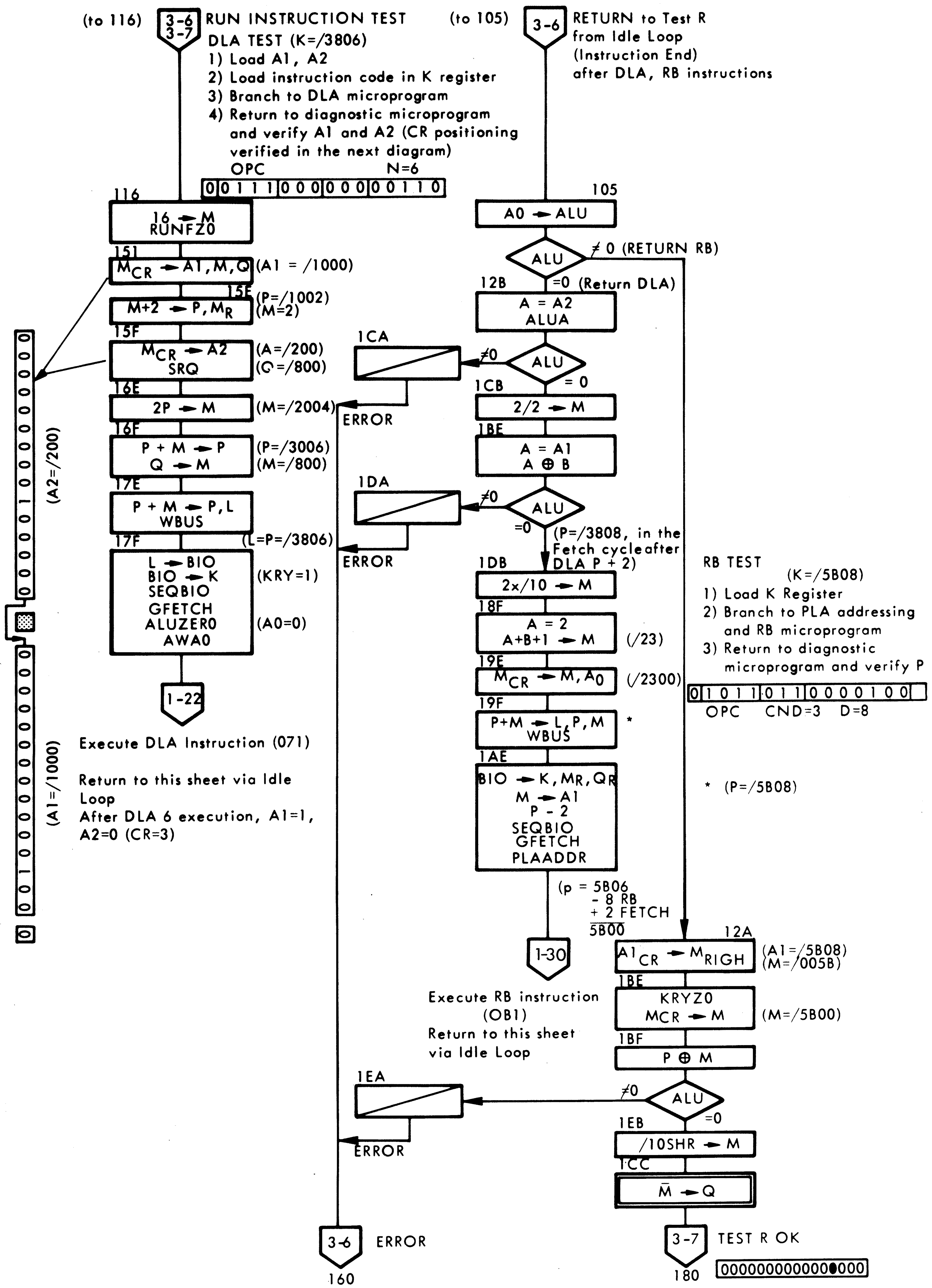


Figure 3-10 Run CPU Instruction Tests (Test R)

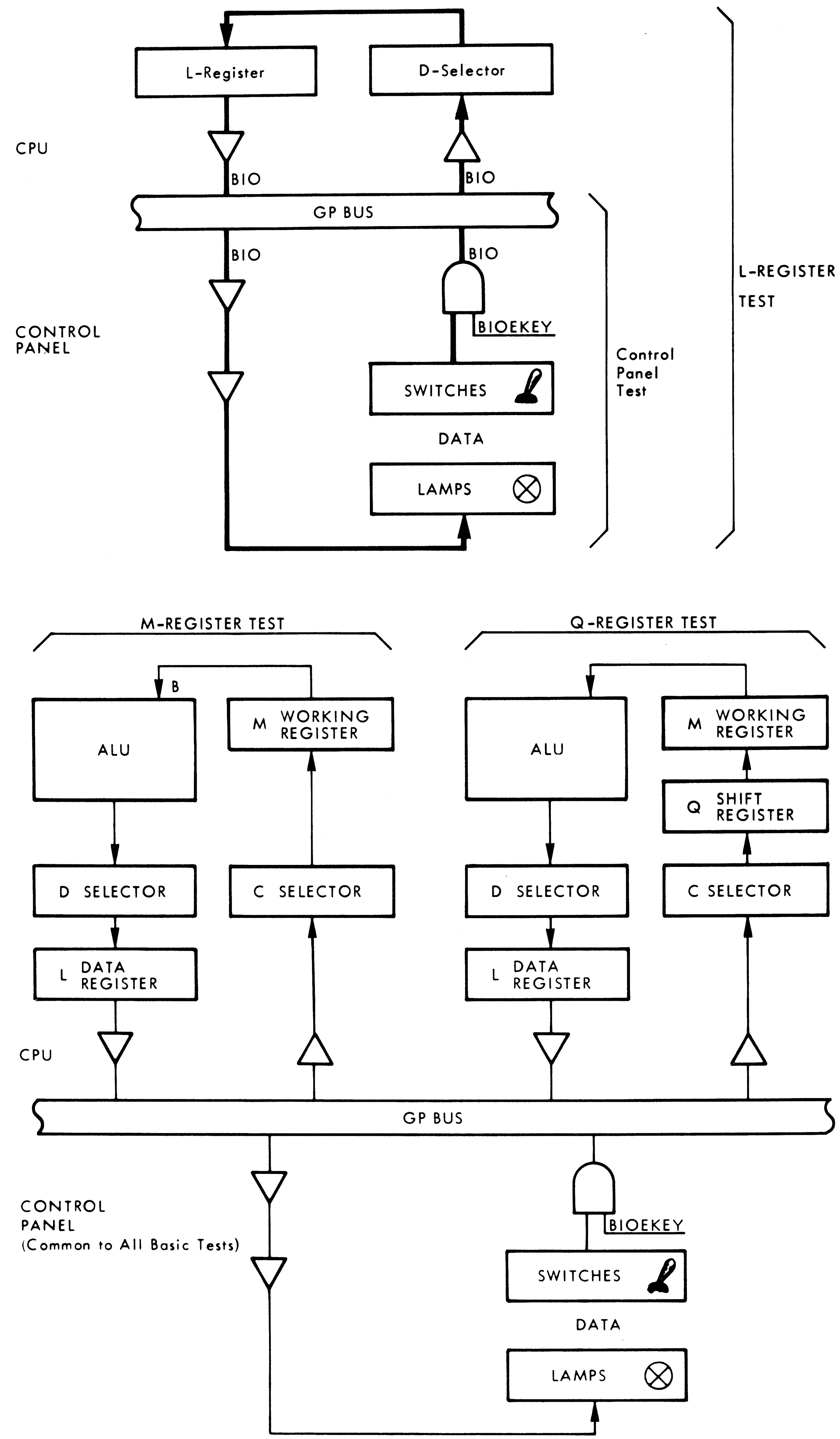


Figure 3-11 Logic Tested by Basic Tests

3.31 TROUBLESHOOTING

Figure 3-12 shows a flowchart of a fault finding procedure covering the most probable areas of a system breakdown assuming that the fault is continuous. When a decision point is reached on the flowchart the engineer must choose the path most appropriate to the fault symptoms. Where the choice is not clear, first take the path that most easily isolates and tests one part of the system and then gradually include other parts of the system until the fault is isolated. For intermittant faults the same procedure can be used but must be repeated each time the symptoms occur (if it is impossible to simulate the fault). It should also be remembered that software not debugged properly can cause faults, which to the programmer appear to be hardware oriented. Therefore, before starting on the hardware, check with the operator if the fault occurs with all programs or with only certain programs, and if these programs are new to the system. If the standard test programs run correctly but the customer's programs do not run correctly, the Software Product Support group should be contacted.

3.32 Microdiagnostics

The microdiagnostic test will enable you to isolate the faulty area in the system and these should be run first before you try any of the other tests suggested in this section. Quite often what appears to be the faulty part of the system may in fact be serviceable, but due to another part of the system the wrong responses are being received. For example, a CU and its device may appear faulty when in fact the cause is operation of the Condition Register.

3.33 Bus Failures

If a GP Bus failure is indicated check the following signals, with no program running, which should all be high if the Bus is operable: RSLN, TMRN, TRMN, TMPN, TPMN, TMEN, MSN, and BSYN.

3.34 RSLN (3B17). This signal is used by the CPU card, the CU cards, and the memories and when it is low it indicates a failure in the power supply. If the power supply is correct, the most probable cause is either the CPU card or one of the memory cards. Disconnect the memory cards one at a time and then the CPU card remembering that the signal RSLN comes from a transistor with an open collector, so for this test connect a 1k ohm resistor between pins 3B17 and 3B19 (+5V).

3.35 TMRN (3A29). This signal is generated by either the CPU card or the IOP and is only used by the IOP or the memories and does not leave the CPU chassis. If this signal is low it indicates a fault in one of the memories, the IOP, or the CPU card. These should be disconnected one at a time from the system until the fault disappears.

3.36 TRMN (3A28). This signal is generated by either the IOP or the memories and is used by these during an exchange with a peripheral. If this signal is low it indicates a fault in one of the memories, the IOP, or the CPU card. These should be disconnected one at a time from the system until the fault disappears.

3.37 TMPN (3A31). This signal is generated by either the CPU or the IOP and is used by all the controllers on the Bus. If this signal is low it indicates that the fault is in the IOP, the CPU, or one of the CU's. The last one removed contains the faulty logic element.

3.38 TPMN (3A32). This signal is generated by each CU and is used by either the CPU or the IOP. If the signal is low it indicates that the fault is either in one of the CU's or in one of the I/O cables. Isolate the CU's and I/O cables one at a time until the fault disappears. The fault will be found in the last item removed.

3.39 TMEN (3A30). This signal is generated by the CPU to validate the address of the external register of either the IOP or MMU. If the signal is low check the CPU, IOP, MMU and the backpanel wiring.

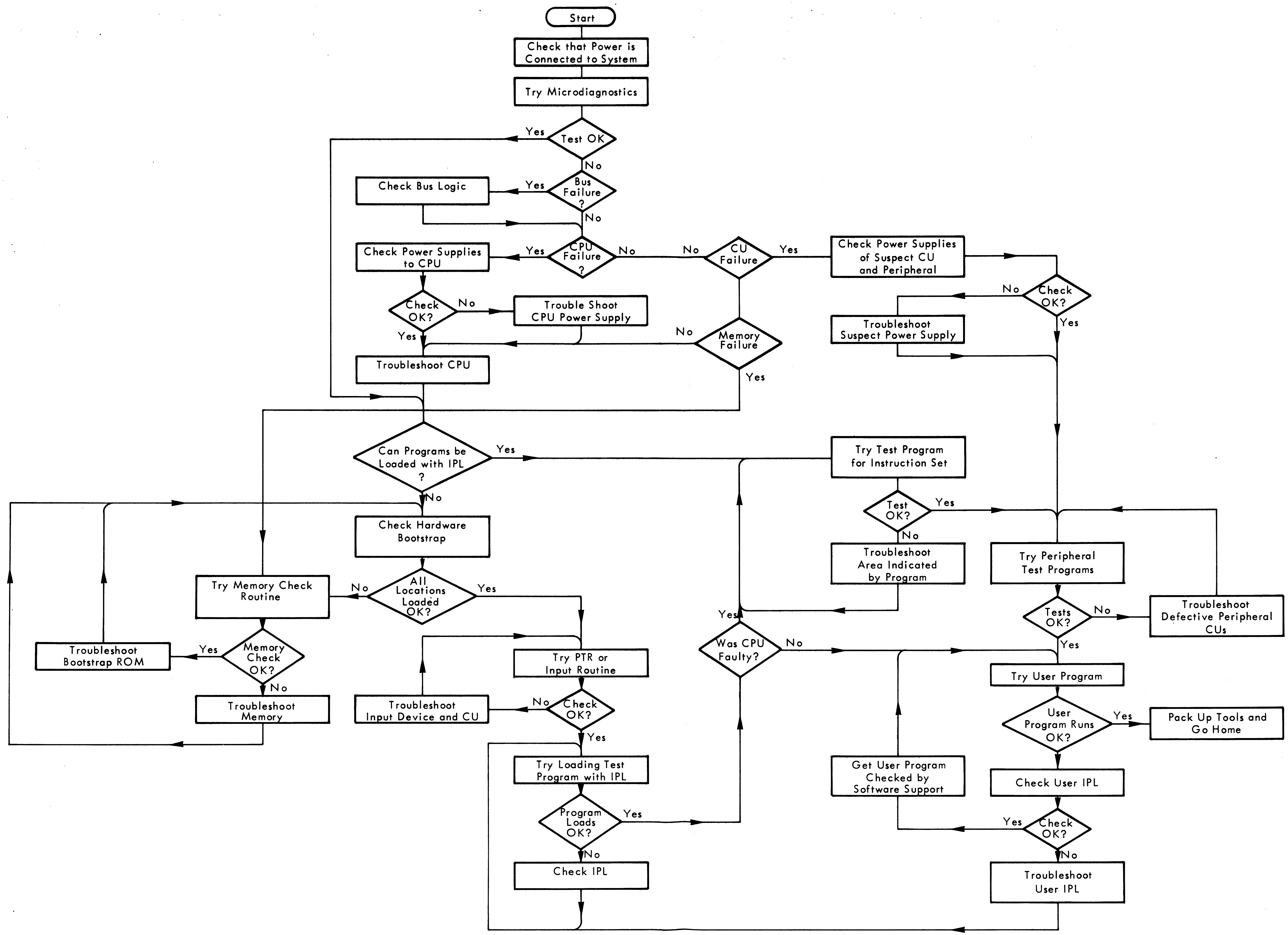


Figure 3-12 Fault Finding Flow Chart

3.40 MSN (3A37). This signal is generated by all the masters connected to the Bus and is used by the masters and the CPU, and it does not leave the CPU chassis. When this signal is low it stops any other master from accessing the Bus and stops the CPU from displaying information on the indicator lamps of the control panel. To find the faulty element disconnect the masters from the Bus one at a time starting with the IOP(s). After these have been eliminated disconnect the masters of the CPU. If the fault still exists check the back panel wiring.

3.41 BSYN (3A38). This signal is generated by all the masters and the CPU, and it does not leave the CPU chassis. When this signal is low it indicates a faulty master and the masters should be isolated from the Bus one at a time until the fault disappears. The fault is either in the last master to be disconnected or, if all the masters have been eliminated, check the back panel wiring.

3.42 Control Panel

In principle, operation of the READ REGISTER and READ STATUS buttons of the control panel is not able to block the Bus. Operation of the LOAD REGISTER, READ MEMORY, and LOAD MEMORY buttons is able to block the Bus. If the control panel is suspect check the signals from the control panel and in particular the signals RUN and START. The IPL operation can also cause the Bus to block especially with an IOP peripheral. In this case check that the signals OKO, OKI, and SPYC are sent correctly; if they are correct the fault will be found in the IOP.

3.43 Bus Blocks Whilst Program is Running

These faults are almost certainly caused by a timing error of one of the Bus signals. Try to find out if the fault occurs with the Programmed Channel peripherals or the IOP Channel peripherals. If the fault is intermittent it will be necessary to try first one channel and then the other using the appropriate test program. In either case check all Bus signals before changing any major components.

3.44 Condition Register Check

If the Condition Register is suspect try the following routines to check that it is being set to the correct state for each arithmetic operation:

- Load any memory address in register A0 (Program Counter).
- Set the hexadecimal pattern 1101 on the DATA switches and push the LM button to load into memory (this loads the instruction ADK A1,1 into the selected memory location).
- Set the hexadecimal pattern 207F on the DATA switches and push the LM button (this loads a Halt Instruction into memory).
- Set the hexadecimal pattern FFFF on the DATA switches, set the 8 4 2 1 switches to select register A1, then push the LR to load the pattern into the register.
- Load the memory address that contains the ADK instruction into register A0.
- Push the RUN button. The routine will execute the instruction and stop at the halt instruction.
- Push the RST button and read the state of the Condition Register (bits 6 and 7). For the pattern above the state should be 00 indication a Zero result in register A1.
- Repeat the routine using the hexadecimal pattern 80 — the result should be 01 indicating a positive Sign bit in register A1.
- Repeat the routine using the hexadecimal pattern FFFE — the result should be 10 indicating a negative Sign bit in register A1.
- Repeat the routine using the hexadecimal pattern 7FFF — the result should be 11 indicating an Overflow condition in register A1.

3.45 If the expected result is not obtained for any of the patterns, repeat that pattern using a Data Probe to check the cause of the fault.

3.46 ALU Data Path Check

Use the following routine to check the data path logic of the ALU:

- Select a memory address on the DATA switches and load it into register A0.
- Set the DATA switches to hexadecimal pattern 3941 and push the LM button. This loads the instruction SLL A1,1 into memory.

- Set the DATA switches to hexadecimal pattern 207F and push the LM button to load the Halt instruction.
- Set the DATA switches to hexadecimal pattern 5F06 and push the LM button to load the RB instruction.
- Set the DATA switches to hexadecimal pattern 0001. Set the 8 4 2 1 switches to address register A1 and push the LR button to load the pattern into the register.
- Load the memory address that contains the SLL instruction into register A0.
- Set the 8 4 2 1 switches to address register A1.
- Push the RUN button. The routine will execute the instruction and stop at the Halt instruction.
- Push the RR button and check that the contents of register A1 have been shifted one place left.
- Alternately push the RUN and RR buttons and check that the contents of register A1 are shifted one place left for each operation of the buttons until the register contains zeros.
- Repeat the routine using an SRL instruction hexadecimal pattern 3961 and test pattern 1000 in register A1.
- Repeat the routine using a DLC instruction hexadecimal pattern 38C1 and test pattern 0001 loaded in register A2. For this test the contents of both register A1 and A2 should be checked for left shifts.
- Repeat the routine using a DRC instruction hexadecimal pattern 38E1 and test pattern 0001 loaded in register A2. For this test the contents of both register A1 and A2 should be checked for right shifts.

3.47 Control Unit Fault

If the microdiagnostics indicate a CU failure check that the CU address on the DATA switches corresponds to the address set up by the U links on that particular CU card. If the two addresses correspond use the following routine and check the state of the MAD lines and timing signals:

- Load any memory address in register A0.
- Set the DATA switches to either of two hexadecimal patterns: 4980 plus the device address for a TST instruction, or 4180 for a CIO Halt instruction (these two instructions are always accepted by the CU irrespective of the sequensor state); then push the LM button to load the instruction in memory.
- Set the DATA switches to hexadecimal pattern 5F04 then push the LM button to load the RB instruction into memory.
- Load the memory address that contains the I/O instruction into register A0.
- Push the RUN button. The routine will execute the instruction repeatedly (because of the RB instruction) until stopped by pushing the INST button.

Whilst the routine is running check the state of the MAD lines and timing signals both on the Backpanel and on the CU card using a Data Probe. If the fault cannot be found on the CU card addressed by the Bootstrap ROM, remove the other CU cards one at a time until the fault disappears then troubleshoot the last card removed. If the fault persists with all other CU cards removed check the printed circuit backpanel and the operation of the CPU Condition Register.

3.48 SOFTWARE IPL

The type of IPL used by the customer will depend on the system configuration, but all IPLs perform a similar function in that they load programs from a chosed device. The three IPL listings given are all used by the Standard Test Programs: the IPL52S is a Stand Alone program loader which prints out the program IDENT, the IPL52H is similar except it has a Halt feature, and the IPL52N is the same as the IPL52S except it does not print out the program IDENT. Only the IPL52S will be described here and like all IPLs it is in two parts, the Low Core IPL and the High Core IPL.

3.49 Low Core IPL

The Bootstrap loads 80 characters then jumps to location hexadecimal 84 which is the start address of the Low Core IPL. The Low Core IPL has three functions:

- It computes the size of memory and subtracts 400 words which gives the first address of the High Core IPL.
- It modifies the contents of some of the Bootstrap instructions then jumps back to Bootstrap (as necessary) to load the High Core IPL.
- When loading is complete it jumps to the start address of the High Core IPL.

3.50 High Core IPL

Once loaded by the Low Core IPL and Bootstrap the High Core IPL loads the Object Program and enables:

- The peripheral, whose address is on the DATA switches (and has been stored in register A15 by the Bootstrap), to be used as the input device.
- The IDENT of the program to be printed out on Operator's peripheral.
- The program to be loaded into its designated place in memory (overwriting the Bootstrap and Low Core IPL).
- The program to be started either automatically or manually.

3.51 Verification of the Low Core IPL

This can only be carried out if loading stops before the message IDENT 'Name' has been typed out on the Operator's peripheral. To verify that loading is taking place (even if the input device is apparently operating, the Bootstrap may not actually be loading the IPL data) try the following routine:

- Stop the CPU by pushing the INST button.
- Load the hexadecimal value 84 into location 84 (the start address of the Low Core IPL) and try to load the IPL again. If the CPU loops on location 84, the Bootstrap is not loading the IPL and the status of peripheral will be found in register A7.
- If loading takes place beyond location 84 but still stops before the Ident is printed, load hexadecimal 80 into register A0.
- Push the INST button and check the contents of memory by comparing the data displayed against the IPL listing. Repeat until each location of the Low Core IPL has been checked or the faulty location found.

3.52 Verification of the High Core IPL

This can only be carried out if the messages EOS, EOF have not been typed out on the Operator's peripheral. If the program loops in the High Core IPL the loading address +2 will be found in register A11, otherwise compute the loading address by subtracting hexadecimal 400 from the memory size, then use the following routine to find where the loading aborts:

- Stop the CPU by pushing the INST button.
- Stop the CPU by pushing the INST button.
- Load the start address of the High Core IPL into register A0.
- Push the INST button and compare the displayed memory contents with the IPL listing.

3.53 If both the Low Core and High Core IPLs and the program appear to be loading correctly but the program does not start, check the contents of register A0 to find out where the CPU has stopped. This check should indicate whether the program has tried to start or if the fault is due to the IPL. Another possible fault that points to a malfunction of the IPL is if the Operator's peripheral or its CU is malfunctioning.

IPL52S

```
                         IDENT   IPL52S
                         ENTRY   IPL88
                         ENTRY   IPL44
                *
                *                            BELIER IPL GENERATION
           000C NBREC    EQU     12
                *
           0000 BIPLPR   EQU     *
           0000 IPL88    EQU     *
     0000  010C          LDK     A1,NBREC
     0002  8220          LDK.L   A2,BIPL-80
     0004  0080  R
     0006  0785          LDK     A7,/85              BASIC WRITE
     0008  80A0          LDKL    A8,DECB
     000A  001E  R
           000C PUNCH    EQU     *
     000C  1250          ADK     A2,80
     000E  8241          ST      A2,DECBUF           SET BUFF ADDR
     0010  0020  R
     0012  2804          LKM                         PUNCH ONE RECORD
     0014  0001          DATA    1
     0016  1901          SUK     A1,1                COUNT DONE ?
     0018  590E          RB(1)   PUNCH               NO
     001A  2804          LKM                         YES
     001C  0003          DATA    3                   EXIT
                *
           001E DECB     EQU     *
     001E  0003          DATA    3                   FILE CODE
     0020       DECBUF   RES     1
     0022  0050          DATA    80
     0024  0000          DATA    0
     0026  0000          DATA    0
     0028  0000          DATA    0
                         EJECT
                         RORG    BIPLPR+/100
                *
           0100 BIPL     EQU     *
                *                            1ST RECORD = LOW CORE IPL
                *                            LOW CORE IPL,1 RECORD,LOADED AT /80 AND
                *                            STARTED AT /84
     0100  FFFF LDFLG    DATA    /FFFF               LDFLG = -1 IF JUST LOADED
     0102  0000          DATA    0                   NOT USED
           0104 IPL      EQU     *
                *                            ADDR OF IPL = BASE ADDR
     0104  82A0          LDKL    A10,/84
     0106  0084
                *                            CHECK IF JUST LOADED OR NOT
     0108  904B          IM      LDFLG-IPL,A10
     010A  FFFC
     010C  311E          RF(1)   IPL100              NO,NOT THE 1ST TIME
```

```
                        *
                        *
                        *                                 YES,COMPUTE HIGH CORE LIMIT
                        *
010E  8720                       LDK.L     A7,/5555            PATTERN
0110  5555
0112  81A0                       LDKL      A9,/FC00            HIGH CORE -/400
0114  FC00
      0116              IPL010   EQU       *
0116  8727                       STR       A7,A9
0118  EF26                       CWR*      A7,A9               IF THE LOCATION (A9) EXISTS,(A7)=((A9))
011A  5006                       RF(0)     IPL020              MATCH
011C  99A0                       SUKL      A9,/2000            NO,NEXT LOWER BLOCK OF 4K
011E  2000
0120  5F0C                       RB        IPL010
      0122              IPL020   EQU       *
0122  8386                       LDR       A11,A9              SAVE BASE ADDR OF HIGH CORE IPL
0124  93A0                       ADKL      A11,2               START ADD.
0126  0002
0128  8486                       LDR       A12,A9              SAVE LOAD ADD.
012A  5704                       RF        IPL110
      012C              IPL100   EQU       *
012C  94A0                       ADKL      A12,80
012E  0050
                        * LOAD ADDRESS OF NEXT RECORD
      0130              IPL110   EQU       *
0130  871F                       LDR       A7,A15              RESTORE CIO INST IN BOOT
0132  273F                       ANK       A7,/3F
0134  9720                       ADK.L     A7,/41C0            CIO INSTRUCTION
0136  41C0
0138  872D                       STR       A7,A3
                        *
013A  0557                       LDK       A5,/57              CHANGE LOC. /5A OF BOOT IN ORDER TO
013C  E541                       SC        A5,/5A              CANCEL LEADING CHARACTER FLAG
013E  005A
                        *                                 RE INITIALIZE A5,A6
0140  0550                       LDK       A5,80               # OF CHARACTERS
0142  8612                       LDR       A6,A12              LOAD ADDR.
                        *                                 CHECK IF COUNT DONE
0144  904B                       IM        CNTFLG-IPL,A10
0146  0048
0148  890E                       ABR(1)    A11
014A  0F42                       AB        /42                 START BOOT ABAIN
      014C              CNTFLG   EQU       *                   - NUMBER OF RECORDS OF HIGH CORE IPL
014C  FFF5                       DATA      1-NBREC
014E  0000                       DATA      0
0150  0000                       DATA      0
0152  0000                       DATA      0
0154  0000                       DATA      0
0156  0000                       DATA      0
0158  0000                       DATA      0
015A  0000                       DATA      0
015C  0000                       DATA      0
015E  0000                       DATA      0
0160  0000                       DATA      0
0162  0000                       DATA      0
0164  0000                       DATA      0
0166  0000                       DATA      0
```

```
00096  0168  0000          DATA    0
00097  016A  0000          DATA    0
00098  016C  0000          DATA    0
00099  016E  0000          DATA    0
00100  0170  0000          DATA    0
00101  0172  0000          DATA    0
00102  0174  0000          DATA    0
00103  0176  0000          DATA    0
00104  0178  0000          DATA    0
00105  017A  0000          DATA    0
00106  017C  0000          DATA    0
00107  017E  0000          DATA    0
00108  0180  0000          DATA    0
00109  0182  0000          DATA    0
00110  0184  0000          DATA    0
00111  0186  0000          DATA    0
00112  0188  0000          DATA    0
00113  018A  0000          DATA    0
00114  018C  0000          DATA    0
00115  018E  0000          DATA    0
00116  0190  0000          DATA    0
00117  0192  0000          DATA    0
00118  0194  0000          DATA    0
00119  0196  0000          DATA    0
00120  0198  0000          DATA    0
00121  019A  0000          DATA    0
00122  019C  0000          DATA    0
00123  019E  0000          DATA    0
00124  01A0  0000          DATA    0
00125  01A2  0000          DATA    0
00126  01A4  0000          DATA    0
00127  01A6  0000          DATA    0
00128  01A8  0000          DATA    0
00129  01AA  0000          DATA    0
00130  01AC  0000          DATA    0
00131  01AE  0000          DATA    0
00132  01B0  0000          DATA    0
00133  01B2  0000          DATA    0
00134  01B4  0000          DATA    0
00135  01B6  0000          DATA    0
00136  01B8  0000          DATA    0
00137  01BA  0000          DATA    0
00138  01BC  0000          DATA    0
00139  01BE  0000          DATA    0
00140  01C0  0000          DATA    0
00141  01C2  0000          DATA    0
00142  01C4  0000          DATA    0
00143  01C6  0000          DATA    0
00144  01C8  0000          DATA    0
00145  01CA  0000          DATA    0
00146  01CC  0000          DATA    0
00147  01CE  0000          DATA    0
00148  01D0  0000          DATA    0
00149  01D2  0000          DATA    0
00150  01D4  0000          DATA    0
00151  01D6  0000          DATA    0
00152  01D8  0000          DATA    0
00153  01DA  0000          DATA    0
```

```
00154  01DC  0000              DATA    0
00155  01DE  0000              DATA    0
00156  01E0  0000              DATA    0
00157  01E2  0000              DATA    0
00158  01E4  0000              DATA    0
00159  01E6  0000              DATA    0
00160  01E8  0000              DATA    0
00161  01EA  0000              DATA    0
00162  01EC  0000              DATA    0
00163  01EE  0000              DATA    0
00164  01F0  0000              DATA    0
00165  01F2  0000              DATA    0
00166  01F4  0000              DATA    0
00167  01F6  0000              DATA    0
00168  01F8  0000              DATA    0
00169  01FA  0000              DATA    0
00170  01FC  0000              DATA    0
00171  01FE  0000              DATA    0
00172  0200  0000              DATA    0
00173  0202  0000              DATA    0
00174  0204  0000              DATA    0
00175                          EJECT
00176                                                            HIGH CORE IPL
00177        0000     PTR      EQU     0                         DEVICE ADDR WILL BE INITIALIZED BY HCIPL
00178        0010     ASR      EQU     /10
00179        0001     S        EQU     1
00180        0000     H        EQU     0
00181        0001     ROM      EQU     1
00182        0000     SA       EQU     0
00183                 *
00184        0000     MODE     EQU     SA                        FOR MESSAGE  OUTPUT
00185                          RORG    BIPL+80                   1ST RECORD OF HIGH CORE IPL
00186        0150     HCIPL1   EQU     *                         1ST WORD OF THE CURRENT RECORD
00187  0150  FEFE              DATA    /FEFE
00188        0152     HCIPL    EQU     *
00189  0152  85A0              LDKL    A13,OUTMSG-HCIPL
       0154  00EA
00190  0156  958E              ADR     A13,A11                   LOAD A13 WITH ADDR OF OUTMSG ROUTINE
00191  0158  868E              LDR     A14,A11
00192  015A  96A0              ADKL    A14,CFZON-HCIPL
       015C  00E8
00193                 *                ADDR OF STACK
00194  015E  871E              LDR     A7,A15                    GET DEVICE ADDR
00195  0160  87CF              ST      A15,SAVA15-HCIPL,A11      SAVE  4*4  FLAG
       0162  0354
00196  0164  5606              RF(6)   *+8
00197  0166  904F              IM      COREND+2-HCIPL,A11
       0168  0358
00198  016A  5C06              RB(4)   *-4
00199  016C  273F              ANK     A7,/3F
00200                 *                          INIT IO INSTRUCTIONS
00201  016E  974F              ADS     A7,CIOPTR-HCIPL,A11
       0170  0142
00202  0172  974F              ADS     A7,INP2-HCIPL,A11
       0174  0188
00203  0176  974F              ADS     A7,END2-HCIPL,A11
       0178  01CF
00204  017A  974F              ADS     A7,END1-HCIPL,A11
```

```
       017C  0208
00205  017E  974F            ADS     A7,INR44-HCIPL,A11
       0180  01A2
00206  0182  811E            LDR     A1,A15
00207  0184  39C8            SLC     A1,8              TEST MULTI-SINGLE DEVICE CONT.
00208  0186  5602            RF(6)   HCIPL2
00209  0188  270F            ANK     A7,/F             MULTI DEVICE CONTROLLER
00210        018A  HCIPL2    EQU     *
00211  018A  974F            AD,S    A7,SSTMLX-HCIPL,A11
       018C  014E
00212  018E  974F            ADS     A7,SSTPR2-HCIPL,A11
       0190  018C
00213  0192  974F            ADS     A7,SSTPTR-HCIPL,A11
       0194  01D0
00214  0196  974F            ADS     A7,SSTPR1-HCIPL,A11
       0198  020A
00215  019A  3F41            SLL     A7,1
00216  019C  974F            AD,S    A7,WER1-HCIPL,A11  INIT WER/RER INST
       019E  013A
00217  01A0  974F            AD,S    A7,WER2-HCIPL,A11
       01A2  013C
00218  01A4  974F            AD,S    A7,RER-HCIPL,A11
       01A6  017C
00219  01A8  57AC            RF      CKOK
00220  01AA  4F42  SYSMSG    DATA    'OBJCT TAPE ON RE'
       01AC  4A43
       01AE  5420
       01B0  5441
       01B2  5045
       01B4  204F
       01B6  4E20
       01B8  5245
00221  01BA  4144            DATA    'ADER, THINK OF B'
       01BC  4552
       01BE  2E20
       01C0  5448
       01C2  494E
       01C4  4B20
       01C6  4F46
       01C8  2042
00222  01CA  4153            DATA    'ASE !'
       01CC  4520
       01CE  2120
00223  01D0  0D0A            DATA    /0D0A
00224        01D2  STAD      EQU     *
00225  01D2  4543  ECMSG     DATA    'EC'
00226  01D4  0D0A            DATA    X'0D0A'
00227  01D6  4F56  OFLMSG    DATA    'OVER'
       01D8  4652
00228  01DA  0D0A            DATA    /0D0A
00229  01DC        BUFF      RES     39
00230  022A  FFFF            DATA    /FFFF
00231  022C  0000  MASTFG    DATA    0
00232  022E  0000  SAVBAS    DATA    0
00233  0230                  RES     6                 * STACK  AREA
00234        023A  CEZON     EQU     *-2
00235        023C  OUTMSG    EQU     *
```

```
023C  0304           LDK      A3,4
023E  4300           CIO      A3,S,ASR
0240  E324           LCR      A3,A1
0242  4310   OTR     OTR      A3,0,ASR
0244  5C04           RB(4)    *-2
0246  1101           ADK      A1,1
0248  1A01           SUK      A2,1
024A  5C0C           RB(4)    OTR-2
024C  0300           LDK      A3,0
024E  4390           CIO      A3,H,ASR
0250  4BD0           SST      A3,ASR
0252  5C04           RB(4)    *-2
0254  F03A           RTN      A14
             *
             *
      0256   CKOK    EQU      *
0256  81A0           LDK.L    A9,0                 SET LOADING ADDRESS
0258  0000
             *
             *
             *                               GO LOAD SYSTEM
025A  84A0           LDK.L    A12,MNLD-HCIPL
025C  029C
025E  948F           ADR      A12,A11
0260  F693           CFR      A14,A12
             *
             *
0262  810A           LDR      A1,A10               TEST START ADDRESS
0264  8C04           ABR(4)   A1                   EOS  OR EOF HAS BEEN READ
0266  207F           HLT                           NO START ADDRESS
0268  0000   BADDR   DATA     0
026A  0300   RAFL    LDK      A3,0
026C  0700           LDK      A7,0
026E  8520           LDK.L    A5,BUFF-HCIPL
0270  008A
0272  950F           ADR      A5,A11
0274  80CE           LD       A8,SAVA15-HCIPL,A11
0276  0354
0278  5610           RF(6)    INPA
027A  0604           LDK      A6,4                 4*4 SO ASR
027C  0111           LDK      A1,/11               SEND X-ON
027E  4600           CIO      A6,S,ASR
0280  4110           OTR      A1,0,ASR
0282  5C04           RB(4)    *-2
0284  4290           CIO      A2,H,ASR
0286  4AD0           SST      A2,ASR
0288  5C04           RB(4)    *-2
      028A   INPA    EQU      *
028A  0650           LDK      A6,80                LENGTH
028C  7600   WER1    WER      A6,0                 INIT MLX WHATEVER
028E  7501   WER2    WER      A5,1                 CHANNEL IS
0290  8602           LDR      A6,A8
0292  3E68           SRL      A6,8
      0294   CIOPTR  EQU      *
0294  46C0           CIO      A6,S,PTR             AVAILABLE ON THE PAPER READER
0296  5C04           RB(4)    *-2
0298  824F           LD       A2,SAVA15-HCIPL,A11
029A  0354
```

```
029C  3AC3                SLC     A2,3            MLX ?
029E  523A                RF(2)   INP2            NO
      02A0      SSTMLX    EQU     *
02A0  49C0                SST     A1,PTR
02A2  5C04                RB(4)   *-2
      02A4      TMTEST    EQU     *
02A4  A120                ANKL    A1,/1007        TAPE MARK OR FATAL ERROR ?
02A6  1007
02A8  501A                RF(0)   MLX1            NO
02AA  2107                ANK     A1,/7           FATAL ERROR ?
02AC  5486                RF(4)   ECCLS           YES
                *
02AE  8120                LDK.L   A1,':E'         PUT EOF IN BUFF
02B0  3A45
02B2  8135                STR     A1,A5
02B4  8120                LDK.L   A1,'OF'
02B6  4F46
02B8  8155                ST      A1,2,A5
02BA  0002
02BC  0704                LDK     A7,4            UPDATA POINTRERS
02BE  951C                ADR     A5,A7
02C0  579F                RF      PRASCI
                *
02C2  5F5A      RAFL3     RB      RAFL            RELAY TOWARDS RAFL
                *
      02C4      MLX1      EQU     *
02C4  E134                LCR     A1,A5
02C6  F920                CWK     A1,/18          IS IT ASCII
02C8  0018
02CA  55FA                RF(5)   RELAY           NO
02CC  0750                LDK     A7,80           YES
02CE  7E00      RER       RER     A6,0            READ REMAINING LENGTH
02D0  A620                ANK.L   A6,/FFF
02D2  0FFF
02D4  9F18                SUR     A7,A6           COMPUTE TRANSMITTED LGT
02D6  951C                ADR     A5,A7           UPDATE BUFF POINTER
02D8  5786                RF      PRASCI
02DA  4A00      INP2      INR     A2,0,PTR
02DC  50DA                RF(0)   SWITCH          INR HAS BEEN ACCEPTED ;
                *                                 PROCESS THE CHARTER
02DE  4AC0      SSTPR2    SST     A2,PTR          TRY A SST
02E0  5C08                RB(4)   INP2            TRY AGAIN INR WHEN SST REFUSED
02E2  8108                LDR     A1,A2
02E4  1300                ADK     A3,0            CHECK WETHER ASCII OR OBJECT
02E6  5278                RF(2)   PRASCI          RECORD WAS ASCII ; PRINT IT
02E8  0600                LDK     A6,0            STORE A NON ASCII CHARACTER AT
02EA  8635                STR     A6,A5           THE END OF BUFFER ,
                *                                 TO BE USED IN MX1
02EC  5F4A                RB      TMTEST
      02EE      OBJINP    EQU     *
02EE  8082                LDR     A8,A8
02F0  560C                RF(6)   EIGHT           8-8 DO NOT INPUT 2ND CHARACTER
02F2  220F                ANK     A2,/F
02F4  4E00      INR44     INR     A6,0,PTR
02F6  5C04                RB(4)   *-2
02F8  260F                ANK     A6,/F
02FA  3A44                SLL     A2,4
02FC  9218                ADR     A2,A6           JOIN THE TWO HALF CHARACTERS
```

```
        02FE   EIGHT    EQU      *
02FE    EF04            CWR      A7,A1
0300    501F            RF(0)    END2
0302    E235            SCR      A2,A5
0304    B408            XRR      A4,A2
0306    1501            ADK      A5,1
0308    EF20            CWK      A7,1
030A    0001
030C    5408            RF(4)    OBJIN1
030E    0400            LDK      A4,0
0310    8108            LDR      A1,A2
0312    9108            ADR      A1,A2
0314    1103            ADK      A1,3
0316    1701   OBJIN1   ADK      A7,1
0318    5F40   INP2R    RB       INP2
031A    2207   FIRST    ANK      A2,/7
031C    0150            LDK      A1,80
031E    5F22            RB(7)    EIGHT
0320    4280   END2     CIO      A2,H,PTR
        0322   SSTPTR   EQU      *
0322    49C0            SST      A1,PTR
0324    5C04            RB(4)    *-2
0326    8082            LDR      A8,A8                  IF ASR WAIT
0328    5606            RF(6)    *+8
032A    904F            IM       COREND+2-HCIPL,A11     WAIT
032C    0358
032E    5C06            RB(4)    *-4
0330    24FF            ANK      A4,/FF
0332    50D4            RF(0)    PROL01+2
        0334   ECCLS    EQU      *
0334    0180            LDK      A1,ECMSG-HCIPL
0336    910E            ADR      A1,A11
0338    0204            LDK      A2,4
033A    84A0            LDK,L    A12,OUTMSG-HCIPL
033C    00FA
033E    948E            ADR      A12,A11
0340    F693            CFR      A14,A12
0342    207F   STOP     HLT
0344    5FDC            RB(7)    RAFL
0346    EA20   ASCINP   CWK      A2,/0D
0348    000D
034A    500E            RF(0)    END1
034C    EF20            CWK      A7,68
034E    0044
0350    5878            RB(0)    INP2
0352    E235            SCR      A2,A5
0354    1501            ADK      A5,1
0356    1701            ADK      A7,1
0358    5F80            RB(7)    INP2
035A    4280   END1     CIO      A2,H,PTR
        035C   SSTPR1   EQU      *
035C    4AC0            SST      A2,PTR
035E    5C04            RB(4)    *-2
        0360   PRASCI   EQU      *
0360    018A            LDK      A1,BUFF-HCIPL
0362    910E            ADR      A1,A11
0364    821C            LDR      A2,A7
0366    0320            LDK      A3,/20
```

```
0368  E335                    SCR     A3,A5
036A  1501                    ADK     A5,1
036C  8320                    LDK.L   A3,/0D0A
036E  0D0A
0370  8335                    STR     A3,A5
0372  1203                    ADK     A2,3
0374  3A61                    SRL     A2,1
0376  3A41                    SLL     A2,1
0378  8082                    LDR     A8,A8                IF ASR DO NOT OUTPUT ASCII
037A  5608                    RF(6)   PRASCA               A SECOND TIME (ALREADY PRINTED WHEN READ
037C  904F                    IM      COREND+2-HCIPL,A11   BUT WAIT ABIT
037E  0358
0380  5C06                    RB(4)   *-4
0382  5702                    RF      PRASCB
      0384          PRASCA    EQU     *
0384  F697                    CFR     A14,A13
      0386          PRASCB    EQU     *
0386  028A                    LDK     A2,BUFF-HCIPL
0388  920E                    ADR     A2,A11
038A  8328                    LDR*    A3,A2
038C  EB20                    CWK     A3,/3A45             * :E
038E  3A45
0390  5CD0                    RB(4)   RAFL3
0392  1202                    ADK     A2,2
0394  8328                    LDR*    A3,A2
0396  EB20                    CWK     A3,/4F46             *OF
0398  4F46
039A  5CDA                    RB(4)   RAFL3
                    *
                    *
                    *                            SET RETURN PARAMETERS
                    *
039C  82CE                    LD      A10,COREND-HCIPL,A11  START ADDRESS IN A10
039E  0356
03A0  84A0                    LDK.L   A12,MNLD-HCIPL
03A2  029C
03A4  948E                    ADR     A12,A11              MNLD ADDRESS  IN A12
03A6  81CE                    LD      A9,SAVBAS-HCIPL,A11  LOADING BASE  IN A9
03A8  00DC
03AA  80CE                    LD      A8,BADDR-HCIPL,A11   ENDING ADDRESS IN A8
03AC  0116
03AE  874E                    LD      A7,MASTFG-HCIPL,A11  MASTER FLAG IN A7
03B0  00DA
                    *
                    *
03B2  F03A                    RTN     A14                  RETURN TO CALLING
                    *
03B4  5FF4          RAFL2     RB      RAFL3                * RELAY TOWARDS RAFL
03B6  5750          RELAY     RF      PROL01+2
                    *
03B8  1300          SWITCH    ADK     A3,0
03BA  59CE                    RB(1)   OBJINP
03BC  227F                    ANK     A2,/7F
03BE  58E6                    RB(0)   INP2
03C0  FA20                    CWK     A2,/7F
03C2  007F
03C4  58EC                    RB(0)   INP2
03C6  1300                    ADK     A3,0
```

```
03C8  5A84              RB(2)   ASCINP
03CA  EA20              CWK     A2,/1F
03CC  001F
03CE  5116              RF(1)   ASCIT
03D0  EA20              CWK     A2,/14
03D2  0014
03D4  510C              RF(1)   OBJEC
03D6  EA20              CWK     A2,8
03D8  0008
03DA  5206              RF(2)   OBJEC
03DC  EA20              CWK     A2,/10
03DE  0010
03E0  5CCA              RB(4)   INP2R
03E2  1301      OBJEC   ADK     A3,1
03E4  5FCC              RB(7)   FIRST
03E6  1801      ASCII   SUK     A3,1
03E8  5FA4              RB(7)   ASCINP
                        EJECT
                *
                *
                *  MAIN  LOADING  PART          ENTRY PARAMETERS :
                *
                *                                  A1 = MASTER FLAG
                *                                  A9 = BASE ADDRESS
                *
                *
                *
                *               1- OUTPUT MSG REQUESTING  TAPE ON READER
                *               2- PERFORMS A 'HALT' - THE USER HAS THEN THE
                *                  POSSIBILITY TO ALTER THE BASE ADDRESS (REG,A9)
                *                                    AND  THE MASTER FLAG  (REG,A1)
                *                                                   (A1=0 MASTER)
                *                                                   (A1=1 USER)
                *               3- LOADING PROCESS STARTS  WHEN USER DEPPRESS START
                *                  BUTTON
                *
                *
                *
                *
      03EA      RAFL1   EQU     *
03EA  5F38              RB      RAFL2
                *
                *
                *                               ERRONNOUS CLUSTER,PRINT ERR MESSAGE
      03EC      CLC01   EQU     *
03EC  5FBA              RB      ECCLS
                *
                *
                *
                *
      03EE      MNLD    EQU     *
03EE  8404              LDR     A4,A1                   SAVE  MASTER FLAG
03F0  0200              LDK     A2,0
03F2  824F              ST      A2,COREND-HCIPL,A11    RA7  START  ADDRESS
03F4  0356
                        IFT     MODE=ROM
                        XIF
```

```
03F6  81CF           ST      A9,SAVBAS-HCIPL,A11
03F8  00DC
                *
                *
                *
                * PROCESS LOADING : THIS  MODULE READ A  CLUSTER
                *          AND  BRANCH ACCORDING TO THE CLUSTER TYPE
                *          ON EXIT   A1 = BUFF  ADDRESS +1
                *                    A2 = WORD  COUNT
                *                    A3 = TYPE
                *          THE TYPE MUST BE    3,4,7   IF NOT THIS :HALT
03FA  82A0      PROLO   LDK.L   A10,STAD-HCIPL        END ADDRESS
03FC  0080
03FE  928F              ADR     A10,A11
      0000      ABA     EQU     0
0400  81CF              ST      A9,BADDR-HCIPL,A11    BADDR =BASE ADDRESS
0402  0116
0404  20BF      PROGLD  INH
      0406      PROLO1  EQU     *
0406  5F1E              RB      RAFL1
0408  8120              LDK.L   A1,BUFF-HCIPL
040A  008A
040C  910E              ADR     A1,A11
040E  0401              LDK     A4,1
0410  0300              LDK     A3,0
0412  F324              LCR     A3,A1                 A3 = TYPE
0414  1101              ADK     A1,1
0416  0200              LDK     A2,0
0418  E224              LCR     A2,A1                 A2 = WORD COUNT
041A  1101              ADK     A1,1
041C  EB20              CWK     A3,3
041E  0003
0420  500F              RF(0)   CLCODF                BRANCH  ON CLUSTER CODE
0422  EB20              CWK     A3,4
0424  0004
0426  503A              RF(0)   CLTMOD                INTERNAL MODIFICATION
0428  EB20              CWK     A3,7
042A  0007
042C  505E              RF(0)   CLEND                 END/START
042E  5F2A              RB(7)   PROLO1
                ************************************
                * CLUSTER CODE  TYPE 3
                *       UPON ENTRY: A1=ADDRESS OF BUFF+1 (RBK
                *                   A2=WORD COUNT
                *                   A9=BADDRESS
                *                   A10=ENDADDRESS
                ***********************************************
0430  834F      CLCODF  LD      A3,BUFF+6-HCIPL,A11
0432  0090
0434  5C30              RB(4)   PROLO1                EMBK SET SKIP THE CLUSTER
0436  834E      CLCO1A  LD      A3,BUFF+4-HCIPL,A11
0438  008F
043A  A311              TM      A3,A4                 IS IT  RELOCATABLE SECTION
043C  5004              RF(0)   CLCO4
043E  3301              XRK     A3,1
0440  9306              ADR     A3,A9
0442  8524      CLCO4   LDR*    A5,A1                 A5=(RBK)
```

```
0444  1106              ADK      A1,6                     A1= ADDRESS OF ST CODE WORD IN BUFF
0446  1A03              SUK      A2,3                     A2= NUMBER OF CODE WORD
                *                                      A3= STORAGE ADDRESS
                *                                      A4= MASK FOR RBK
                *                                      A6= CODE WORD
0448  3CF1      CLC05   SRC      A4,1
044A  8624              LDR*     A6,A1
044C  EB0A              CWR      A3,A10                   COMPARE LOAD ADDRESS WITH AD OF IPL
044E  5864              RB(0)    CLC01
0450  A511              TM       A5,A4
0452  5002              RF(0)    CLC07
0454  9606              ADR      A6,A9
0456  862D      CLC07   STR      A6,A3                    STORE CODE W-RDS
0458  1102              ADK      A1,2
045A  1302              ADK      A3,2
045C  1A01              SUK      A2,1
045E  5C18              RB(4)    CLC05
0460  5F68              RB(7)    PROLO
                *
                ***************************************************************
                * INTERNAL MODIFICATION CLUSTER
                ***************************************************
0462  0701      CLIMOD  LDK      A7,1                     A7= MASK FOR ADDRESS
0464  8524              LDR*     A5,A1                    A5=(RBK)
0466  1A01              SUK      A2,1
0468  3CE1      CLIM1   SRC      A4,1
046A  1102              ADK      A1,2
046C  8324              LDR*     A3,A1                    A3=ADDRESS
046E  A31D              TM       A3,A7                    IS IT RELOCATABLE
0470  5008              RF(0)    CLIM2                    NO
0472  3301              XRK      A3,1
0474  9306              ADR      A3,A9                    YES AD BASE
0476  EB0A              CWR      A3,A10                   ADDRESS OK
0478  588F              RB(0)    CLC01
047A  1102      CLIM2   ADK      A1,2
047C  8624              LDR*     A6,A1                    TAKE CODE WORD
047E  A511              TM       A5,A4                    IS IT RELOCATABLE
0480  5002              RF(0)    CLIM3
0482  9606              ADR      A6,A9                    AD BASE
0484  862D      CLIM3   STR      A6,A3                    STORE CODE WORD
0486  1A02              SUK      A2,2
0488  5C22              RB(4)    CLIM1
048A  5F92              RB(7)    PROLO
                *******************************************
                *  CLUSTER  END/START
                ********************************************
048C  8324      CLEND   LDR*     A3,A1
048E  500A              RF(0)    CLEN3A                   FINISH WO START
0490  A311              TM       A3,A4
0492  5002              RF(0)    CLEN1
0494  9306              ADR      A3,A9
                *
                *
      0496      CLEN1   EQU      *
0496  834F              ST       A3,COREND-HCIPL,A11
0498  0356
049A  814F      CLEN3A  LD       A1,BUFF+6-HCIPL,A11   UPDATE BASE ADDRESS
049C  0090
```

```
049E  914F              AD,S     A1,BADDR-HCIPL,A11
04A0  0116
04A2  9184              ADR      A9,A1
04A4  5FAC              RB       PROLO
                  *
                  *
04A6  0000        SAVA15  DATA     0
                  *
                  ****************************************************************************
                  *
04A8  0000        COREND  DATA     0
                  *
                  *
                  ****************************************************************************
04AA  0000              DATA     0
04AC  0000              DATA     0
04AF  0000              DATA     0
04B0  0000              DATA     0
04B2  0000              DATA     0
04B4  0000              DATA     0
04B6  0000              DATA     0
04B8  0000              DATA     0
04BA  0000              DATA     0
04BC  0000              DATA     0
04BE  0000              DATA     0
04C0  0000              DATA     0
04C2  0000              DATA     0
04C4  0000              DATA     0
04C6  0000              DATA     0
04C8  0000              DATA     0
04CA  0000              DATA     0
04CC  0000              DATA     0
04CE  0000              DATA     0
04D0  0000              DATA     0
04D2  0000              DATA     0
04D4  0000              DATA     0
04D6  0000              DATA     0
04D8  0000              DATA     0
04DA  0000              DATA     0
04DC  0000              DATA     0
04DE  0000              DATA     0
04F0  0000              DATA     0
04E2  0000              DATA     0
04E4  0000              DATA     0
04E6  0000              DATA     0
04F8  0000              DATA     0
04FA  0000              DATA     0
04FC  0000              DATA     0
04FE  0000              DATA     0
04F0  0000              DATA     0
04F2  0000              DATA     0
04F4  0000              DATA     0
04F6  0000              DATA     0
04F8  0000              DATA     0
04FA  0000              DATA     0
04FC  0000              DATA     0
04FE  0000              DATA     0
0500  0000              DATA     0
```

```
0502  0000                 DATA    0
0504  0000                 DATA    0
0506  0000                 DATA    0
0508  0000                 DATA    0
050A  0000                 DATA    0
050C  0000                 DATA    0
050E  0000                 DATA    0
0510  0000                 DATA    0
0512  0000                 DATA    0
0514  0000                 DATA    0
0516  0000                 DATA    0
0518  0000                 DATA    0
051A  0000                 DATA    0
051C  0000                 DATA    0
051E  0000                 DATA    0
0520  0000                 DATA    0
                           EJECT
      0522       IPL44     EQU     *
0522  030C                 LDK     A3,NBREC
0524  8420                 LDKL    A4,BTPL
0526  0100  R
0528  0785                 LDK     A7,/85
                 * BASIC WRITE
052A  80A0                 LDKL    A8,DECB44
052C  0582  R
      052E       PUNCH4    EQU     *
052E  0550                 LDK     A5,80
0530  8620                 LDKL    A6,BUF44
0532  058E  R
      0534       PCH400    EQU     *
0534  0100                 LDK     A1,0
0536  0200                 LDK     A2,0
0538  E130                 LCR     A1,A4
053A  3964                 SRL     A1,4
053C  E921                 CCK     A1,0
053E  0000
0540  5006                 RF(0)   PCH405
0542  E921                 CCK     A1,/500
0544  0500
0546  5202                 RF(2)   PCH410              LESS THAN 5
      0548       PCH405    EQU     *
0548  2910                 ORK     A1,/10
      054A       PCH410    EQU     *
054A  3948                 SLL     A1,8
054C  E130                 LCR     A1,A4
054E  A120                 ANKL    A1,/FF0F
0550  FF0F
0552  E921                 CCK     A1,0
0554  0000
0556  5006                 RF(0)   PCH415
0558  E921                 CCK     A1,/500
055A  0500
055C  5202                 RF(2)   PCH420
      055E       PCH415    EQU     *
055E  2910                 ORK     A1,/10
      0560       PCH420    EQU     *
0560  8139                 STR     A1,A6
                 * STORE WORD
```

```
0562  1602                  ADK     A6,2                  NEXT WORD IN BUFFER
0564  1401                  ADK     A4,1                  NEXT CHZR. IN IPL
0566  1D01                  SUK     A5,1
                    *
0568  5936                  RB(1)   PCH400
056A  EB20                  CWK     A3,1                  LAST RECORD ?
056C  0001
056E  5406                  RF(4)   PCH430                NO
0570  0113                  LDK     A1,/13                XOFF
0572  8141                  ST      A1,BUF44+158
0574  062C  R
      0576          PCH430  EQU     *
0576  2804                  LKM
0578  0001                  DATA    1
057A  1B01                  SUK     A3,1
057C  5950                  RB(1)   PUNCH4
                    *
057E  2804                  LKM
0580  0003                  DATA    3
                    *
      0582          DECB44  EQU     *
0582  0003                  DATA    3
0584  058E  R               DATA    BUF44
0586  00A0                  DATA    160
0588  0000                  DATA    0
058A  0000                  DATA    0
058C  0000                  DATA    0
058E                BUF44   RES     80
                    *
                    *
                    *
                            END     BIPLPR
```

SYMBOL TABLE

ABA	0000	A	ASCII	03F6	R	ASCINP	0346	R	ASR	0010	A
BADDR	0268	R	BIPL	0100	R	BIPLPR	0000	R	BOM	0001	A
BUF44	058E	R	BUFF	01DC	R	CFZON	023A	R	CIOPTR	0294	R
CKOK	0256	R	CLCO1	03EC	R	CLCO1A	0436	R	CLCO4	0442	R
CLCO5	0448	R	CLCO7	0456	R	CLCODE	0430	R	CLEN1	0496	R
CLEN3A	049A	R	CLEND	048C	R	CLIM1	0468	R	CLIM2	047A	R
CLIM3	0484	R	CLIMOD	0462	R	CNTFLG	014C	R	COREND	04A8	R
DECB	001E	R	DECB44	0582	R	DECBUF	0020	R	ECCLS	0334	R
ECMSG	01D2	R	EIGHT	02FF	R	END1	035A	R	END2	0320	R
FIRST	031A	R	H	0000	A	HCIPL	0152	R	HCIPL1	0150	R
HCIPL2	018A	R	INP2	02DA	R	INP2R	0318	R	INPA	028A	R
INR44	02F4	R	IPL	0104	R	IPL010	0116	R	IPL020	0122	R
IPL100	012C	R	IPL110	0130	R	IPL44	0522	R	IPL88	0000	R
LDFLG	0100	R	MASTFG	022C	R	MLX1	02C4	R	MNLD	03FE	R
MODE	0000	A	NBREC	000C	A	OBJEC	03F2	R	OBJIN1	0316	R
OBJINP	02EE	R	OFLMSG	01D6	R	OTR	0242	R	OUTMSG	023C	R
PCH400	0534	R	PCH405	0548	R	PCH410	054A	R	PCH415	055E	R
PCH420	0560	R	PCH430	0576	R	PRASCA	0384	R	PRASCB	0386	R
PRASCI	0360	R	PROGLD	0404	R	PROLD	03FA	R	PROL01	0406	R
PTR	0000	A	PUNCH	000C	R	PUNCH4	052E	R	RAFL	026A	R
RAFL1	03EA	R	RAFL2	03B4	R	RAFL3	02C2	R	RELAY	03B6	R
RER	02CE	R	S	0001	A	SA	0000	A	SAVA15	04A6	R
SAVBAS	022E	R	SSTMLX	02A0	R	SSTPR1	035C	R	SSTPR2	020F	R
SSTPTR	0322	R	STAD	01D2	R	STOP	0342	R	SWITCH	03B8	R
SYSMSG	01AA	R	TMTEST	02A4	R	WER1	028C	R	WER2	028E	R

IPL52H

```
                                IDENT   IPL52H
                                ENTRY   IPL88
                                ENTRY   IPL44
                        *
                        *                       BELIER IPL GENERATION
          000C          NBREC   EQU     12
                        *
          0000          BIPLPR  EQU     *
          0000          IPL88   EQU     *
  0000    010C                  LDK     A1,NBREC
  0002    8220                  LDK.L   A2,BIPL-80
  0004    0080    R
  0006    0785                  LDK     A7,/85          BASIC WRITE
  0008    80A0                  LDKL    A8,DECB
  000A    001E    R
          000C          PUNCH   EQU     *
  000C    1250                  ADK     A2,80
  000E    8241                  ST      A2,DECBUF       SET BUFF ADDR
  0010    0020    R
  0012    2804                  LKM                     PUNCH ONE RECORD
  0014    0001                  DATA    1
  0016    1901                  SUK     A1,1            COUNT DONE ?
  0018    590F                  RB(1)   PUNCH           NO
  001A    2804                  LKM                     YES
  001C    0003                  DATA    3               EXIT
                        *
          001E          DECB    EQU     *
  001E    0003                  DATA    3               FILE CODE
  0020                  DECBUF  RES     1
  0022    0050                  DATA    80
  0024    0000                  DATA    0
  0026    0000                  DATA    0
  0028    0000                  DATA    0
                                EJECT
                                RORG    BIPLPR+/100
                        *
          0100          BIPL    EQU     *
                        *                       1ST RECORD = LOW CORE IPL
                        *                       LOW CORE IPL,1 RECORD,LOADED AT /80 AND
                        *                       STARTED AT /84
                        *
  0100    FFFF          LDFLG   DATA    /FFFF           LDFLG = -1 IF JUST LOADED
  0102    0000                  DATA    0               NOT USED
          0104          IPL     EQU     *
                        *                       ADDR OF IPL = BASE ADDR
  0104    82A0                  LDKL    A10,/84
  0106    0084
                        *                       CHECK IF JUST LOADED OR NOT
  0108    904B                  TM      LDFLG-IPL,A10
  010A    FFFC
  010C    511F                  RF(1)   IPL100          NO,NOT THE 1ST TIME
```

```
                        *
                        *
                        *                                   YES,COMPUTE HIGH CORE LIMIT
                        *
010E  8720                        LDK.L     A7,/5555             PATTERN
0110  5555
0112  81A0                        LDKL      A9,/FC00             HIGH CORE =/400
0114  FC00
      0116              IPL010    EQU       *
0116  8727                        STR       A7,A9
0118  EF26                        CWR*      A7,A9                IF THE LOCATION (A9) EXISTS,(A7)=((A9))
011A  5006                        RF(0)     IPL020               MATCH
011C  99A0                        SUKL      A9,/2000             NO,NEXT LOWER BLOCK OF 4K
011E  2000
0120  5F0C                        RB        IPL010
      0122              IPL020    EQU       *
0122  8386                        LDR       A11,A9               SAVE BASE ADDR OF HIGH CORE IPL
0124  93A0                        ADKL      A11,2                START ADD.
0126  0002
0128  8486                        LDR       A12,A9               SAVE LOAD ADD.
012A  5704                        RF        IPL110
      012C              IPL100    EQU       *
012C  94A0                        ADKL      A12,80
012E  0050
                        * LOAD ADDRESS OF NEXT RECORD
      0130              IPL110    EQU       *
0130  871E                        LDR       A7,A15               RESTORE CIO INST IN BOOT
0132  273F                        ANK       A7,/3F
0134  9720                        ADK.L     A7,/41C0             CIO INSTRUCTION
0136  41C0
0138  872D                        STR       A7,A3
                        *
013A  0557                        LDK       A5,/57               CHANGE LOC. /5A OF BOOT IN ORDER TO
013C  F541                        SC        A5,/5A               CANCEL LEADING CHARACTER FLAG
013E  005A
                        *                                   RE INITIALIZE A5,A6
0140  0550                        LDK       A5,80                # OF CHARACTERS
0142  8612                        LDR       A6,A12               LOAD ADDR.
                        *                                   CHECK IF COUNT DONE
0144  904B                        IM        CNTFLG-IPL,A10
0146  0048
0148  890E                        ABR(1)    A11
014A  0F42                        AB        /42                  START BOOT ABAIN
      014C              CNTFLG    EQU       *                    - NUMBER OF RECORDS OF HIGH CORE IPL
014C  FFF5                        DATA      1-NBREC
014E  0000                        DATA      0
0150  0000                        DATA      0
0152  0000                        DATA      0
0154  0000                        DATA      0
0156  0000                        DATA      0
0158  0000                        DATA      0
015A  0000                        DATA      0
015C  0000                        DATA      0
015E  0000                        DATA      0
0160  0000                        DATA      0
0162  0000                        DATA      0
0164  0000                        DATA      0
0166  0000                        DATA      0
```

```
00096  0168  0000          DATA    0
00097  016A  0000          DATA    0
00098  016C  0000          DATA    0
00099  016E  0000          DATA    0
00100  0170  0000          DATA    0
00101  0172  0000          DATA    0
00102  0174  0000          DATA    0
00103  0176  0000          DATA    0
00104  0178  0000          DATA    0
00105  017A  0000          DATA    0
00106  017C  0000          DATA    0
00107  017E  0000          DATA    0
00108  0180  0000          DATA    0
00109  0182  0000          DATA    0
00110  0184  0000          DATA    0
00111  0186  0000          DATA    0
00112  0188  0000          DATA    0
00113  018A  0000          DATA    0
00114  018C  0000          DATA    0
00115  018E  0000          DATA    0
00116  0190  0000          DATA    0
00117  0192  0000          DATA    0
00118  0194  0000          DATA    0
00119  0196  0000          DATA    0
00120  0198  0000          DATA    0
00121  019A  0000          DATA    0
00122  019C  0000          DATA    0
00123  019E  0000          DATA    0
00124  01A0  0000          DATA    0
00125  01A2  0000          DATA    0
00126  01A4  0000          DATA    0
00127  01A6  0000          DATA    0
00128  01A8  0000          DATA    0
00129  01AA  0000          DATA    0
00130  01AC  0000          DATA    0
00131  01AE  0000          DATA    0
00132  01B0  0000          DATA    0
00133  01B2  0000          DATA    0
00134  01B4  0000          DATA    0
00135  01B6  0000          DATA    0
00136  01B8  0000          DATA    0
00137  01BA  0000          DATA    0
00138  01BC  0000          DATA    0
00139  01BE  0000          DATA    0
00140  01C0  0000          DATA    0
00141  01C2  0000          DATA    0
00142  01C4  0000          DATA    0
00143  01C6  0000          DATA    0
00144  01C8  0000          DATA    0
00145  01CA  0000          DATA    0
00146  01CC  0000          DATA    0
00147  01CE  0000          DATA    0
00148  01D0  0000          DATA    0
00149  01D2  0000          DATA    0
00150  01D4  0000          DATA    0
00151  01D6  0000          DATA    0
00152  01D8  0000          DATA    0
00153  01DA  0000          DATA    0
```

```
0IDC  0000                  DATA     0
01DE  0000                  DATA     0
01E0  0000                  DATA     0
01E2  0000                  DATA     0
01E4  0000                  DATA     0
01E6  0000                  DATA     0
01F8  0000                  DATA     0
01FA  0000                  DATA     0
01EC  0000                  DATA     0
01EE  0000                  DATA     0
01F0  0000                  DATA     0
01F2  0000                  DATA     0
01F4  0000                  DATA     0
01F6  0000                  DATA     0
01F8  0000                  DATA     0
01FA  0000                  DATA     0
01FC  0000                  DATA     0
01FE  0000                  DATA     0
0200  0000                  DATA     0
0202  0000                  DATA     0
0204  0000                  DATA     0
                            EJECT
                                                               HIGH CORE IPL
      0000        PTR       EQU      0                         DEVICE ADDR WILL BE INITIALIZED BY HCIPL
      0010        ASR       EQU      /10
      0001        S         EQU      1
      0000        H         EQU      0
      0001        BOM       EQU      1
      0000        SA        EQU      0
                  *
      0001        MODE      EQU      BOM                       FOR MESSAGE  OUTPUT
                            RORG     BIPL+80                   1ST RECORD OF HIGH CORE IPL
      0150        HCIPL1    EQU      *                         1ST WORD OF THE CURRENT RECORD
0150  FEFE                  DATA     /FEFE
      0152        HCIPL     EQU      *
0152  85A0                  LDKL     A13,OUTMSG-HCIPL
0154  00EA
0156  958E                  ADR      A13,A11                   LOAD A13 WITH ADDR OF OUTMSG ROUTINE
0158  868F                  LDR      A14,A11
015A  9640                  ADKL     A14,CEZON-HCIPL
015C  00F8
                  *                  ADDR OF STACK
015E  871E                  LDR      A7,A15                    GET DEVICE ADDR
0160  87CF                  ST       A15,SAVA15-HCIPL,A11      SAVE  4*4  FLAG
0162  0366
0164  5606                  RF(6)    *+8
0166  904F                  TM       COREND+2-HCIPL,A11
0168  036A
016A  5C06                  RB(4)    *-4
016C  273F                  ANK      A7,/3F
                  *                        INIT IO INSTRUCTIONS
016E  974F                  ADS      A7,CIOPTR-HCIPL,A11
0170  0142
0172  974F                  ADS      A7,INP2-HCIPL,A11
0174  0188
0176  974F                  ADS      A7,END2-HCIPL,A11
0178  01CE
```

```
00204  017A  974F            ADS     A7,END1-HCIPL,A11
       017C  0208
00205  017E  974F            ADS     A7,INR44-HCIPL,A11
       0180  01A2
00206  0182  811E            LDR     A1,A15
00207  0184  39C8            SLC     A1,8              TEST MULTI-SINGLE DEVICE CONT.
00208  0186  5602            RF(6)   HCIPL2
00209  0188  270F            ANK     A7,/F             MULTI DEVICE CONTROLLER
00210        018A   HCIPL2   EQU     *
00211  018A  974F            AD.S    A7,SSTMLX-HCIPL,A11
       018C  014F
00212  018E  974F            ADS     A7,SSTPR2-HCIPL,A11
       0190  018C
00213  0192  974F            ADS     A7,SSTPTR-HCIPL,A11
       0194  01D0
00214  0196  974F            ADS     A7,SSTPR1-HCIPL,A11
       0198  020A
00215  019A  3F41            SLL     A7,1
00216  019C  974F            AD.S    A7,WER1-HCIPL,A11  INIT WER/RER INST
       019E  013A
00217  01A0  974F            AD.S    A7,WER2-HCIPL,A11
       01A2  013C
00218  01A4  974F            AD.S    A7,RER-HCIPL,A11
       01A6  017C
00219  01A8  57AC            RF      CKOK
00220  01AA  4F42   SYSMSG   DATA    'OBJCT TAPE ON RE'
       01AC  4A43
       01AE  5420
       01B0  5441
       01B2  5045
       01B4  204F
       01B6  4E20
       01B8  5245
00221  01BA  4144            DATA    'ADER, THINK OF B'
       01BC  4552
       01BE  2E20
       01C0  5448
       01C2  494F
       01C4  4B20
       01C6  4F46
       01C8  2042
00222  01CA  4153            DATA    'ASE !'
       01CC  4520
       01CE  2120
00223  01D0  0D0A            DATA    /0D0A
00224        01D2   STAD     EQU     *
00225  01D2  4543   ECMSG    DATA    'EC'
00226  01D4  0D0A            DATA    X'0D0A'
00227  01D6  4F56   OFLMSG   DATA    'OVFR'
       01D8  4652
00228  01DA  0D0A            DATA    /0D0A
00229  01DC         BUFF     RES     39
00230  022A  FFFF            DATA    /FFFF
00231  022C  0000   MASTFG   DATA    0
00232  022E  0000   SAVBAS   DATA    0
00233  0230                  RES     6                 * STACK  AREA
00234        023A   CFZON    EQU     *-2
00235        023C   OUTMSG   EQU     *
```

```
023C  0304              LDK       A3,4
023E  43D0              CIO       A3,S,ASR
0240  F324              LCR       A3,A1
0242  4310    OTR       OTR       A3,0,ASR
0244  5C04              RB(4)     *-2
0246  1101              ADK       A1,1
0248  1A01              SUK       A2,1
024A  5C0C              RB(4)     OTR-2
024C  0300              LDK       A3,0
024E  4390              CIO       A3,H,ASR
0250  4BD0              SST       A3,ASR
0252  5C04              RB(4)     *-2
0254  F03A              RTN       A14
                *
                *
      0256      CKOK    EQU       *
0256  81A0              LDK.L     A9,0                      SET LOADING ADDRESS
0258  0000
                *
                *
                *                                GO LOAD SYSTEM
025A  84A0              LDK.L     A12,MNLD-HCIPL
025C  029C
025E  948E              ADR       A12,A11
0260  F693              CFR       A14,A12
                *
                *
0262  810A              LDR       A1,A10                    TEST START ADDRESS
0264  8C04              ABR(4)    A1                        EOS  OR EOF HAS BEEN READ
0266  207F              HLT                                 NO START ADDRESS
0268  0000      BADDR   DATA      0
026A  0300      RAFL    LDK       A3,0
026C  0700              LDK       A7,0
026E  8520              LDK.L     A5,BUFF-HCIPL
0270  008A
0272  950E              ADR       A5,A11
0274  80CE              LD        A8,SAVA15-HCIPL,A11
0276  0366
0278  5610              RF(6)     INPA
027A  0604              LDK       A6,4                      4*4 SO ASR
027C  0111              LDK       A1,/11                    SEND X-ON
027E  46D0              CIO       A6,S,ASR
0280  4110              OTR       A1,0,ASR
0282  5C04              RB(4)     *-2
0284  4290              CIO       A2,H,ASR
0286  4AD0              SST       A2,ASR
0288  5C04              RB(4)     *-2
      028A      INPA    EQU       *
028A  0650              LDK       A6,80                     LENGTH
028C  7600      WER1    WER       A6,0                      INIT MLX WHATEVER
028E  7501      WER2    WER       A5,1                      CHANNEL IS
0290  8602              LDR       A6,A8
0292  3E68              SRL       A6,8
      0294      CIOPTR  EQU       *
0294  46C0              CIO       A6,S,PTR                  AVAILABLE ON THE PAPER READER
0296  5C04              RB(4)     *-2
0298  824E              LD        A2,SAVA15-HCIPL,A11
029A  0366
```

```
029C  3AC3              SLC     A2,3            MLX ?
029E  523A              RF(2)   INP2            NO
      02A0    SSTMLX    EQU     *
02A0  49C0              SST     A1,PTR
02A2  5C04              RB(4)   *-2
      02A4    TMTEST    EQU     *
02A4  A120              ANKL    A1,/1007        TAPE MARK OR FATAL ERROR ?
02A6  1007
02A8  501A              RF(0)   MLX1            NO
02AA  2107              ANK     A1,/7           FATAL ERROR ?
02AC  5486              RF(4)   ECCLS           YES
              *
02AE  8120              LDK.L   A1,':E'         PUT EOF IN BUFF
02B0  3A45
02B2  8135              STR     A1,A5
02B4  8120              LDK.L   A1,'OF'
02B6  4F46
02B8  8155              ST      A1,2,A5
02BA  0002
02BC  0704              LDK     A7,4            UPDATA POINTRERS
02BE  951C              ADR     A5,A7
02C0  579E              RF      PRASCI
              *
02C2  5F5A    RAFL3     RB      RAFL            RELAY TOWARDS RAFL
              *
      02C4    MLX1      EQU     *
02C4  E134              LCR     A1,A5
02C6  F920              CWK     A1,/18          IS IT ASCII
02C8  0018
02CA  55FA              RF(5)   RELAY           NO
02CC  0750              LDK     A7,80           YES
02CE  7E00    RER       RER     A6,0            READ REMAINING LENGTH
02D0  A620              ANK.L   A6,/FFF
02D2  0FFF
02D4  9F18              SUR     A7,A6           COMPUTE TRANSMITTED LGT
02D6  951C              ADR     A5,A7           UPDATE BUFF POINTER
02D8  5786              RF      PRASCI
02DA  4A00    INP2      INR     A2,0,PTR
02DC  500A              RF(0)   SWITCH          INR HAS BEEN ACCEPTED ;
              *                                 PROCESS THE CHARACTER
02DE  4AC0    SSTPR2    SST     A2,PTR          TRY A SST
02E0  5C08              RB(4)   INP2            TRY AGAIN INR WHEN SST REFUSED
02E2  8108              LDR     A1,A2           SAVE STATUS IN A1
02E4  1300              ADK     A3,0            CHECK WETHER ASCII OR OBJECT
02E6  5278              RF(2)   PRASCI          RECORD WAS ASCII ; PRINT IT
02E8  0600              LDK     A6,0            STORE A NON ASCII CHARACTER AT
02EA  8635              STR     A6,A5           THE END OF BUFFER ;
              *                                 TO BE USED IN MX1
02EC  5F4A              RB      TMTEST
      02EE    OBJINP    EQU     *
02EE  8082              LDR     A8,A8
02F0  560C              RF(6)   EIGHT           8=8 DO NOT INPUT 2ND CHARACTER
02F2  220F              ANK     A2,/F
02F4  4E00    INR44     INR     A6,0,PTR
02F6  5C04              RB(4)   *-2
02F8  260F              ANK     A6,/F
02FA  3A44              SLL     A2,4
02FC  9218              ADR     A2,A6           JOIN THE TWO HALF CHARACTERS
```

```
        02FF     FIGHT    EQU     *
  02FE  EF04              CWR     A7,A1
  0300  501E              RF(0)   END2
  0302  E235              SCR     A2,A5
  0304  B408              XRR     A4,A2
  0306  1501              ADK     A5,1
  0308  FF20              CWK     A7,1
  030A  0001
  030C  5408              RF(4)   OBJIN1
  030E  0400              LDK     A4,0
  0310  8108              LDR     A1,A2
  0312  9108              ADR     A1,A2
  0314  1103              ADK     A1,3
  0316  1701     OBJIN1   ADK     A7,1
  0318  5F40     INP2R    RB      INP2
  031A  2207     FIRST    ANK     A2,/7
  031C  0150              LDK     A1,80
  031E  5F22              RB(7)   FIGHT
  0320  4280     END2     CIO     A2,H,PTR
        0322     SSTPTR   EQU     *
  0322  49C0              SST     A1,PTR
  0324  5C04              RB(4)   *-2
  0326  8082              LDR     A8,A8                  IF ASR WAIT
  0328  5606              RF(6)   *+8
  032A  904F              IM      COREND+2-HCIPL,A11     WAIT
  032C  036A
  032E  5C06              RB(4)   *-4
  0330  24FF              ANK     A4,/FF
  0332  50E6              RF(0)   PROLO1+2
        0334     ECCLS    EQU     *
  0334  0180              LDK     A1,ECMSG-HCIPL
  0336  910F              ADR     A1,A11
  0338  0204              LDK     A2,4
  033A  84A0              LDK.L   A12,OUTMSG-HCIPL
  033C  00FA
  033E  948E              ADR     A12,A11
  0340  F693              CER     A14,A12
  0342  207F     STOP     HLT
  0344  5FDC              RB(7)   RAFL
  0346  EA20     ASCINP   CWK     A2,/0D
  0348  000D
  034A  500F              RF(0)   END1
  034C  FF20              CWK     A7,68
  034E  0044
  0350  5878              RB(0)   INP2
  0352  E235              SCR     A2,A5
  0354  1501              ADK     A5,1
  0356  1701              ADK     A7,1
  0358  5F80              RB(7)   INP2
  035A  4280     END1     CIO     A2,H,PTR
        035C     SSTPR1   EQU     *
  035C  4AC0              SST     A2,PTR
  035E  5C04              RB(4)   *-2
        0360     PRASCI   EQU     *
  0360  018A              LDK     A1,BUFF-HCIPL
  0362  910F              ADR     A1,A11
  0364  821C              LDR     A2,A7
  0366  0320              LDK     A3,/20
```

```
00394  0368  E335          SCR     A3,A5
00395  036A  1501          ADK     A5,1
00396  036C  8320          LDK.L   A3,/0D0A
       036E  0D0A
00397  0370  8335          STR     A3,A5
00398  0372  1203          ADK     A2,3
00399  0374  3A61          SRL     A2,1
00400  0376  3A41          SLL     A2,1
00401  0378  8082          LDR     A8,A8                  IF ASR DO NOT OUTPUT ASCII
00402  037A  5608          RF(6)   PRASCA                 A SECOND TIME (ALREADY PRINTED WHEN READ
00403  037C  904F          IM      COREND+2-HCIPL,A11     BUT WAIT ABIT
       037E  036A
00404  0380  5C06          RB(4)   *-4
00405  0382  5702          RF      PRASCB
00406        0384  PRASCA  EQU     *
00407  0384  5700          RF      *+2                    *****************
00408        0386  PRASCB  EQU     *
00409  0386  028A          LDK     A2,BUFF-HCIPL
00410  0388  920E          ADR     A2,A11
00411  038A  8328          LDR*    A3,A2
00412  038C  EB20          CWK     A3,/3A45               * :E
       038E  3A45
00413  0390  5CD0          RB(4)   RAFL3
00414  0392  1202          ADK     A2,2
00415  0394  8328          LDR*    A3,A2
00416  0396  EB20          CWK     A3,/4F46               *OF
       0398  4F46
00417  039A  5CDA          RB(4)   RAFL3
00418              *
00419              *
00420              *                           SET RETURN PARAMETERS
00421              *
00422  039C  82CF          LD      A10,COREND-HCIPL,A11   START ADDRESS IN A10
       039E  0368
00423  03A0  84A0          LDK.L   A12,MNLD-HCIPL
       03A2  029C
00424  03A4  948E          ADR     A12,A11                MNLD ADDRESS  IN A12
00425  03A6  81CF          LD      A9,SAVBAS-HCIPL,A11    LOADING BASE  IN A9
       03A8  00DC
00426  03AA  80CE          LD      A8,BADDR-HCIPL,A11     ENDING ADDRESS IN A8
       03AC  0116
00427  03AE  874E          LD      A7,MASTFG-HCIPL,A11    MASTER FLAG IN A7
       03B0  00DA
00428              *
00429              *
00430  03B2  F03A          RTN     A14                    RETURN TO CALLING
00431              *
00432  03B4  5FF4  RAFL2   RB      RAFL3                  * RELAY TOWARDS RAFL
00433  03B6  5762  RELAY   RF      PROL01+2
00434              *
00435  03B8  1300  SWITCH  ADK     A3,0
00436  03BA  59CE          RB(1)   OBJINP
00437  03BC  227F          ANK     A2,/7F
00438  03BE  58E6          RB(0)   INP2
00439  03C0  EA20          CWK     A2,/7F
       03C2  007F
00440  03C4  58EC          RB(0)   INP2
00441  03C6  1300          ADK     A3,0
```

```
03C8  5A84            RB(2)    ASCINP
03CA  FA20            CWK      A2,/1F
03CC  001F
03CE  5116            RF(1)    ASCII
03D0  FA20            CWK      A2,/14
03D2  0014
03D4  510C            RF(1)    OBJEC
03D6  FA20            CWK      A2,8
03D8  0008
03DA  5206            RF(2)    OBJEC
03DC  FA20            CWK      A2,/10
03DE  0010
03E0  5CCA            RB(4)    INP2R
03E2  1301   OBJEC    ADK      A3,1
03E4  5FCC            RB(7)    FIRST
03E6  1B01   ASCII    SUK      A3,1
03E8  5FA4            RB(7)    ASCINP
                      FJECT
             *
             *
             *   MAIN  LOADING  PART          ENTRY PARAMETFRS :
             *
             *                                  A1 = MASTER FLAG
             *                                  A9 = BASE ADDRESS
             *
             *
             *
             *                   1- OUTPUT MSG REQUESTING  TAPE ON READER
             *                   2- PERFORMS A 'HALT' - THE USER HAS THEN THE
             *                      POSSIBILITY TO ALTER THE BASE ADDRESS (REG,A9)
             *                                     AND  THE MASTER FLAG  (REG,A1)
             *                                                 (A1=0 MASTER)
             *                                                 (A1=1 USER)
             *                   3- LOADING PROCESS STARTS  WHEN USER DEPPRESS START
             *                      BUTTON
             *
             *
             *
             *
      03EA   RAFL1    EQU      *
03EA  5F38            RB       RAFL2
             *
             *
             *                                   ERRONNOUS CLUSTER,PRINT ERR MESSAGE
      03EC   CLC01    EQU      *
03EC  5FBA            RB       ECCLS
             *
             *
             *
             *
      03EE   MNLD     EQU      *
03EE  8404            LDR      A4,A1                    SAVE  MASTER FLAG
03F0  0200            LDK      A2,0
03F2  824F            ST       A2,COREND-HCIPL,A11      RAZ  START  ADDRESS
03F4  0368
                      IFT,     MODE=BOM
03F6  8120            LDK.L    A1,SYSMSG-HCIPL
03F8  0058
```

```
03FA  910E           ADR     A1,A11
03FC  0228           LDK     A2,/28
03FE  5700           RF      *+2                       ****************
                *
                *
0400  8110           LDR     A1,A4                     RESTORE MASTER FLAG
                *
0402  207F           HLT                               HALT FOR EVENTUAL REGISTERS MODIF
                *
0404  814F           ST      A1,MASTFG-HCIPL,A11
0406  00DA
                     XIF
0408  81CF           ST      A9,SAVBAS-HCIPL,A11
040A  00DC
                *
                *
                *
                * PROCESS LOADING : THIS  MODULE READ A  CLUSTER
                *          AND  BRANCH ACCORDING TO THE CLUSTER TYPE
                *          ON EXIT   A1 = BUFF  ADDRESS +1
                *                    A2 = WORD  COUNT
                *                    A3 = TYPE
                *          THE TYPE MUST BE    3,4,7   IF NOT THIS :HALT
040C  82A0      PROLO    LDK.L   A10,STAD-HCIPL        END ADDRESS
040E  0080
0410  928F               ADR     A10,A11
      0000      ABA      EQU     0
0412  81CF               ST      A9,BADDR-HCIPL,A11    BADDR =BASE ADDRESS
0414  0116
0416  20BF      PROGLD   INH
      0418      PROLO1   EQU     *
0418  5F30               RB      RAFL1
041A  8120               LDK.L   A1,BUFF-HCIPL
041C  008A
041E  910E               ADR     A1,A11
0420  0401               LDK     A4,1
0422  0300               LDK     A3,0
0424  E324               LCR     A3,A1                 A3 = TYPE
0426  1101               ADK     A1,1
0428  0200               LDK     A2,0
042A  E224               LCR     A2,A1                 A2 = WORD COUNT
042C  1101               ADK     A1,1
042E  EB20               CWK     A3,3
0430  0003
0432  500E               RF(0)   CLCODE                BRANCH  ON CLUSTER CODE
0434  EB20               CWK     A3,4
0436  0004
0438  503A               RF(0)   CLIMOD                INTERNAL MODIFICATION
043A  EB20               CWK     A3,7
043C  0007
043E  505E               RF(0)   CLEND                 END/START
0440  5F2A               RB(7)   PROLO1
                **********************************
                * CLUSTER CODE  TYPE 3
                *        UPON ENTRY: A1=ADDRESS OF BUFF+1 (RBK
                *                    A2=WORD COUNT
                *                    A9=BADDRESS
                *                    A10=ENDADDRESS
                ********************************************
```

```
0442  834E   CLCODE  LD      A3,BUFF+6=HCIPL,A11
0444  0090
0446  5C30           RB(4)   PROLO1             EMBK SET SKIP THE CLUSTER
0448  834E   CLCO1A  LD      A3,BUFF+4=HCIPL,A11
044A  008E
044C  A311           TM      A3,A4              IS IT  RELOCATABLE SECTION
044E  5004           RF(0)   CLCO4
0450  3301           XRK     A3,1
0452  9306           ADR     A3,A9
0454  8524   CLCO4   LDR*    A5,A1              A5=(RBK)
0456  1106           ADK     A1,6               A1= ADDRESS OF ST CODE WORD IN BUFF
0458  1A03           SUK     A2,3               A2= NUMBER OF CODE WORD
             *                               A3= STORAGE ADDRESS
             *                               A4= MASK FOR RBK
             *                               A6= CODE WORD
045A  3CE1   CLCO5   SRC     A4,1
045C  8624           LDR*    A6,A1
045E  EB0A           CWR     A3,A10             COMPARE LOAD ADDRESS WITH AD OF IPL
0460  5876           RB(0)   CLCO1
0462  A511           TM      A5,A4
0464  5002           RF(0)   CLCO7
0466  9606           ADR     A6,A9
0468  862D   CLCO7   STR     A6,A3              STORE CODE W=RDS
046A  1102           ADK     A1,2
046C  1302           ADK     A3,2
046E  1A01           SUK     A2,1
0470  5C18           RB(4)   CLCO5
0472  5F68           RB(7)   PROLO
             *
             ******************************************************************
             * INTERNAL MODIFICATION CLUSTER
             ************************************************
0474  0701   CLIMOD  LDK     A7,1               A7= MASK FOR ADDRESS
0476  8524           LDR*    A5,A1              A5=(RBK)
0478  1A01           SUK     A2,1
047A  3CE1   CLIM1   SRC     A4,1
047C  1102           ADK     A1,2
047E  8324           LDR*    A3,A1              A3=ADDRESS
0480  A31D           TM      A3,A7              IS IT RELOCATABLE
0482  5008           RF(0)   CLIM2              NO
0484  3301           XRK     A3,1
0486  9306           ADR     A3,A9              YES AD BASE
0488  EB0A           CWR     A3,A10             ADDRESS OK
048A  58A0           RB(0)   CLCO1
048C  1102   CLIM2   ADK     A1,2
048E  8624           LDR*    A6,A1              TAKE CODE WORD
0490  A511           TM      A5,A4              IS IT RELOCATABLE
0492  5002           RF(0)   CLIM3
0494  9606           ADR     A6,A9              AD BASE
0496  862D   CLIM3   STR     A6,A3              STORE CODE WORD
0498  1A02           SUK     A2,2
049A  5C22           RB(4)   CLIM1
049C  5F92           RB(7)   PROLO
             ******************************************
             *  CLUSTER  END/START
             *******************************************
049E  8324   CLEND   LDR*    A3,A1
```

```
04A0  500A           RF(0)   CLEN3A                 FINISH WD START
04A2  A311           TM      A3,A4
04A4  5002           RF(0)   CLEN1
04A6  9306           ADR     A3,A9
                *
                *
      04A8      CLEN1    EQU     *
04A8  834F               ST      A3,COREND-HCIPL,A11
04AA  0368
04AC  814E      CLEN3A   LD      A1,BUFF+6-HCIPL,A11   UPDATE BASE ADDRESS
04AE  0090
04B0  914F               AD,S    A1,BADDR-HCIPL,A11
04B2  0116
04B4  9184               ADR     A9,A1
04B6  5FAC               RB      PROLO
                *
                *
04B8  0000      SAVA15   DATA    0
                *
                *************************************************************************
                *
04BA  0000      COREND   DATA    0
                *
                *
                *************************************************************************
04BC  0000               DATA    0
04BE  0000               DATA    0
04C0  0000               DATA    0
04C2  0000               DATA    0
04C4  0000               DATA    0
04C6  0000               DATA    0
04C8  0000               DATA    0
04CA  0000               DATA    0
04CC  0000               DATA    0
04CE  0000               DATA    0
04D0  0000               DATA    0
04D2  0000               DATA    0
04D4  0000               DATA    0
04D6  0000               DATA    0
04D8  0000               DATA    0
04DA  0000               DATA    0
04DC  0000               DATA    0
04DE  0000               DATA    0
04E0  0000               DATA    0
04E2  0000               DATA    0
04E4  0000               DATA    0
04E6  0000               DATA    0
04E8  0000               DATA    0
04EA  0000               DATA    0
04EC  0000               DATA    0
04EE  0000               DATA    0
04F0  0000               DATA    0
04F2  0000               DATA    0
04F4  0000               DATA    0
04F6  0000               DATA    0
04F8  0000               DATA    0
04FA  0000               DATA    0
```

```
00654  04FC  0000              DATA    0
00655  04FE  0000              DATA    0
00656  0500  0000              DATA    0
00657  0502  0000              DATA    0
00658  0504  0000              DATA    0
00659  0506  0000              DATA    0
00660  0508  0000              DATA    0
00661  050A  0000              DATA    0
00662  050C  0000              DATA    0
00663  050E  0000              DATA    0
00664  0510  0000              DATA    0
00665  0512  0000              DATA    0
00666  0514  0000              DATA    0
00667  0516  0000              DATA    0
00668  0518  0000              DATA    0
00669  051A  0000              DATA    0
00670  051C  0000              DATA    0
00671  051E  0000              DATA    0
00672  0520  0000              DATA    0
00673  0522  0000              DATA    0
00674  0524  0000              DATA    0
00675  0526  0000              DATA    0
00676  0528  0000              DATA    0
00677  052A  0000              DATA    0
00678  052C  0000              DATA    0
00679  052E  0000              DATA    0
00680  0530  0000              DATA    0
00681  0532  0000              DATA    0
00682                          EJECT
00683        0534      IPL44   EQU     *
00684  0534  030C              LDK     A3,NBREC
00685  0536  8420              LDKL    A4,BIPL
       0538  0100  R
00686  053A  0785              LDK     A7,/85
00687                  * BASIC WRITE
00688  053C  80A0              LDKL    A8,DFCB44
       053E  0594  R
00689        0540      PUNCH4  EQU     *
00690  0540  0550              LDK     A5,80
00691  0542  8620              LDKL    A6,BUF44
       0544  05A0  R
00692        0546      PCH400  EQU     *
00693  0546  0100              LDK     A1,0
00694  0548  0200              LDK     A2,0
00695  054A  E130              LCR     A1,A4
00696  054C  3964              SRL     A1,4
00697  054E  F921              CCK     A1,0
       0550  0000
00698  0552  5006              RF(0)   PCH405
00699  0554  F921              CCK     A1,/500
       0556  0500
00700  0558  5202              RF(2)   PCH410              LESS THAN 5
00701        055A      PCH405  EQU     *
00702  055A  2910              ORK     A1,/10
00703        055C      PCH410  EQU     *
00704  055C  3948              SLL     A1,8
00705  055E  E130              LCR     A1,A4
```

```
0560  A120                ANKL     A1,/FF0F
0562  FF0F
0564  E921                CCK      A1,0
0566  0000
0568  5006                RF(0)    PCH415
056A  E921                CCK      A1,/500
056C  0500
056E  5202                RF(2)    PCH420
      0570      PCH415    EQU      *
0570  2910                ORK      A1,/10
      0572      PCH420    EQU      *
0572  8139                STR      A1,A6
                * STORE WORD
0574  1602                ADK      A6,2             NEXT WORD IN BUFFER
0576  1401                ADK      A4,1             NEXT CHZR, IN IPL
0578  1D01                SUK      A5,1
                *
057A  5936                RB(1)    PCH400
057C  EB20                CWK      A3,1             LAST RECORD ?
057E  0001
0580  5406                RF(4)    PCH430           NO
0582  0113                LDK      A1,/13           XOFF
0584  8141                ST       A1,BUF44+158
0586  063E  R
      0588      PCH430    EQU      *
0588  2804                LKM
058A  0001                DATA     1
058C  1B01                SUK      A3,1
058E  5950                RB(1)    PUNCH4
                *
0590  2804                LKM
0592  0003                DATA     3
                *
      0594      DECB44    EQU      *
0594  0003                DATA     3
0596  05A0  R             DATA     BUF44
0598  00A0                DATA     160
059A  0000                DATA     0
059C  0000                DATA     0
059E  0000                DATA     0
05A0            BUF44     RES      80
                *
                *
                *
                          END      BIPLPR
```

SYMBOL TABLE

ABA	0000	A	ASCII	03E6	R	ASCINP	0346	R	ASR	0010	A
BADDR	0268	R	BIPL	0100	R	BIPLPR	0000	R	BOM	0001	A
BUF44	05A0	R	BUFF	01DC	R	CFZON	023A	R	CIOPTR	0294	R
CKOK	0256	R	CLC01	03FC	R	CLC01A	0448	R	CLC04	0454	R
CLC05	045A	R	CLC07	0468	R	CLCODE	0442	R	CLEN1	04A8	R
CLEN3A	04AC	R	CLEND	049E	R	CLIM1	047A	R	CLIM2	048C	R
CLIM3	0496	R	CLIMOD	0474	R	CNTFLG	014C	R	COREND	04BA	R
DECB	001E	R	DECB44	0594	R	DECBUF	0020	R	ECCLS	0334	R
ECMSG	01D2	R	EIGHT	02FF	R	END1	035A	R	END2	0320	R
FIRST	031A	R	H	0000	A	HCIPL	0152	R	HCIPL1	0150	R
HCIPL2	018A	R	INP2	02DA	R	INP2R	0318	R	INPA	028A	R
INR44	02F4	R	IPL	0104	R	IPL010	0116	R	IPL020	0122	R
IPL100	012C	R	IPL110	0130	R	IPL44	0534	R	IPL88	0000	R
LDFLG	0100	R	MASTFG	022C	R	MLX1	02C4	R	MNLD	03FF	R
MODE	0001	A	NBREC	000C	A	OBJEC	03E2	R	OBJIN1	0316	R
OBJINP	02EE	R	OFLMSG	01D6	R	OTR	0242	R	OUTMSG	023C	R
PCH400	0546	R	PCH405	055A	R	PCH410	055C	R	PCH415	0570	R
PCH420	0572	R	PCH430	0588	R	PRASCA	0384	R	PRASCB	0386	R
PRASCI	0360	R	PROGLD	0416	R	PROLO	040C	R	PROLO1	0418	R
PTR	0000	A	PUNCH	000C	R	PUNCH4	0540	R	RAFL	026A	R
RAFL1	03EA	R	RAFL2	03B4	R	RAFL3	02C2	R	RELAY	03B6	R
RER	02CE	R	S	0001	A	SA	0000	A	SAVA15	04B8	R
SAVBAS	022E	R	SSTMLX	02A0	R	SSTPR1	035C	R	SSTPR2	02DE	R
SSTPTR	0322	R	STAD	01D2	R	STOP	0342	R	SWITCH	03B8	R
SYSMSG	01AA	R	TMTEST	02A4	R	WER1	028C	R	WER2	028E	R

IPL52N

```
00000                           IDENT   IPL52N
00001                           ENTRY   IPL88
00002                           ENTRY   IPL44
00003                   *
00004                   *                         BELIER IPL GENERATION
00005        000C       NBREC   EQU     12
00006                   *
00007        0000       BIPLPR  EQU     *
00008        0000       IPL88   EQU     *
00009  0000  010C               LDK     A1,NBREC
00010  0002  8220               LDK.L   A2,BIPL-80
       0004  0080  R
00011  0006  0785               LDK     A7,/85              BASIC WRITE
00012  0008  80A0               LDKL    A8,DECB
       000A  001E  R
00013        000C       PUNCH   EQU     *
00014  000C  1250               ADK     A2,80
00015  000E  8241               ST      A2,DECBUF           SET BUFF ADDR
       0010  0020  R
00016  0012  2804               LKM                         PUNCH ONE RECORD
00017  0014  0001               DATA    1
00018  0016  1901               SUK     A1,1                COUNT DONE ?
00019  0018  590E               RB(1)   PUNCH               NO
00020  001A  2804               LKM                         YES
00021  001C  0003               DATA    3                   EXIT
00022                   *
00023        001E       DECB    EQU     *
00024  001E  0003               DATA    3                   FILE CODE
00025  0020             DECBUF  RES     1
00026  0022  0050               DATA    80
00027  0024  0000               DATA    0
00028  0026  0000               DATA    0
00029  0028  0000               DATA    0
00030                           EJECT
00031                           RORG    BIPLPR+/100
00032                   *
00033        0100       BIPL    EQU     *
00034                   *                         1ST RECORD = LOW CORE IPL
00035                   *                         LOW CORE IPL,1 RECORD,LOADED AT /80 AND
00036                   *                         STARTED AT /84
00037                   *
00038  0100  FFFF       LDFLG   DATA    /FFFF               LDFLG = -1 IF JUST LOADED
00039  0102  0000               DATA    0                   NOT USED
00040        0104       IPL     EQU     *
00041                   *                         ADDR OF IPL = BASE ADDR
00042  0104  82A0               LDKL    A10,/84
       0106  0084
00043                   *                         CHECK IF JUST LOADED OR NOT
00044  0108  9048               IM      LDFLG-IPL,A10
       010A  FFFC
00045  010C  511E               RF(1)   IPL100              NO,NOT THE 1ST TIME
```

```
                 *
                 *
                 *                                YES,COMPUTE HIGH CORE LIMIT
                 *
010E  8720                LDK.L    A7,/5555               PATTERN
0110  5555
0112  81A0                LDKL     A9,/FC00               HIGH CORE -/400
0114  FC00
      0116       IPL010   EQU      *
0116  8727                STR      A7,A9
0118  EF26                CWR*     A7,A9                  IF THE LOCATION (A9) EXISTS,(A7)=((A9))
011A  5006                RF(0)    IPL020                 MATCH
011C  99A0                SUKL     A9,/2000               NO,NEXT LOWER BLOCK OF 4K
011E  2000
0120  5F0C                RB       IPL010
      0122       IPL020   EQU      *
0122  8386                LDR      A11,A9                 SAVE BASE ADDR OF HIGH CORE IPL
0124  93A0                ADKL     A11,2                  START ADD.
0126  0002
0128  8486                LDR      A12,A9                 SAVE LOAD ADD.
012A  5704                RF       IPL110
      012C       IPL100   EQU      *
012C  94A0                ADKL     A12,80
012E  0050
                 * LOAD ADDRESS OF NEXT RECORD
      0130       IPL110   EQU      *
0130  871E                LDR      A7,A15                 RESTORE CIO INST IN BOOT
0132  273F                ANK      A7,/3F
0134  9720                ADK.L    A7,/41C0               CIO INSTRUCTION
0136  41C0
0138  872D                STR      A7,A3
                 *
013A  0557                LDK      A5,/57                 CHANGE LOC. /5A OF BOOT IN ORDER TO
013C  E541                SC       A5,/5A                 CANCEL LEADING CHARACTER FLAG
013E  005A
                 *                                RE INITIALIZE A5,A6
0140  0550                LDK      A5,80                  # OF CHARACTERS
0142  8612                LDR      A6,A12                 LOAD ADDR.
                 *                                CHECK IF COUNT DONE
0144  9048                IM       CNTFLG-IPL,A10
0146  0048
0148  890E                ABR(1)   A11
014A  0F42                AB       /42                    START BOOT ABAIN
      014C       CNTFLG   EQU      *                      - NUMBER OF RECORDS OF HIGH CORE IPL
014C  FFF5                DATA     1-NBREC
014E  0000                DATA     0
0150  0000                DATA     0
0152  0000                DATA     0
0154  0000                DATA     0
0156  0000                DATA     0
0158  0000                DATA     0
015A  0000                DATA     0
015C  0000                DATA     0
015E  0000                DATA     0
0160  0000                DATA     0
0162  0000                DATA     0
0164  0000                DATA     0
0166  0000                DATA     0
```

```
00096   0168   0000          DATA    0
00097   016A   0000          DATA    0
00098   016C   0000          DATA    0
00099   016E   0000          DATA    0
00100   0170   0000          DATA    0
00101   0172   0000          DATA    0
00102   0174   0000          DATA    0
00103   0176   0000          DATA    0
00104   0178   0000          DATA    0
00105   017A   0000          DATA    0
00106   017C   0000          DATA    0
00107   017E   0000          DATA    0
00108   0180   0000          DATA    0
00109   0182   0000          DATA    0
00110   0184   0000          DATA    0
00111   0186   0000          DATA    0
00112   0188   0000          DATA    0
00113   018A   0000          DATA    0
00114   018C   0000          DATA    0
00115   018E   0000          DATA    0
00116   0190   0000          DATA    0
00117   0192   0000          DATA    0
00118   0194   0000          DATA    0
00119   0196   0000          DATA    0
00120   0198   0000          DATA    0
00121   019A   0000          DATA    0
00122   019C   0000          DATA    0
00123   019E   0000          DATA    0
00124   01A0   0000          DATA    0
00125   01A2   0000          DATA    0
00126   01A4   0000          DATA    0
00127   01A6   0000          DATA    0
00128   01A8   0000          DATA    0
00129   01AA   0000          DATA    0
00130   01AC   0000          DATA    0
00131   01AE   0000          DATA    0
00132   01B0   0000          DATA    0
00133   01B2   0000          DATA    0
00134   01B4   0000          DATA    0
00135   01B6   0000          DATA    0
00136   01B8   0000          DATA    0
00137   01BA   0000          DATA    0
00138   01BC   0000          DATA    0
00139   01BE   0000          DATA    0
00140   01C0   0000          DATA    0
00141   01C2   0000          DATA    0
00142   01C4   0000          DATA    0
00143   01C6   0000          DATA    0
00144   01C8   0000          DATA    0
00145   01CA   0000          DATA    0
00146   01CC   0000          DATA    0
00147   01CE   0000          DATA    0
00148   01D0   0000          DATA    0
00149   01D2   0000          DATA    0
00150   01D4   0000          DATA    0
00151   01D6   0000          DATA    0
00152   01D8   0000          DATA    0
00153   01DA   0000          DATA    0
```

```
00154  01DC  0000                 DATA     0
00155  01DE  0000                 DATA     0
00156  01E0  0000                 DATA     0
00157  01E2  0000                 DATA     0
00158  01E4  0000                 DATA     0
00159  01E6  0000                 DATA     0
00160  01E8  0000                 DATA     0
00161  01EA  0000                 DATA     0
00162  01EC  0000                 DATA     0
00163  01EE  0000                 DATA     0
00164  01F0  0000                 DATA     0
00165  01F2  0000                 DATA     0
00166  01F4  0000                 DATA     0
00167  01F6  0000                 DATA     0
00168  01F8  0000                 DATA     0
00169  01FA  0000                 DATA     0
00170  01FC  0000                 DATA     0
00171  01FE  0000                 DATA     0
00172  0200  0000                 DATA     0
00173  0202  0000                 DATA     0
00174  0204  0000                 DATA     0
00175                             EJECT
00176                                                          HIGH CORE IPL
00177        0000       PTR       EQU      0                   DEVICE ADDR WILL BE INITIALIZED BY HCIPL
00178        0010       ASR       EQU      /10
00179        0001       S         EQU      1
00180        0000       H         EQU      0
00181        0001       BOM       EQU      1
00182        0000       SA        EQU      0
00183                   *
00184        0000       MODE      EQU      SA                  FOR MESSAGE  OUTPUT
00185                             RORG     BIPL+80             1ST RECORD OF HIGH CORE IPL
00186        0150       HCIPL1    EQU      *                   1ST WORD OF THE CURRENT RECORD
00187  0150  FEFE                 DATA     /FEFE
00188        0152       HCIPL     EQU      *
00189  0152  85A0                 LDKL     A13,OUTMSG-HCIPL
       0154  00FA
00190  0156  958F                 ADR      A13,A11             LOAD A13 WITH ADDR OF OUTMSG ROUTINE
00191  0158  868F                 LDR      A14,A11
00192  015A  96A0                 ADKL     A14,CEZON-HCIPL
       015C  00F8
00193                   *                  ADDR OF STACK
00194  015E  871E                 LDR      A7,A15              GET DEVICE ADDR
00195  0160  87CF                 ST       A15,SAVA15-HCIPL,A11  SAVE  4*4  FLAG
       0162  0354
00196  0164  5606                 RF(6)    *+8
00197  0166  904F                 IM       COREND+2-HCIPL,A11
       0168  0358
00198  016A  5C06                 RB(4)    *-4
00199  016C  273F                 ANK      A7,/3F
00200                   *                       INIT IO INSTRUCTIONS
00201  016E  974F                 ADS      A7,CTOPTR-HCIPL,A11
       0170  0142
00202  0172  974F                 ADS      A7,INP2-HCIPL,A11
       0174  0188
00203  0176  974F                 ADS      A7,END2-HCIPL,A11
       0178  01CE
```

```
00204  017A  974F            ADS     A7,END1-HCIPL,A11
       017C  0208
00205  017E  974F            ADS     A7,INR44-HCIPL,A11
       0180  01A2
00206  0182  811F            LDR     A1,A15
00207  0184  39C8            SLC     A1,8                       TEST MULTI-SINGLE DEVICE CONT.
00208  0186  5602            RF(6)   HCIPL2
00209  0188  270F            ANK     A7,/F                      MULTI DEVICE CONTROLLER
00210        018A    HCIPL2  EQU     *
00211  018A  974F            AD,S    A7,SSTMLX-HCIPL,A11
       018C  014E
00212  018E  974F            ADS     A7,SSTPR2-HCIPL,A11
       0190  018C
00213  0192  974F            ADS     A7,SSTPTR-HCIPL,A11
       0194  01D0
00214  0196  974F            ADS     A7,SSTPR1-HCIPL,A11
       0198  020A
00215  019A  3F41            SLL     A7,1
00216  019C  974F            AD,S    A7,WER1-HCIPL,A11          INIT WER/RER INST
       019E  013A
00217  01A0  974F            AD,S    A7,WER2-HCIPL,A11
       01A2  013C
00218  01A4  974F            AD,S    A7,RER-HCIPL,A11
       01A6  017C
00219  01A8  57AC            RF      CKOK
00220  01AA  4F42    SYSMSG  DATA    'OBJCT TAPE ON RE'
       01AC  4A43
       01AE  5420
       01B0  5441
       01B2  5045
       01B4  204F
       01B6  4E20
       01B8  5245
00221  01BA  4144            DATA    'ADER, THINK OF B'
       01BC  4552
       01BE  2E20
       01C0  5448
       01C2  494E
       01C4  4B20
       01C6  4F46
       01C8  2042
00222  01CA  4153            DATA    'ASE !'
       01CC  4520
       01CE  2120
00223  01D0  0D0A            DATA    /0D0A
00224        01D2    STAD    EQU     *
00225  01D2  4543    ECMSG   DATA    'EC'
00226  01D4  0D0A            DATA    X'0D0A'
00227  01D6  4F56    OFLMSG  DATA    'OVFR'
       01D8  4652
00228  01DA  0D0A            DATA    /0D0A
00229  01DC          BUFF    RES     39
00230  022A  FFFF            DATA    /FFFF
00231  022C  0000    MASTFG  DATA    0
00232  022E  0000    SAVBAS  DATA    0
00233  0230                  RES     6                          * STACK  AREA
00234        023A    CFZON   EQU     *-2
00235        023C    OUTMSG  EQU     *
```

```
023C  0304                    LDK       A3,4
023E  43D0                    CIO       A3,S,ASR
0240  E324                    LCR       A3,A1
0242  4310          OTR       OTR       A3,0,ASR
0244  5C04                    RB(4)     *-2
0246  1101                    ADK       A1,1
0248  1A01                    SUK       A2,1
024A  5C0C                    RB(4)     OTR-2
024C  0300                    LDK       A3,0
024E  4390                    CIO       A3,H,ASR
0250  48D0                    SST       A3,ASR
0252  5C04                    RB(4)     *-2
0254  F03A                    RTN       A14
                    *
                    *
      0256          CKOK      EQU       *
0256  81A0                    LDK.L     A9,0                     SET LOADING ADDRESS
0258  0000
                    *
                    *
                    *                                  GO LOAD SYSTEM
025A  84A0                    LDK.L     A12,MNLD-HCIPL
025C  029C
025E  948E                    ADR       A12,A11
0260  F693                    CFR       A14,A12
                    *
                    *
0262  810A                    LDR       A1,A10                   TEST START ADDRESS
0264  8C04                    ABR(4)    A1                       EOS  OR EOF HAS BEEN READ
0266  207F                    HLT                                NO START ADDRESS
0268  0000          BADDR     DATA      0
026A  0300          RAFL      LDK       A3,0
026C  0700                    LDK       A7,0
026E  8520                    LDK.L     A5,BUFF-HCIPL
0270  008A
0272  950E                    ADR       A5,A11
0274  80CE                    LD        A8,SAVA15-HCIPL,A11
0276  0354
0278  5610                    RF(6)     INPA
027A  0604                    LDK       A6,4                     4*4 SO ASR
027C  0111                    LDK       A1,/11                   SEND X-ON
027E  46D0                    CIO       A6,S,ASR
0280  4110                    OTR       A1,0,ASR
0282  5C04                    RB(4)     *-2
0284  4290                    CIO       A2,H,ASR
0286  4AD0                    SST       A2,ASR
0288  5C04                    RB(4)     *-2
      028A          INPA      EQU       *
028A  0650                    LDK       A6,80                    LENGTH
028C  7600          WER1      WER       A6,0                     INIT MLX WHATEVER
028E  7501          WER2      WER       A5,1                     CHANNEL IS
0290  8602                    LDR       A6,A8
0292  3E68                    SRL       A6,8
      0294          CIOPTR    EQU       *
0294  46C0                    CIO       A6,S,PTR                 AVAILABLE ON THE PAPER READER
0296  5C04                    RB(4)     *-2
0298  824E                    LD        A2,SAVA15-HCIPL,A11
029A  0354
```

```
029C  3AC3           SLC     A2,3             MLX ?
029E  523A           RF(2)   INP2             NO
      02A0  SSTMLX   EQU     *
02A0  49C0           SST     A1,PTR
02A2  5C04           RB(4)   *-2
      02A4  TMTEST   EQU     *
02A4  A120           ANKL    A1,/1007         TAPE MARK OR FATAL ERROR ?
02A6  1007
02A8  501A           RF(0)   MLX1             NO
02AA  2107           ANK     A1,/7            FATAL ERROR ?
02AC  5486           RF(4)   ECCLS            YES
            *
02AE  8120           LDK.L   A1,':E'          PUT EOF IN BUFF
02B0  3A45
02B2  8135           STR     A1,A5
02B4  8120           LDK.L   A1,'OF'
02B6  4F46
02B8  8155           ST      A1,2,A5
02BA  0002
02BC  0704           LDK     A7,4             UPDATA POINTRERS
02BE  951C           ADR     A5,A7
02C0  579E           RF      PRASCI
            *
02C2  5F5A  RAFL3    RB      RAFL             RELAY TOWARDS RAFL
            *
      02C4  MLX1     EQU     *
02C4  E134           LCR     A1,A5
02C6  F920           CWK     A1,/18           IS IT ASCII
02C8  0018
02CA  55EA           RF(5)   RELAY            NO
02CC  0750           LDK     A7,80            YES
02CE  7E00  RER      RER     A6,0             READ REMAINING LENGTH
02D0  A620           ANK.L   A6,/FFF
02D2  0FFF
02D4  9F18           SUR     A7,A6            COMPUTE TRANSMITTED LGT
02D6  951C           ADR     A5,A7            UPDATE BUFF POINTER
02D8  5786           RF      PRASCI
02DA  4A00  INP2     INR     A2,0,PTR
02DC  500A           RF(0)   SWITCH           INR HAS BEEN ACCEPTED ;
            *                                 PROCESS THE CHARTER
02DE  4AC0  SSTPR2   SST     A2,PTR           TRY A SST
02E0  5C08           RB(4)   INP2             TRY AGAIN INR WHEN SST REFUSED
02E2  8108           LDR     A1,A2
02E4  1300           ADK     A3,0             CHECK WETHER ASCII OR OBJECT
02E6  5278           RF(2)   PRASCI           RECORD WAS ASCII ; PRINT IT
02E8  0600           LDK     A6,0             STORE A NON ASCII CHARACTER AT
02EA  8635           STR     A6,A5            THE END OF BUFFER ,
            *                                 TO BE USED IN MX1
02EC  5F4A           RB      TMTEST
      02EE  OBJINP   EQU     *
02EE  8082           LDR     A8,A8
02F0  560C           RF(6)   EIGHT            8-8 DO NOT INPUT 2ND CHARACTER
02F2  220F           ANK     A2,/F
02F4  4E00  INR44    INR     A6,0,PTR
02F6  5C04           RB(4)   *-2
02F8  260F           ANK     A6,/F
02FA  3A44           SLL     A2,4
```

```
02FC  9218           ADR     A2,A6                JOIN THE TWO HALF CHARACTERS
      02FE  EIGHT    EQU     *
02FE  FF04           CWR     A7,A1
0300  501E           RF(0)   END2
0302  E235           SCR     A2,A5
0304  B408           XRR     A4,A2
0306  1501           ADK     A5,1
0308  EF20           CWK     A7,1
030A  0001
030C  5408           RF(4)   OBJIN1
030E  0400           LDK     A4,0
0310  8108           LDR     A1,A2
0312  9108           ADR     A1,A2
0314  1103           ADK     A1,3
0316  1701  OBJIN1   ADK     A7,1
0318  5F40  INP2R    RB      INP2
031A  2207  FIRST    ANK     A2,/7
031C  0150           LDK     A1,80
031E  5F22           RB(7)   EIGHT
0320  4280  END2     CIO     A2,H,PTR
      0322  SSTPTR   EQU     *
0322  49C0           SST     A1,PTR
0324  5C04           RB(4)   *-2
0326  8082           LDR     A8,A8                IF ASR WAIT
0328  5606           RF(6)   *+8
032A  904F           IM      COREND+2-HCIPL,A11   WAIT
032C  0358
032E  5C06           RB(4)   *-4
0330  24FF           ANK     A4,/FF
0332  5004           RF(0)   PROLO1+2
      0334  ECCLS    EQU     *
0334  0180           LDK     A1,ECMSG-HCIPL
0336  910F           ADR     A1,A11
0338  0204           LDK     A2,4
033A  84A0           LDK.L   A12,OUTMSG-HCIPL
033C  00FA
033E  948F           ADR     A12,A11
0340  F693           CFR     A14,A12
0342  207F  STOP     HLT
0344  5FDC           RB(7)   RAFL
0346  FA20  ASCINP   CWK     A2,/0D
0348  000D
034A  500E           RF(0)   END1
034C  EF20           CWK     A7,68
034E  0044
0350  5878           RB(0)   INP2
0352  E235           SCR     A2,A5
0354  1501           ADK     A5,1
0356  1701           ADK     A7,1
0358  5F80           RB(7)   INP2
035A  4280  END1     CIO     A2,H,PTR
      035C  SSTPR1   EQU     *
035C  4AC0           SST     A2,PTR
035E  5C04           RB(4)   *-2
      0360  PRASCI   EQU     *
0360  018A           LDK     A1,BUFF-HCIPL
0362  910F           ADR     A1,A11
0364  821C           LDR     A2,A7
0366  0320           LDK     A3,/20
```

```
00394  0368  E335            SCR     A3,A5
00395  036A  1501            ADK     A5,1
00396  036C  8320            LDK.L   A3,/0D0A
       036E  0D0A
00397  0370  8335            STR     A3,A5
00398  0372  1203            ADK     A2,3
00399  0374  3A61            SRL     A2,1
00400  0376  3A41            SLL     A2,1
00401  0378  8082            LDR     A8,A8                  IF ASR DO NOT OUTPUT ASCII
00402  037A  5608            RF(6)   PRASCA                 A SECOND TIME (ALREADY PRINTED WHEN READ
00403  037C  904F            IM      COREND+2-HCIPL,A11     BUT WAIT ABIT
       037E  0358
00404  0380  5C06            RB(4)   *-4
00405  0382  5702            RF      PRASCB
00406        0384  PRASCA    EQU     *
00407  0384  1100            ADK     A1,0                   PASS WAS CFR A14,A13 IN IPL52S
00408              * TO AVOID PRINT MESSAGE
00409        0386  PRASCB    EQU     *
00410  0386  028A            LDK     A2,BUFF-HCIPL
00411  0388  920E            ADR     A2,A11
00412  038A  8328            LDR*    A3,A2
00413  038C  EB20            CWK     A3,/3A45               * 1E
       038E  3A45
00414  0390  5CD0            RB(4)   RAFL3
00415  0392  1202            ADK     A2,2
00416  0394  8328            LDR*    A3,A2
00417  0396  EB20            CWK     A3,/4F46               *OF
       0398  4F46
00418  039A  5CDA            RB(4)   RAFL3
00419              *
00420              *
00421              *                              SET RETURN PARAMETERS
00422              *
00423  039C  82CE            LD      A10,COREND-HCIPL,A11   START ADDRESS IN A10
       039E  0356
00424  03A0  84A0            LDK.L   A12,MNLD-HCIPL
       03A2  029C
00425  03A4  948E            ADR     A12,A11                MNLD ADDRESS  IN A12
00426  03A6  81CE            LD      A9,SAVBAS-HCIPL,A11    LOADING BASE  IN A9
       03A8  00DC
00427  03AA  80CE            LD      A8,BADDR-HCIPL,A11     ENDING ADDRESS IN A8
       03AC  0116
00428  03AE  874E            LD      A7,MASTFG-HCIPL,A11    MASTER FLAG IN A7
       03B0  00DA
00429              *
00430              *
00431  03B2  F03A            RTN     A14                    RETURN TO CALLING
00432              *
00433  03B4  5FF4  RAFL2     RB      RAFL3                  * RELAY TOWARDS RAFL
00434  03B6  5750  RELAY     RF      PROLO1+2
00435              *
00436  03B8  1300  SWITCH    ADK     A3,0
00437  03BA  59CE            RB(1)   OBJINP
00438  03BC  227F            ANK     A2,/7F
00439  03BE  58E6            RB(0)   INP2
00440  03C0  EA20            CWK     A2,/7F
       03C2  007F
00441  03C4  58EC            RB(0)   INP2
```

```
03C6  1300                ADK       A3,0
03C8  5A84                RB(2)     ASCINP
03CA  EA20                CWK       A2,/1F
03CC  001F
03CE  5116                RF(1)     ASCII
03D0  EA20                CWK       A2,/14
03D2  0014
03D4  510C                RF(1)     OBJEC
03D6  EA20                CWK       A2,8
03D8  0008
03DA  5206                RF(2)     OBJEC
03DC  EA20                CWK       A2,/10
03DE  0010
03E0  5CCA                RB(4)     INP2R
03E2  1301      OBJEC     ADK       A3,1
03E4  5FCC                RB(7)     FIRST
03E6  1B01      ASCII     SUK       A3,1
03E8  5FA4                RB(7)     ASCINP
                          EJECT
                *
                *
                *  MAIN   LOADING   PART            ENTRY PARAMETERS :
                *
                *                                     A1 = MASTER FLAG
                *                                     A9 = BASE ADDRESS
                *
                *
                *
                *                   1- OUTPUT MSG REQUESTING  TAPE ON READER
                *                   2- PERFORMS A 'HALT' - THE USER HAS THEN THE
                *                      POSSIBILITY TO ALTER THE BASE ADDRESS (REG,A9)
                *                                     AND  THE MASTER FLAG  (REG,A1)
                *                                                  (A1=0 MASTER)
                *                                                  (A1=1 USER)
                *                   3- LOADING PROCESS STARTS  WHEN USER DEPPRESS START
                *                      BUTTON
                *
                *
                *
                *
      03FA      RAFL1     EQU       *
03FA  5F38                RB        RAFL2
                *
                *
                *                                  ERRONNOUS CLUSTER,PRINT ERR MESSAGE
      03EC      CLC01     EQU       *
03FC  5FBA                RB        ECCLS
                *
                *
                *
                *
      03FF      MNLD      EQU       *
03EE  8404                LDR       A4,A1                 SAVE  MASTER FLAG
03F0  0200                LDK       A2,0
03F2  824F                ST        A2,COREND-HCIPL,A11   RAZ  START  ADDRESS
03F4  0356
                          IFT       MODE=BOM
                          XIF
```

```
03F6  81CF             ST        A9,SAVBAS-HCIPL,A11
03F8  00DC
                 *
                 *
                 *
                 * PROCESS LOADING : THIS  MODULE READ A  CLUSTER
                 *         AND  BRANCH ACCORDING TO THE CLUSTER TYPE
                 *         ON EXIT   A1 = BUFF  ADDRESS +1
                 *                   A2 = WORD  COUNT
                 *                   A3 = TYPE
                 *         THE TYPE MUST BE    3,4,7   IF NOT THIS :HALT
03FA  82A0       PROLO   LDK.L     A10,STAD-HCIPL        END ADDRESS
03FC  0080
03FE  928F               ADR       A10,A11
      0000       ABA     EQU       0
0400  81CF               ST        A9,BADDR-HCIPL,A11    BADDR =BASE ADDRESS
0402  0116
0404  20BF       PROGLD  INH
      0406       PROLO1  EQU       *
0406  5F1E               RB        RAFL1
0408  8120               LDK.L     A1,BUFF-HCIPL
040A  008A
040C  910F               ADR       A1,A11
040E  0401               LDK       A4,1
0410  0300               LDK       A3,0
0412  E324               LCR       A3,A1                 A3 = TYPE
0414  1101               ADK       A1,1
0416  0200               LDK       A2,0
0418  E224               LCR       A2,A1                 A2 = WORD COUNT
041A  1101               ADK       A1,1
041C  EB20               CWK       A3,3
041E  0003
0420  500F               RF(0)     CLCODE                BRANCH  ON CLUSTER CODE
0422  EB20               CWK       A3,4
0424  0004
0426  503A               RF(0)     CLIMOD                INTERNAL MODIFICATION
0428  EB20               CWK       A3,7
042A  0007
042C  505E               RF(0)     CLEND                 END/START
042E  5F2A               RB(7)     PROLO1
                 ***********************************
                 * CLUSTER CODE  TYPE 3
                 *       UPON ENTRY: A1=ADDRESS OF BUFF+1 (RBK
                 *                   A2=WORD COUNT
                 *                   A9=BADDRESS
                 *                   A10=ENDADDRESS
                 ******************************************
0430  834E       CLCODE  LD        A3,BUFF+6-HCIPL,A11
0432  0090
0434  5C30               RB(4)     PROLO1                EMBK SET SKIP THE CLUSTER
0436  834F       CLCO1A  LD        A3,BUFF+4-HCIPL,A11
0438  008E
043A  A311               TM        A3,A4                 IS IT  RELOCATABLE SECTION
043C  5004               RF(0)     CLCO4
043E  3301               XRK       A3,1
0440  9306               ADR       A3,A9
0442  8524       CLCO4   LDR*      A5,A1                 A5=(RBK)
0444  1106               ADK       A1,6                  A1= ADDRESS OF ST CODE WORD IN BUFF
```

```
0446  1A03              SUK     A2,3                    A2= NUMBER OF CODE WORD
                  *                           A3= STORAGE ADDRESS
                  *                           A4= MASK FOR RBK
                  *                           A6= CODE WORD
0448  3CE1        CLC05   SRC     A4,1
044A  8624                LDR*    A6,A1
044C  EB0A                CWR     A3,A10                COMPARE LOAD ADDRESS WITH AD OF IPL
044E  5864                RB(0)   CLC01
0450  A511                TM      A5,A4
0452  5002                RF(0)   CLC07
0454  9606                ADR     A6,A9
0456  862D        CLC07   STR     A6,A3                 STORE CODE W=RDS
0458  1102                ADK     A1,2
045A  1302                ADK     A3,2
045C  1A01                SUK     A2,1
045E  5C18                RB(4)   CLC05
0460  5F68                RB(7)   PROLO
                  *
                  ***************************************************************
                  * INTERNAL MODIFICATION CLUSTER
                  ************************************************
0462  0701        CLIMOD  LDK     A7,1                  A7= MASK FOR ADDRESS
0464  8524                LDR*    A5,A1                 A5=(RBK)
0466  1A01                SUK     A2,1
0468  3CE1        CLIM1   SRC     A4,1
046A  1102                ADK     A1,2
046C  8324                LDR*    A3,A1                 A3=ADDRESS
046E  A31D                TM      A3,A7                 IS IT RELOCATABLE
0470  5008                RF(0)   CLIM2                 NO
0472  3301                XRK     A3,1
0474  9306                ADR     A3,A9                 YES AD BASE
0476  EB0A                CWR     A3,A10                ADDRESS OK
0478  588E                RB(0)   CLC01
047A  1102        CLIM2   ADK     A1,2
047C  8624                LDR*    A6,A1                 TAKE CODE WORD
047E  A511                TM      A5,A4                 IS IT RELOCATABLE
0480  5002                RF(0)   CLIM3
0482  9606                ADR     A6,A9                 AD BASE
0484  862D        CLIM3   STR     A6,A3                 STORE CODE WORD
0486  1A02                SUK     A2,2
0488  5C22                RB(4)   CLIM1
048A  5F92                RB(7)   PROLO
                  ****************************************
                  *  CLUSTER  END/START
                  *****************************************
048C  8324        CLEND   LDR*    A3,A1
048E  500A                RF(0)   CLEN3A                FINISH NO START
0490  A311                TM      A3,A4
0492  5002                RF(0)   CLEN1
0494  9306                ADR     A3,A9
                  *
                  *
      0496        CLEN1   EQU     *
0496  834F                ST      A3,COREND-HCIPL,A11
0498  0356
049A  814F        CLEN3A  LD      A1,BUFF+6-HCIPL,A11   UPDATE BASE ADDRESS
049C  0090
```

```
049E  914F          AD.S     A1,BADDR=HCIPL,A11
04A0  0116
04A2  9184          ADR      A9,A1
04A4  5FAC          RB       PROLO
                    *
                    *
04A6  0000  SAVA15  DATA     0
                    *
                    ************************************************************************
                    *
04A8  0000  COREND  DATA     0
                    *
                    *
                    ************************************************************************
04AA  0000          DATA     0
04AC  0000          DATA     0
04AE  0000          DATA     0
04B0  0000          DATA     0
04B2  0000          DATA     0
04B4  0000          DATA     0
04B6  0000          DATA     0
04B8  0000          DATA     0
04BA  0000          DATA     0
04BC  0000          DATA     0
04BE  0000          DATA     0
04C0  0000          DATA     0
04C2  0000          DATA     0
04C4  0000          DATA     0
04C6  0000          DATA     0
04C8  0000          DATA     0
04CA  0000          DATA     0
04CC  0000          DATA     0
04CE  0000          DATA     0
04D0  0000          DATA     0
04D2  0000          DATA     0
04D4  0000          DATA     0
04D6  0000          DATA     0
04D8  0000          DATA     0
04DA  0000          DATA     0
04DC  0000          DATA     0
04DE  0000          DATA     0
04E0  0000          DATA     0
04E2  0000          DATA     0
04E4  0000          DATA     0
04E6  0000          DATA     0
04E8  0000          DATA     0
04EA  0000          DATA     0
04EC  0000          DATA     0
04EE  0000          DATA     0
04F0  0000          DATA     0
04F2  0000          DATA     0
04F4  0000          DATA     0
04F6  0000          DATA     0
04F8  0000          DATA     0
04FA  0000          DATA     0
04FC  0000          DATA     0
04FE  0000          DATA     0
0500  0000          DATA     0
```

```
0502  0000                DATA    0
0504  0000                DATA    0
0506  0000                DATA    0
0508  0000                DATA    0
050A  0000                DATA    0
050C  0000                DATA    0
050E  0000                DATA    0
0510  0000                DATA    0
0512  0000                DATA    0
0514  0000                DATA    0
0516  0000                DATA    0
0518  0000                DATA    0
051A  0000                DATA    0
051C  0000                DATA    0
051E  0000                DATA    0
0520  0000                DATA    0
                          EJECT
      0522        IPL44   EQU     *
0522  030C                LDK     A3,NBREC
0524  8420                LDKL    A4,BIPL
0526  0100  R
0528  0785                LDK     A7,/85
                  * BASIC WRITE
052A  80A0                LDKL    A8,DECB44
052C  0582  R
      052E        PUNCH4  EQU     *
052E  0550                LDK     A5,80
0530  8620                LDKL    A6,BUF44
0532  058E  R
      0534        PCH400  EQU     *
0534  0100                LDK     A1,0
0536  0200                LDK     A2,0
0538  E130                LCR     A1,A4
053A  3964                SRL     A1,4
053C  E921                CCK     A1,0
053E  0000
0540  5006                RF(0)   PCH405
0542  E921                CCK     A1,/500
0544  0500
0546  5202                RF(2)   PCH410             LESS THAN 5
      0548        PCH405  EQU     *
0548  2910                ORK     A1,/10
      054A        PCH410  EQU     *
054A  3948                SLL     A1,8
054C  E130                LCR     A1,A4
054E  A120                ANKL    A1,/FF0F
0550  FF0F
0552  E921                CCK     A1,0
0554  0000
0556  5006                RF(0)   PCH415
0558  E921                CCK     A1,/500
055A  0500
055C  5202                RF(2)   PCH420
      055E        PCH415  EQU     *
055E  2910                ORK     A1,/10
      0560        PCH420  EQU     *
0560  8139                STR     A1,A6
                  * STORE WORD
```

```
0562  1602            ADK     A6,2              NEXT WORD IN BUFFER
0564  1401            ADK     A4,1              NEXT CHZR. IN IPL
0566  1D01            SUK     A5,1
                *
0568  5936            RB(1)   PCH400
056A  FB20            CWK     A3,1              LAST RECORD ?
056C  0001
056E  5406            RF(4)   PCH430            NO
0570  0113            LDK     A1,/13            XOFF
0572  8141            ST      A1,BUF44+158
0574  062C  R
      0576      PCH430 EQU    *
0576  2804            LKM
0578  0001            DATA    1
057A  1B01            SUK     A3,1
057C  5950            RB(1)   PUNCH4
                *
057E  2804            LKM
0580  0003            DATA    3
                *
      0582      DECB44 EQU    *
0582  0003            DATA    3
0584  058E  R         DATA    BUF44
0586  00A0            DATA    160
0588  0000            DATA    0
058A  0000            DATA    0
058C  0000            DATA    0
058E            BUF44  RES    80
                *
                *
                *
                      END     BIPLPR
```

SYMBOL TABLE

ABA	0000	A	ASCII	03E6	R	ASCINP	0346	R	ASR	0010	A
BADDR	0268	R	BIPL	0100	R	BIPLPR	0000	R	BOM	0001	A
BUF44	058E	R	BUFF	01DC	R	CFZON	023A	R	CIOPTR	0294	R
CKOK	0256	R	CLC01	03EC	R	CLC01A	0436	R	CLC04	0442	R
CLC05	0448	R	CLC07	0456	R	CLCODE	0430	R	CLEN1	0496	R
CLEN3A	049A	R	CLEND	048C	R	CLIM1	0468	R	CLIM2	047A	R
CLIM3	0484	R	CLIMOD	0462	R	CNTFLG	014C	R	COREND	04A8	R
DECB	001E	R	DECB44	0582	R	DECBUF	0020	R	ECCLS	0334	R
ECMSG	01D2	R	EIGHT	02FE	R	END1	035A	R	END2	0320	R
FIRST	031A	R	H	0000	A	HCIPL	0152	R	HCIPL1	0150	R
HCIPL2	018A	R	INP2	02DA	R	INP2R	0318	R	INPA	028A	R
INR44	02F4	R	IPL	0104	R	IPL010	0116	R	IPL020	0122	R
IPL100	012C	R	IPL110	0130	R	IPL44	0522	R	IPL88	0000	R
LDFLG	0100	R	MASTFG	022C	R	MLX1	02C4	R	MNLD	03EE	R
MODE	0000	A	NBREC	000C	A	OBJEC	03E2	R	OBJIN1	0316	R
OBJINP	02EE	R	OFLMSG	01D6	R	OTR	0242	R	OUTMSG	023C	R
PCH400	0534	R	PCH405	0548	R	PCH410	054A	R	PCH415	055F	R
PCH420	0560	R	PCH430	0576	R	PRASCA	0384	R	PRASCB	0386	R
PRASCI	0360	R	PROGLD	0404	R	PROLO	03FA	R	PROLO1	0406	R
PTR	0000	A	PUNCH	000C	R	PUNCH4	052E	R	RAFL	026A	R
RAFL1	03FA	R	RAFL2	03B4	R	RAFL3	02C2	R	RELAY	03B6	R
RER	02CE	R	S	0001	A	SA	0000	A	SAVA15	04A6	R
SAVBAS	022E	R	SSTMLX	02A0	R	SSTPR1	035C	R	SSTPR2	02DE	R
SSTPTR	0322	R	STAD	01D2	R	STOP	0342	R	SWITCH	03B8	R
SYSMSG	01AA	R	TMTEST	02A4	R	WER1	028C	R	WER2	028E	R

3.54 SIMPLE TEST PROGRAMS

The Simple Test programs that follow this paragraph are all Stand Alone programs that can be loaded into memory direct from the Control Panel switches. They should be tried if the Standard Test Programs either cannot be loaded or will not run. To avoid errors which may be introduced when loading the programs by hand it is suggested that once the program has been loaded and tested it should be reproduced on a suitable input media so that for future programs on an input media, together with suggestions for producing more sophisticated programs, will be found in paragraphs 3-64 to 3-74 of this section.

3.55 Memory Test Program MEMHAN

If the Standard Memory Test Program either cannot be used, or if you wish to test only a certain area of memory with a particular test pattern, try the following program which can be loaded either by hand from the Control Panel switches or by the IPL routine if it has been reproduced on a suitable input media. If the program is loaded from the Control Panel switches it can be loaded into any area of memory, but if it is loaded by the IPL routine the addresses of the instructions will be as shown in the program listing. The program loads the Test Pattern into A12. When all the required locations have been loaded, the program reads each location and compares each with the data in register A13. If an error is found during the read part of the program it will stop and the address where the error has occurred will be found in register A9, the pattern read will be found in register A8, and the expected pattern will be in register A13. If you wish the program to continue after an error has been detected press the RUN button. If there are no errors the program will run until the contents of A7 are zero.

3.56 Program CHECK

This program enables you to check the type of input from the Operator's device i.e. 8-bit data no parity, 7-bit data with parity etc. The program can be loaded by hand with the control panel switches or by using the IPL routine. Once loaded it is only necessary to load the start address into register A0 and then push the RUN button. You can now type in any ten characters from the keyboard which are stored in the Buffer (address 00A8). When the buffer is full the program branches back to the start of program and you can check the codes received for each character using the Read Memory Routine. If you wish to check more characters push the RUN button and repeat as necessary. This program is useful if you wish to check for certain characters in your programs.

3.57 ASR, PER3100, and Display

If the Standard Test Program either cannot be loaded or will not run, one of the following programs should be tried. They can all be loaded either by the Control Panel switches or by the IPL routine. Two of the programs (LINE and NOECHO) have been written for devices that have been wired for Even Parity so if your device is wired for any other mode of operation the parameters loaded into register A2 must be changed accordingly.

3.58 Program LINE

This program can be loaded either from the Control Panel switches or by using the IPL routine. If the program is loaded by hand it can be loaded into any area of memory, but if loaded by the IPL routine the locations of the instructions will be as shown in the program listing. The program enables a selected ASCII character to be sent to the device a selected number of times (the number of times selected MUST NOT exceed the maximum number of characters per line specified for the device being tested), then it sends the Carriage Return/Line Feed characters and branches back to the start of program and stops ready for a new character to be selected. If you wish the program can be modified to run continuously by changing two of the instructions as follows:

- Change the LDR instruction in line 27 to LDK A4,64 hexadecimal value 0440.
- Change the RB instruction value in line 41 so that it branches back to line 24 of the program by using hexadecimal value 5F24.

Once the RUN button has been pushed it will then run and continue repeating the same line until you stop the program by pushing the INST button.

MEMHAM

```
                            IDENT   MEMHAN
              0000   BEGIN  EQU     *
                            RORG    BEGIN+/80
                     *
                     *      THIS PROGRAM ENABLES THE ENGINEER TO WRITE AND CHECK READ
                     *      A SELECTED PATTERN INTO A SELECTED AREA OF MEMORY AND TO
                     *      REPEAT THE TEST A SELECTED NUMBER OF TIMES UP TO 255
                     *
                     *       LOAD THE NUMBER OF TIMES YOU WANT THE TEST REPEATED INTO
                     *      REGISTER A7 FROM THE CONTROL PANEL
                     *
                     *      LOAD THE START ADDRESS OF THE AREA TO BE TESTED INTO
                     *      REGISTER A11 FROM THE CONTROL PANEL
                     *
                     *      LOAD THE END ADDRESS OF THE AREA TO BE TESTED INTO
                     *      REGISTER A12 FROM THE CONTROL PANEL
                     *
                     *      LOAD THE TEST PATTERN INTO REGISTER A13 FROM THE CONTROL PANEL
                     *
                     *      LOAD THE START ADDRESS OF THE PROGRAM INTO REGISTER A0 FROM THE
                     *      CONTROL PANEL THEN PUSH THE RUN BUTTON
                     *
                     *
       0080   FFFF          DATA    /FFFF
       0082   0000          DATA    0
       0084   207F   START  HLT
       0086   818E   WRITE  LDR     A9,A11              LOAD START ADDRESS INTO A9
       0088   85A7   LOAD   STR     A13,A9              STORE TEST PATTEN INTO MEMORY
       008A   91A0          ADKL    A9,2                UPDATE MEMORY ADDRESS COUNTER
       008C   0002
       008E   F992          CWR     A9,A12              CHECK FOR END ADDRESS IN A9
       0090   5D0A          RB(NG)  LOAD                IF NOT EQUAL LOAD NEXT ADDRESS
       0092   818E          LDR     A9,A11              LOAD START ADDRESS INTO A9
       0094   80A6   READ   LDR*    A8,A9               LOAD CONTENTS OF A9'S MEMORY ADDRESS
       0096   F896          CWR     A8,A13              AND COMPARE WITH PATTERN IN A13
       0098   5002          RF(Z)   CONT                IF EQUAL CONTINUE WITH READ ROUTINE
       009A   207F          HLT                         IF NOT EQUAL STOP
       009C   91A0   CONT   ADKL    A9,2                UPDATE ADDRESS COUNTER
       009E   0002
       00A0   F992          CWR     A9,A12              CHECK FOR END ADDRESS IN A9
       00A2   5D10          RB(NG)  READ                IF NOT EQUAL READ NEXT WORD
       00A4   1F01          SUK     A7,1                SUBTRACT 1 FROM REPEAT COUNTER
       00A6   5C22          RB(NZ)  WRITE               AND START NEXT W/R CYCLE IF NOT ZERO
       00A8   5F26          RB      START               WHEN FINISHED GO TO START OF PROGRAM FOR
                     *                                  POSSIBLE NEW PARAMETERS
                            END     START
```

CHECK

```
                               IDENT   CHECK
           0000        BEGIN   EQU     *
                               RORG    BEGIN+/80
                       *
                       *       THIS PROGRAM ENABLES THE ENGINEER TO CHECK
                       *       THE CHARACTER CODE FROM THE OPERATORS DEVICE
                       *
                       *
                       *       LOAD THE PROGRAM START ADDRESS INTO A0 FROM THE
                       *       CONTROL PANEL - THEN PUSH THE RUN BUTTON
                       *
                       *
0080       FFFF                DATA    /FFFF
0082       0000                DATA    0
0084       207F        START   HLT
0086       20BF                INH
                       *
0088       0300                LDK     A3,0             ZEROISE BUFF COUNTER
008A       0201                LDK     A2,1             LOAD PARAMETERS FOR INPUT CHARACTER
                       *                                                     WITH NO
008C       4200                CIO     A2,1,/10         SEND START COMMAND TO DEVICE
008E       5C04                RB(NA)  *-2
0090       4D10        INCHAR  INR     A5,0,/10         SEND INR INSTRUCTION TO DEVICE
0092       5C04                RB(NA)  *-2              AND WAIT FOR CHAR TO BE INPUT
0094       F54D                SC      A5,BUFF,A3       STORE CHAR IN BUFF
0096       00A8   R
0098       1301                ADK     A3,1             UPDATE BUFF COUNTER
009A       EB20                CWK     A3,10            CHECK FOR BUFFER FULL
009C       000A
009E       5C10                RB(NE)  INCHAR           ACCEPT NEW CHAR IF NOT FULL
00A0       4290                CIO     A2,0,/10         STOP INPUT FUNCTION
00A2       4AD0                SST     A2,/10           AND GET STATUS
00A4       5C04                RB(NA)  *-2
00A6       5F24                RB      START
00A8                   BUFF    RES     5
                               END     START
```

LINE

```
                            IDENT   LINE
           0000     BEGIN   EQU     *
                            RORG    BEGIN+/80
                    *
                    *       THIS PROGRAM ENABLES THE ENGINEER TO SEND A SELECTED
                    *       ASCII CHAR TO THE OPERATOR'S DEVICE A SELECTED
                    *       NUMBER OF TIMES
                    *
                    *
                    *       LOAD THE ASCII CHARACTER INTO REGISTER A6 FROM THE
                    *       CONTROL PANEL
                    *
                    *       LOAD THE NUMBER OF TIMES YOU WANT IT REPEATED
                    *       INTO REGISTER A7 FROM THE CONTROL PANEL
                    *
                    *       LOAD THE PROGRAM START ADDRESS INTO REGISTER A0
                    *       FROM THE CONTROL PANEL AND PUSH THE RUN BUTTON
                    *
                    *
    0080   FFFF             DATA    /FFFF
    0082   0000             DATA    0
    0084   207F     START   HLT
    0086   20BF             INH                         THIS PROGRAM RUNS IN INHIBIT MODE
                    *
    0088   0204             LDK     A2,4                LOAD PARAMETERS FOR OUTPUT CHARACTER
                    *                                   WITH EVEN PARITY
    008A   4200             CIO     A2,1,/10            SEND START COMMAND TO DEVICE
    008C   5C04             RB(NA)  *-2                 KEEP TRYING UNTIL ACCEPTED
    008E   841C             LDR     A4,A7               LOAD COUNT INTO A4
    0090   4610     OUCHAR  OTR     A6,0,/10            OUTPUT ASCII CHAR TO DEVICE
    0092   5C04             RB(NA)  *-2                 TRY AGAIN IF NOT ACCEPTED
    0094   1C01             SUK     A4,1                SUBTRACT ONE FROM COUNT
    0096   5C08             RB(NZ)  OUCHAR              SEND CHAR AGAIN IF NOT ZERO
                    *                                   IF ZERO
    0098   8520     CRLF    LDKL    A5,/0A0D            LOAD CR/LF CHARS
    009A   0A0D
    009C   4510     OUCRLF  OTR     A5,0,/10            SENDS CR/LF CHARS
    009E   5C04             RB(NA)  *-2
    00A0   3D68             SRL     A5,8                POSITIONS 0A CHAR FOR SENDING
    00A2   5C08             RB(NZ)  OUCRLF              BRANCHES FIRST TIME TO SEND SECOND CHAR
    00A4   4290             CIO     A2,0,/10            SENDS STOP COMMAND TO DEVICE
    00A6   4AD0             SST     A2,/10              AND THEN GETS STATUS WORD
    00A8   5C04             RB(NA)  *-2
    00AA   5F28             RB      START               RETURN TO START  COUNT DONE
                            END     START
```

3.59 Programs NOEC57 and ECHO57

These programs enable you to verify the input/output functions of the peripheral and the V24 Serial CU and they can be loaded either by hand from the Control Panel or by using the IPL routine. The only differences between the two programs are the parameters loaded into register A2, and with ECHO57 it is the CU that sends the character back to the device but for NOEC57 it is the program. In both programs 7 data bits with even parity are selected so if your CU has been programmed a different type of data (by means of the U-links) you will have to change the parameters in register A2 to aggree with the type of parity and number of data bits being used. Once loaded, put the start address in register A0 and then push the RUN button; thereafter the program will run until stopped by pushing the INST button. Any printable characters typed in via the keyboard will be reproduced on the console log, and any control characters (e.g. CR LF) are accepted and executed.

NOEC57

```
                       IDENT   NOEC57
      0000     BEGIN   EQU     *
                       RORG    BEGIN+/80
               *
               *       THIS PROGRAM ENABLES THE ENGINEER TO VERIFY THE INPUT/OUTPUT
               *       FUNCTIONS OF THE CU AND ITS PERIPHERAL
               *
               *
               *       LOAD THE PROGRAM START ADDRESS INTO A0 FROM THE
               *       CONTROL PANEL - THEN PUSH THE RUN BUTTON
               *
               *
0080  FFFF             DATA    /FFFF
0082  0000             DATA    0
0084  207F     START   HLT
0086  208F             INH
               *
0088  0205     INCHAR  LDK     A2,5            LOAD PARAMETERS FOR INPUT CHARS NO ECHO
               *                               MODE WITH 7 DATA BITS NO PARITY
008A  42D0             CIO     A2,1,/10        SEND START COMMAND TO DEVICE
008C  5C04             RB(NA)  *-2
008E  4D10             INR     A5,0,/10        SEND INR INSTRUCTION TO DEVICE
0090  5C04             RB(NA)  *-2             AND WAIT FOR CHAR TO BE INPUT
0092  4290             CIO     A2,0,/10        STOP INPUT FUNCTION
0094  4AD0             SST     A2,/10          AND GET STATUS
0096  5C04             RB(NA)  *-2
0098  0204             LDK     A2,4            LOAD PARAMETERS FOR OUTPUT CHARACTER
               *                               WITH EVEN PARITY
009A  42D0             CIO     A2,1,/10        SEND START COMMAND TO DEVICE
009C  5C04             RB(NA)  *-2
009E  4510             OTR     A5,0,/10        SEND CHAR BACK TO DEVICE
00A0  5C04             RB(NA)  *-2
00A2  4290             CIO     A2,0,/10        SEND STOP OUTPUT FUNCTION COMMAND
00A4  4AD0             SST     A2,/10          AND GET STATUS
00A6  5C04             RB(NA)  *-2
00A8  5F22             RB      START+4         RETURN TO START OF INPUT ROUTINE
                       END     START
```

ECHO57

```
                              IDENT    ECH057
           0000    BEGIN      EQU      *
                              RORG     BEGIN+/80
                   *
                   *          THIS PROGRAM ENABLES THE ENGINEER TO VERIFY THE INPUT/OUTPUT
                   *          FUNCTIONS OF THE OPERATOR'S PERIPHERAL IN ECHO MODE
                   *
                   *
                   *          LOAD THE PROGRAM START ADDRESS INTO A0 FROM THE
                   *          CONTROL PANEL - THEN PUSH THE RUN BUTTON
                   *
                   *
    0080   FFFF               DATA     /FFFF
    0082   0000               DATA     0
    0084   207F    START      HLT
    0086   20BF               INH
                   *
    0088   0225    INCHAR     LDK      A2,/25                  LOAD PARAMETERS FOR ECHO MODE
                   *                                           WITH 7 DATA BITS NO PARITY
    008A   4200               CIO      A2,1,/10                SEND START COMMAND TO DEVICE
    008C   5C04               RB(NA)   *-2
    008E   4D10               INR      A5,0,/10                SEND INR INSTRUCTION TO DEVICE
    0090   5C04               RB(NA)   *-2                     AND WAIT FOR CHAR TO BE INPUT
    0092   4290               CIO      A2,0,/10                STOP INPUT FUNCTION
    0094   4AD0               SST      A2,/10                  AND GET STATUS
    0096   5C04               RB(NA)   *-2
    0098   5F12               RB       INCHAR
                              END      START
```

3.60 PROGRAMS FOR THE MCU3 CARD

The MCU3 card can control the operation of a Paper Tape Reader, a Paper Tape Punch, and a Serial Device. Parts of the logic are common to all three devices so if one of the suggested programs runs without error you can almost certainly eliminate the common logic and look for the fault in unique logic of the device whose program does not run. No programs are given for the Serial Device. You can check it using the examples given in paragraphs 3-57 to 3-59 modifying them as necessary for the device concerned. The programs to check the PTR and PTP are: PTR57, PUNCH, PCH57, and CHKTRD. The first two can be modified to run continuously by changing the displacement value of the RB to START instruction so that it branches back to the first instruction in the program instead of the HLT instruction at START. The listings of the programs give the instructions for operating them and the function of the programs is:

- PTR57 — will read any type of coded paper tape and the correct operation of the CU logic and the device can be verified by checking the contents of the buffer against the holes punched in the tape.
- PUNCH — will punch a selected area of memory onto tape and you can either load the data into memory yourself (using the Control Panel), or use an area of memory that you know contains data. The correct operation of the CU logic and the device can be verified by checking the tape produced against the data in the area of memory selected.
- PCH57 — will cause the punch mechanism to select every possible combination of punch pins from all holes to blank tape; you can also modify this program to punch a selected punch combination continuously by changing the value loaded into register A5 and replacing the SUK instruction by an RB to CONT instruction (hexadecimal 5F06). The correct operation of the device can be verified either by checking the tape by eye or by using the following program.
- CHKTRD — will check if the tape produced by PCH57 was correctly punched by adding the value of each character read into register A3 and when all the characters have been read checking the value in A3 against a check sum for correct operation. The only precaution necessary in using this program is that the tape must be accurately positioned so that the first character read is the all holes character and not blank tape. If the checksum is correct the program will halt on location /86, if the checksum is wrong the program will halt on location /AA. When a wrong checksum has been detected try the program two or three times more and check if the result in register A3 is the same every time, then check the tape to see if all the combinations have been punched. If not check the punch logic again. If all combinations have been punched check the reader logic.

In principle it is unlikely that either the PTR or PTP will be operated via the IOP channel so no programs have been given for this type of operation. However, if your system uses the IOP for either of these devices you can modify one of the programs suggested for the MCU2 card devices.

PTR57

```
                                IDENT   PTR57
            0000      BEGIN     EQU     *
                                RORG    BEGIN+/80
                      *
                      *         THIS PROGRAM ENABLES THE ENGINEER TO VERIFY THE
                      *         INPUT FUNCTIOON OF THE CU AND ITS DEVICE
                      *
                      *         LOAD NUMBER OF CHARS TO BE READ INTO A7
                      *
                      *         LOAD THE PROGRAM START ADDRESS INTO A0
                      *
                      *         LOAD TAPE ON READER THEN PUSH RUN BUTTON
                      *
                      *
      0080  FFFF                DATA    /FFFF
      0082  0000                DATA    0
      0084  207F      START     HLT
      0086  208F                INH                               THIS PROGRAM RUNS IN INHIBIT MODE
      0088  861C                LDR     A6,A7                     LOAD CHAR COUNT IN A6
      008A  0300                LDK     A3,0                      ZEROISE BUFFER CHAR COUNT
      008C  0201                LDK     A2,1                      LOAD PARAMETER FOR INPUT CHARACTER
      008E  42E0                CIO     A2,1,/20                  SEND START COMMAND TO DEVICE
      0090  5C04                RB(NA)  *-2
      0092  4D20      INPT      INR     A5,0,/20                  INPUT CHAR FROM READER
      0094  5C04                RB(NA)  *-2
      0096  E54D                SC      A5,BUFF,A3                STORE CHAR IN BUFFER
      0098  00A8  R
      009A  1301                ADK     A3,1                      UPDATE BUFFER CHAR COUNT
      009C  1E01                SUK     A6,1                      SUBTRACT ONE FROM CHAR COUNT
      009E  5C0E                RB(NZ)  INPT                      RETURN TO READ ANOTHER CHAR IF NOT ZERO
      00A0  42A0                CIO     A2,0,/20                  SEND STOP COMMAND TO DEVICE
      00A2  4AE0                SST     A2,/20                    AND GET STATUS
      00A4  5C04                RB(NA)  *-2
      00A6  5F24                RB      START                     GO TO START OF PROGRAM
      00A8            BUFF      RES     40
                                END     START

            SYMBOL TABLE

  BEGIN    0000  R   BUFF    00A8  R   INPT    0092  R   START   0084  R
```

PUNCH

```
                     IDENT    PUNCH
                *
                *    THIS PROGRAM PUNCHES A SELECTED AREA OF MEMORY ONTO PAPER TAPE
                *
      0000      BEGIN   EQU      *
                     RORG     BEGIN+/80
                *
                *
                *    LOAD THE FIRST ADDRESS OF THE AREA INTO REGISTER A12
                *
                *    LOAD THE LAST ADDRESS OF THE AREA INTO REGISTER A11
                *
                *    LOAD THE START ADDRESS OF THE PROGRAM INTO REGISTER A0
                *
                *    RUN THE RUN BUTTON
                *
                *
0080  207F      START   HLT
0082  8112              LDR      A1,A12             SAVE THE START ADDRESS INTO A1
0084  0200              LDK      A2,0
0086  42F0              CIO      A2,1,/30
0088  5C04              RB(NA)   *-2
008A  E524      CONT    LCR      A5,A1              LOAD THE CHAR INTO A5
008C  4530              OTR      A5,0,/30
008E  5C04              RB(NA)   *-2
0090  1101              ADK      A1,1               UPDATE AREA ADDRESS BY ONE
0092  E90E              CWR      A1,A11             CHECK IF LAST CHAR HAS BEEN PUNCHED
0094  5C0C              RB(NE)   CONT               NO? GO AND GET NEXT CHAR
0096  4280              CTO      A2,0,/30
0098  4BF0              SST      A3,/30
009A  5C04              RB(NA)   *-2
009C  5F1E              RB       START
                        END      START
```

SYMBOL TABLE

BEGIN	0000	R	CONT	008A	R	START	0080	R

PCH57

```
                         IDENT    PCH57
                   *
                   *     THIS PROGRAM CHECKS THE FUNCTION OF THE PUNCH'S HOLE SELECTION LOGIC
                   *
      0000         BEGIN EQU      *
                         RORG     BEGIN+/80
                   *
                   *
                   *     PUSH THE RUN BUTTON TO START THE PROGRAM
                   *
                   *
0080  FFFF               DATA     /FFFF
0082  0000               DATA     0
0084  207F         START HLT
0086  05FF               LDK      A5,/FF              LOAD DATA TO PUNCH ALL HOLES
0088  0200               LDK      A2,0
008A  42F0               CIO      A2,1,/30
008C  5C04               RB(NA)   *-2
008E  4530         CONT  OTR      A5,0,/30            SEND PUNCH COMMAND AND CHAR TO CU
0090  5C04               RB(NA)   *-2
0092  1D01               SUK      A5,1                MODIFY HOLE PATTERN
0094  5E08               RB(NN)   CONT                NOT NEGATIVE? GO AND PUNCH NEXT PATTERN
0096  42B0               CIO      A2,0,/30
0098  4BF0               SST      A3,/30
009A  5C04               RB(NA)   *-2
009C  5F1A               RB       START
                         END      START

           SYMBOL TABLE

 BEGIN    0000  R    CONT     008E  R    START    0084  R
```

CHKTRD

```
                           IDENT    CHKTRD
        0000     BEGIN     EQU      *
                           RORG     BEGIN+/80
                 *
                 *         THIS PROGRAM USES THE TAPE PRODUCED BY PCH51 TO VERIFY THE PTR READ CEL
                 *
                 *         LOAD THE PROGRAM START ADDRESS INTO A0
                 *
                 *         LOAD TAPE ON READER THEN PUSH RUN BUTTON
                 *
                 *
0080    FFFF               DATA     /FFFF
0082    0000               DATA     0
0084    207F     START     HLT
0086    208F               INH                            THIS PROGRAM RUNS IN INHIBIT MODE
0088    06FF               LDK      A6,255                LOAD CHAR COUNT IN A6
008A    0300               LDK      A3,0                  ZEROISE BUFFER CHAR COUNT
008C    0201               LDK      A2,1                  LOAD PARAMETER FOR INPUT CHARACTER
008E    42E0               CIO      A2,1,/20              SEND START COMMAND TO DEVICE
0090    5C04               RB(NA)   *-2
0092    4D20     INPT      INR      A5,0,/20              INPUT CHAR FROM READER
0094    5C04               RB(NA)   *-2
0096    9318               ADR      A3,A6                 ADD THE CHAR VALUE TO A3
0098    1E01               SUK      A6,1                  SUBTRACT ONE FROM CHAR COUNT
009A    5C0A               RB(NZ)   INPT                  RETURN TO READ ANOTHER CHAR IF NOT ZERO
009C    42A0               CIO      A2,0,/20              SEND STOP COMMAND TO DEVICE
009E    4AE0               SST      A2,/20                AND GET STATUS
00A0    5C04               RB(NA)   *-2
00A2    FB20               CWK      A3,32640              COMPARE VALUE IN A3 WITH CHECK SUM
00A4    7F80
00A6    5824               RB(E)    START                 IF EQUAL GO TO START
00A8    207F               HLT
                           END      START

        SYMBOL TABLE

 BEGIN    0000  R    INPT     0092  R    START    0084  R
```

3.61 PROGRAMS FOR THE MCU2 CARD

The MCU2 card can control the operations of a Card Reader and a Line Printer. The remarks concerning the common logic of the card given in the previous paragraph (second sentence) also apply to this card. Both devices can be operated on either the IOP or Programmed channel so a program for each type is given that will enable the basic functions of the CU logic and the device to be checked. They can all be modified to run continuously by changing the displacement value of the RB to START instruction so that it branches back to the first instruction of the program instead of the HLT instruction at START. The listings of the programs give the instructions for operating them and the function of the programs is:

- CRPROG and CRIOP — will read one card each time the RUN button is pushed (unless you have modified the program to run continuously) and the two programs operate on the Programmed Channel and IOP Channel respectively. You can verify the correct transfer of data by checking the holes punched in the cards against the data stored in the buffer.
- LPPROG and LPIOP — will store the two selected characters into the buffer (total of 78 printable characters in the unmodified version) until it is full then stores the Line Feed/Carriage Return characters and prints the line of characters. After each line printed you may change the two characters that you want printed; you can also send the other control characters to the printer but in this case you must modify the number of times they are loaded into the buffer. The two programs operate on the Programmed Channel and IOP Channel respectively.

CRPROG

```
                              IDENT    CRPROG
                      *
                      *
                      *       THIS PROGRAM READS A CARD VIA THE PROGRAMMED CHANNEL AND STOPS SO THAT
                      *       CONTENTS OF THE BUFFER CAN BE CHECKED AGAINST THE PUNCHED HOLES IN THE
                      *
                      *
       0000           BEGIN   EQU      *
                              RORG     BEGIN+/80
                      *
                      *
                      *       LOAD THE CARD(S) TO BE READ INTO THE CARD READER
                      *       AND START THE CARD READER
                      *
                      *       PUSH THE RUN BUTTON
                      *

0080   FFFF                   DATA     /FFFF
0082   0000                   DATA     0
0084   207F           START   HLT
0086   208F                   INH
                      *
0088   0200                   LDK      A2,0               LOAD ZERO INTO A2
008A   0100                   LDK      A1,0               ZEROISE WORD COUNTER
008C   8245           STORE   ST       A2,BUFF,A1         STORE ZERO INTOBUFF ADDRESS
008E   00B8  R
0090   1102                   ADK      A1,2               UPDATE WORD COUNT OF BUFF ADDRESS
0092   E920                   CWK      A1,42              CHECK IF LAST WORD REACHED
0094   002A
0096   5C0C                   RB(NE)   STORE              NO? GO AND STORE NEXT WORD
0098   0100                   LDK      A1,0               ZEROISE CHAR COUNTER
009A   0601                   LDK      A6,1               LOAD PARAMS FOR INPUT FUNCTION
009C   46C6                   CIO      A6,1,6             SEND CIO START COMMAND TO CARD READER
009E   5C04                   RB(NA)   *-2
00A0   4D06           READ    INR      A5,0,6             GET CHARS FROM CU
00A2   5C04                   RB(NA)   *-2
00A4   8545                   ST       A5,BUFF,A1         ANS STORE IN BUFFER
00A6   00B8  R
00A8   1102                   ADK      A1,2               UPDATE BUFFER ADDRESS
00AA   E920                   CWK      A1,/50             CHECK IF LAST COLUMN READ
00AC   0050
00AE   5C10                   RB(NE)   READ               NO? GO AND READ NEXT COLUMN
00B0   4686                   CIO      A6,0,6             SEND STOP COMMAND TO CU
00B2   4CC6                   SST      A4,6               GET STATUS FROM CU
00B4   5C04                   RB(NA)   *-2
00B6   5F34                   RB       START              GO AND WAIT FOR NEW DATA
00B8                  BUFF    RES      42
                              END      START
        SYMBOL TABLE

BEGIN    0000  R    BUFF    00B8  R    READ    00A0  R    START    0084  R
STORE    008C  R
```

CRIOP

```
                         IDENT    CRIOP
                   *
                   *
                   *     THIS PROGRAM READS A CARD AND STOPS  SO THAT THE CONTENTS
                   *     OF THE BUFFER CAN BE CHECKED AGAINST THE PUNCHED HOLES IN THE CARD
                   *
                   *
        0000       BEGIN EQU      *
                         RORG     BEGIN+/80
                   *
                   *
                   *     LOAD THE CARD(S) TO BE READ INTO THE CARD READER
                   *     AND START THE CARD READER
                   *
                   *     PUSH THE RUN BUTTON
                   *

  0080  FFFF             DATA     /FFFF
  0082  0000             DATA     0
  0084  207F       START HLT
  0086  20BF             INH
                   *
  0088  0200             LDK      A2,0                    LOAD ZERO INTO A2
  008A  0100             LDK      A1,0                    ZEROISE WORD COUNTER
  008C  8245       STORE ST       A2,BUFF,A1              STORE ZERO IN BUFF ADDRESS
  008E  00B0  R
  0090  1102             ADK      A1,2                    UPDATE WORD COUNT OF BUFF ADDRESS
  0092  E920             CWK      A1,42                   CHEK IF LAST WORD REACHED
  0094  002A
  0096  5C0C             RB(NE)   STORE                   NO? GO AND STORE NEXT WORD
  0098  8120             LDKL     A1,/8050                LOAD PARAMS FOR FIRST WER
  009A  8050
  009C  710C             WER      A1,/C                   AND SENT TO IOP
  009E  8120             LDKL     A1,BUFF                 LOAD BUFF ADDRESS FOR SECOND WER
  00A0  00B0  R
  00A2  710D             WER      A1,/D                   AND SEND TO IOP
  00A4  0601             LDK      A6,1                    LOAD PARAMS FOR INPUT FUNCTION
  00A6  46C6             CIO      A6,1,6                  SEND CIO START COMMAND TO CARD READER
  00A8  5C04             RB(NA)   *-2
  00AA  4CC6             SST      A4,6                    GET STATUS FROM CU
  00AC  5C04             RB(NA)   *-2
  00AE  5F2C             RB       START                   GO AND WAIT FOR NEW DATA
  00B0             BUFF  RES      42
                         END      START
        SYMBOL TABLE

 BEGIN   0000  R   BUFF    00B0  R   START   0084  R   STORE   008C  R
```

LPPROG

```
                             IDENT   LPPROG
                     *
                     *
                     *       THIS PROGRAM STORES TWO SELECTED CHARACTERS INTO A BUFFER
                     *       THEN PRINTS THEM AND STOPS WAITING NEW CHARACTERS TO BE SELECTED
                     *
                     *
      0000           BEGIN   EQU     *
                             RORG    BEGIN+/80
                     *
                     *
                     *       LOAD THE TWO CHARACTERS TO BE PRINTED INTO REGISTER A3
                     *
                     *       PUSH THE RUN BUTTON
                     *

0080  FFFF                   DATA    /FFFF
0082  0000                   DATA    0
0084  207F           START   HLT
0086  20BF                   INH
                     *
0088  850C                   LDR     A5,A3                    LOAD DATA INTO A5
008A  0100                   LDK     A1,0                     ZEROISE CHAR COUNTER
008C  8545           REPT    ST      A5,LPBUF,A1              STORE THE CONTENTS OF A5 INTO LPBUF
008E  00C0   R
0090  1102                   ADK     A1,2                     UPDATE LPBUF ADDRESS
0092  E920                   CWK     A1,78                    AND CHECK IF LINE FULL
0094  004E
0096  5C0C                   RB(NE)  REPT
0098  8520                   LDKL    A5,/0D0A                 LOAD LINE FEED CARRAIGE RETURN CHARS
009A  0D0A
009C  8545                   ST      A5,LPBUF,A1              AND STORE IN LPBUF
009E  00C0   R
00A0  0100                   LDK     A1,0                     ZEROISE CHAR COUNTER
00A2  0600                   LDK     A6,0                     LOAD PARAM FOR OUTPUT FUNCTION
00A4  46C7                   CIO     A6,1,/07                 SENT START COMMAND TO PRINTER
00A6  5C04                   RB(NA)  *-2                      TRY AGAIN IF NOT ACCEPTED
00A8  E544           OUCHAR  LC      A5,LPBUF,A1              GET CHAR FROM LPBUF
00AA  00C0   R
00AC  4507                   OTR     A5,0,7                   AND SEND TO LINE PRINTER
00AE  5C04                   RB(NA)  *-2
00B0  1101                   ADK     A1,1                     UPDATE LPBUF ADDRESS
00B2  E920                   CWK     A1,80                    CHECK IF LAST CHAR SENT
00B4  0050
00B6  5C10                   RB(NE)  OUCHAR                   NOT EQUAL? GOAND SEND NEXT CHAR
00B8  4687                   CIO     A6,0,7                   SEND STOP COMMAND TO LINE PRINTER
00BA  4CC7                   SST     A4,7                     GET STATUS
00BC  5C04                   RB(NA)  *-2
00BE  5F3C                   RB      START                    GO AND WAIT FOR NEW DATA
00C0                 LPBUF   RES     80
                             END     START

      SYMBOL TABLE        BEGIN   0000  R   LPBUF   00C0  R   OUCHAR  00A8  R   REPT    008C  R
                          START   0084  R
```

LPIOP

```
                             IDENT    LPIOP
                       *
                       *
                       *     THIS PROGRAM STORES TWO SELECTED CHARACTERS INTO A BUFFER
                       *     THEN PRINTS THEM AND STOPS WAITING NEW CHARACTERS TO BE SELECTED
                       *
                       *
          0000         BEGIN EQU      *
                             RORG     BEGIN+/80
                       *
                       *
                       *     LOAD THE TWO CHARACTERS TO BE PRINTED INTO REGISTER A3
                       *
                       *     PUSH THE RUN BUTTON
                       *

    0080  FFFF               DATA     /FFFF
    0082  0000               DATA     0
    0084  207F         START HLT
    0086  208F               INH
                       *
    0088  850C               LDR      A5,A3               LOAD DATA INTO A5
    008A  0100               LDK      A1,0                ZEROISE CHAR COUNTER
    008C  8545         REPT  ST       A5,LPBUF,A1         STORE THE CONTENTS OF A5 INTO LPBUF
    008E  00B8   R
    0090  1102               ADK      A1,2                UPDATE LPBUF ADDRESS
    0092  E920               CWK      A1,78               AND CHECK IF LINE FULL
    0094  004E
    0096  5C0C               RB(NE)   REPT
    0098  8520               LDKL     A5,/0D0A            LOAD LINE FEED CARRAIGE RETURN CHARS
    009A  0D0A
    009C  8545               ST       A5,LPBUF,A1         AND STORE IN LPBUF
    009E  00B8   R
    00A0  8120               LDKL     A1,/4050            LOAD PARAMS FOR FIRST WER
    00A2  4050
    00A4  710E               WER      A1,/E               AND SEND TO IOP
    00A6  8120               LDKL     A1,LPBUF            LOAD FIRST ADDRESS OF LPBUF
    00A8  00B8   R
    00AA  710F               WER      A1,/F               AND SEND TO IOP
    00AC  0600               LDK      A6,0                LOAD PARAM FOR OUTPUT FUNCTION
    00AE  46C7               CIO      A6,1,/07            SENT START COMMAND TO PRINTER
    00B0  5C04               RB(NA)   *-2                 TRY AGAIN IF NOT ACCEPTED
    00B2  4CC7               SST      A4,7                GET STATUS
    00B4  5C04               RB(NA)   *-2
    00B6  5F34               RB       START               GO AND WAIT FOR NEW DATA
    00B8               LPBUF RES      80
                             END      START
       SYMBOL TABLE

BEGIN    0000  R    LPBUF   00B8  R    REPT    008C  R    START   0084  R
```

3.62 Program COPY

This program enables you to punch a copy of any paper tape. It can be loaded by hand from the Control Panel or by using the IPL routine. Once loaded push the FEED HOLES button on the Paper Tape Punch to provide a leader of about 10cm, then load the paper tape you want punched on the Paper Tape Reader and push the RUN button. The program will read the tape and punch a copy simultaneously. When the Reader and Punch stop push the FEED HOLES button to provide a trailing edge for the tape. NOTE: The program has been written to reproduce any length of paper tape so no character counter has been included so the program must be loaded for each tape you wish to reproduce.

3.63 Program DUMP

This program enables the engineer to dump a selected area of memory on either the ASR or PER3100. It can be loaded either by hand from the Control Panel switches or using a program tape that contains the MINI-IPL (paragraph 3-72) and program produced by one of the methods described in paragraph 3-64 to 3-70. If the MINI-IPL method is used care must be taken that the program does not overwrite part of the area you wish dumped. The instructions for operating this program are given in the program listing and when the last address has been dumped the program branches back to the start ready for possible new parameters.

COPY

```
                               IDENT    COPY
                        *
                        *      THIS PROGAM  READS A PAPER TAPE AND C PUNCHES A COPY
                        *
             0000       BEGIN  EQU      *
                               RORG     BEGIN+/80
      0080   207F       START  HLT
      0082   0101              LDK      A1,1
      0084   0200              LDK      A2,0
      0086   41F0       READ   CIO      A1,1,/20
      0088   5C04              RB(NA)   *-2
      008A   4D20              INR      A5,0,/20
      008C   5C04              RB(NA)   *-2
      008E   41A0              CIO      A1,0,/20
      0090   4CF0              SST      A4,/20
      0092   5C04              RB(NA)   *-2
      0094   2420              ANK      A4,/20
      0096   EC20              CWK      A4,/20
      0098   0020
      009A   581C              RB(E)    START
      009C   42F0              CIO      A2,1,/30
      009E   5C04              RB(NA)   *-2
      00A0   4530              OTR      A5,0,/30
      00A2   5C04              RB(NA)   *-2
      00A4   42B0              CIO      A2,0,/30
      00A6   4BF0              SST      A3,/30
      00A8   5C04              RB(NA)   *-2
      00AA   5F26              RB       READ
                               END      START
```

DUMP

```
                                 IDENT    DUMP
          0000       BEGIN       EQU      *
                                 RORG     BEGIN+/80
                     *
                     *           THIS PROGRAM ENABLES THE ENGINEER TO DUMP A SELECTED
                     *           AREA OF MEMORY ON EITHER THE ASR OR PER 3100
                     *
                     *
                     *           LOAD THE STARTING ADDRESS OF THE AREA TO BE
                     *           DUMPED INTO REGISTER A7 FROM THE CONTROL PANEL
                     *
                     *           LOAD THE ENDING ADDRESS OF THE AREA INTO
                     *           REGISTER A8 FROM THE CONTROL PANEL
                     *
                     *           LOAD THE PROGRAM STARTING ADDRESS INTO
                     *           REGISTER A0 FROM THE CONTROL PANEL THEN
                     *           PUSH THE RUN BUTTON
                     *
                     *
    0080  FFFF                   DATA     /FFFF
    0082  0000                   DATA     0
    0084  207F       START       HLT
    0086  20BF                   INH                        THE PROGRAM RUNS IN INHIBIT MODE
                     *
    0088  0404                   LDK      A4,4              LOAD PARAMETERS FOR OUTPUT CHARACTER
                     *                                      WITH EVEN PARITY
    008A  44D0                   CIO      A4,1,/10          SEND START COMMAND TO DEVICE
    008C  813C       WORD        LDR*     A1,A7             LOAD THE CONTENTS OF THE MEMORY ADDRESS
                     *                                      CONTAINED IN A7 INTO A1
    008E  0204                   LDK      A2,4              LOAD CHARACTER COUNT FOR WORD
    0090  060F       CONT        LDK      A6,/F             LOAD MASK PATTERN
    0092  A604                   ANR      A6,A1             FIND THE PRINTABLE CHAR ADDRESS IN TABLE
    0094  E558                   LC       A5,TABLE,A6       AND LOAD INTO REGISTER A5
    0096  00C0  R
    0098  F549                   SC       A5,BUFF+1,A2      THEN STORE IN BUFF
    009A  00D1  R
    009C  39E4                   SRC      A1,4              SHIFT TO ISOLATE NEXT CHAR
    009E  1A01                   SUK      A2,1              SUBTRACT ONE FROM CHAR COUNT
    00A0  5C12                   RB(NZ)   CONT              AND BRANCH BACK IF NOT LAST CHAR IN WORD
    00A2  E348       OUCHAR      LC       A3,BUFF,A2        LOAD CHAR FROM BUFF INTO A3
    00A4  00D0  R
    00A6  4310                   OTR      A3,0,/10          AND SEND TO THE DEVICE
    00A8  5C04                   RB(NA)   *-2
    00AA  1201                   ADK      A2,1              UPDATE BUFF CHAR COUNT
    00AC  FA20                   CWK      A2,6              AND CHECK FOR LAST CHAR SENT
    00AE  0006
    00B0  5C10                   RB(NE)   OUCHAR            AND BRANCH BACK IF NOT LAST CHAR
    00B2  1702                   ADK      A7,2              UPDATE MEMORY ADDRESS COUNTER
    00B4  FF02                   CWR      A7,A8             AND CHECK FOR LAST ADDRESS DUMPED
    00B6  5D2C                   RB(NG)   WORD              IF NOT LAST OUTPUT NEW WORD
```

```
00046                                *                                IF LAST ADDRESS THEN
00047   0088   4490                  CIO       A4,0,/10               SEND STOP COMMAND TO THE DEVICE
00048   008A   4CD0                  SST       A4,/10                 AND GET STATUS
00049   008C   5C04                  RB(NA)    *-2
00050   008E   5E3C                  RB        START                  DUMP FINISHED GO TO START OF PROGRAM
00051   00C0   3031        TABLE     DATA      '0123456789ABCDEF'
        00C2   3233
        00C4   3435
        00C6   3637
        00C8   3839
        00CA   4142
        00CC   4344
        00CE   4546
00052   00D0   0A0D        BUFF      DATA      /0A0D
00053                                END       START

          SYMBOL TABLE

BEGIN    0000  R    BUFF     00D0  R    CONT      0090  R    OUCHAR   00A2  R
START    0084  R    TABLE    00C0  R    WORD      008C  R

          ASS.ERR.     00000
:EOF
PROG ELAPSED TIME:   00H-00M-00S-000MS-
```

3.64 PRODUCING PROGRAMS ON PAPER TAPE

For short programs (less than 80 characters), the tape produced can be input using the normal IPL routine; for programs with more than 80 characters MINI-IPL, suggested in this section, will have to be included before your own programs which can then be loaded using the IPL routine.

3.65 Using the ASR Punch in LOCAL Mode

If the ASR is fitted with a punch the hexadecimal characters of the program can be input from the keyboard provided the following rules are observed:

- The LOCAL/REMOTE switch must be set to LOCAL and the punch must be switched ON.
- The punch will treat any input from the keyboard as data and will reproduce the appropriate code on tape so if you make an error while typing in the instructions there is no way it can be corrected and you will have to begin again.
- To produce the tape leader and the trailing end of tape press the HERE IS key three times.
- The number keys 0 to 9 can be used normally but the letter keys A B C D E and F MUST NOT be used, instead use the J K L M N and O keys respectively which will produce the correct hexadecimal code on the tape.
- If the number of characters exceeds the line length the punch must be switched OFF before you press the CR and LF keys, and it must be switched ON again before you continue typing in the program instructions.
- The resulting program tape is in 8-bit ASCII but the Bootstrap can only decode Object code (either 8+8 or 4x4) so to load the program tape using the IPL routine the Control Panel switches must first be set for 4x4 mode (i.e. switches 0, 3 and 10 set to value 1). The program will then be loaded from the PTR.

3.66 If the program produced is less than 80 8-bit characters the Bootstrap will keep on reading (looking for the missing characters) until all the tape has passed through the read head. This does not affect the instructions that have been loaded but you must stop the CPU by pushing the INST button, then push the MC button to reset the device and CU logic. Any parameters can now be loaded into registers and the program will run normally.

3.67 Program ASC4x4

This program allows you to write programs using the operator's peripheral. The hexadecimal characters of the program can be entered in a similar way to that described above using the number keys 0 to 9 and letter keys J to O. However, the program is more flexible in that erroneous characters can be ignored and certain other characters are recognised which make the production of a program tape much easier. Once the program has been loaded put the start address of the program into register A0 and push the RUN button. The program will run and in addition to the keys mentioned above the following keys are significant:

- ↑ or ^ — The symbol and position of this key will depend on the device being used but its function and 7-bit code (/5E) are the same. The use of this key is to delete a character typed in error. When the program recognises this character it down dates the buffer counter and branches back to the input routine to accept a new character.
- Space Bar — The Space Bar is used to tell the program that the four hexadecimal characters of the program instruction are valid and that any corrections have been made. When the program recognises this character code (/20) it branches to the translate routine and stores them in the output buffer; it then branches back to the input routine for four new characters.
- # — This key is used to indicate that the last program character has been typed. When the program recognises this character code (/23) it branches to the CR/LF routine.

3.68 The program translates the valid characters received into 8-bit ASCII format and store them in a buffer until 40 valid characters have been received or the character has been recognised. It then sends the CR and IF characters to the Operator's device and starts the punch routine. If the last character code has not been recognised the program branches back to the input routine to accept new characters; if it has been recognised it branches back to the start of program and stops.

```
                        IDENT    ASC4X4
                *
                *       THIS PROGRAM ALLOWS ENGINEERS TO WRITE PROGRAMS ON THE OPERATOR'S
                *       PERIPHERAL AND THEN PUNCHES THE PROGRAM IN ASCII FOR 4X4 LOADING
                *
      0000      BEGIN   EQU      *
                        RORG     BEGIN+/AA
                *
                *       THIS ROUTINE ACCEPTS CHARACTERS FROM THE
                *       OPERATOR'S PERIPHERAL IN THE NON ECHO MODE
                *
                *
00AA  207F      START   HLT
00AC  20BF              INH
00AE  0100      OUTCT   LDK      A1,0                  OUTPUT BUFFER COUNTER ZEROISED
00B0  0300      INCT    LDK      A3,0                  ZEROISE INPUT BUFFER COUNTER
00B2  0601      INPT    LDK      A6,1
00B4  46D0              CIO      A6,1,/10
00B6  5C04              RB(4)    *-2
00B8  4D10              INR      A5,0,/10
00BA  5C04              RB(4)    *-2
00BC  4690              CIO      A6,0,/10
00BE  4CD0              SST      A4,/10
00C0  5C04              RB(4)    *-2
00C2  0600              LDK      A6,0
00C4  46D0              CIO      A6,1,/10
00C6  5C04              RB(4)    *-2
00C8  4510              OTR      A5,0,/10
00CA  5C04              RB(4)    *-2
00CC  4690              CIO      A6,0,/10
00CE  4CD0              SST      A4,/10
00D0  5C04              RB(4)    *-2
00D2  257F              ANK      A5,/7F                MODIFIES CHARACTER TO 7 DATA BITS
                *
                *       THIS ROUTINE CHECKS FOR  ERRONEOUS CHARS AND END OF PROGRAM
                *
00D4  ED20              CWK      A5,/5E                CHECK IF LAST CHAR WAS TYPED IN ERROR
00D6  005E
00D8  5404              RF(NE)   SPCE
00DA  1B01              SUK      A3,1                  THE LAST CHAR WAS TYPED IN ERROR SO
00DC  5F2C              RB       INPT                  DOWNDATE CHAR COUNTER AND READ NEXT CHAR
00DE  ED20      SPCE    CWK      A5,/20                CHECK FOR SPACE BAR CHARACTER
00E0  0020
00E2  500E              RF(E)    TRAN                  GO TO TRAN IF 4 VALID CHARS INPUT
00E4  ED20              CWK      A5,/23                CHECK FOR END OF PROGRAM    #
00E6  0023
00E8  5038              RF(E)    CRLP
                *
                *       THIS ROUTINE STORES INPUT CHARS IN TEMPORARY BUFFER
                *
```

```
00EA  F54D          SC       A5,TEMBUF,A3        STORE CHAR IN TEMBUF
00EC  0158  R
00EE  1301          ADK      A3,1                UPDATE INPUT BUFFER COUNT
00F0  5F40          RB       INPT                GO TO INPUT ROUTINE
              *
              *     THIS ROUTINE TRANSLATES CHARS INTO ASCII FORMAT
              *
00F2  0300    TRAN  LDK      A3,0                ZEROISE TEMBUF COUNTER
00F4  B208    TRAN1 XRR      A2,A2               CLEAR REGISTER
00F6  E54C          LC       A5,TEMBUF,A3        GET CHAR FROM TEMBUF
00F8  0158  R
00FA  8214          LDR      A2,A5               SAVE CHAR
00FC  250F          ANK      A5,/0F              SAVE RIGHT MOST BITS
00FE  2240          ANK      A2,/40
0100  EA20          CWK      A2,/40              CHECK IF CHAR IS DECIMAL
0102  0040
0104  5404          RF(NE)   DECIM
0106  15C0          ADK      A5,/C0              ALPHA CHAR
0108  5702          RF       STORE
010A  1580    DECIM ADK      A5,/B0              DECIMAL CHAR
              *
              *     THIS ROUTINE STORES THE PROGRAM CHARS IN BUFFER
              *     AND SENDS CR LF AFTER 40 VALID CHARS HAVE BEEN RECIEVED
              *
010C  E545    STORE SC       A5,BUFF,A1          STORE VALID CHAR
010E  015E  R
0110  1101          ADK      A1,1                UPDATE CHAR COUNTER
0112  1301          ADK      A3,1                UPDATE TEMBUF COUNT
0114  EB20          CWK      A3,4                CHECK IF ALL 4 CHARS HAVE BEEN TRANSLATE
0116  0004
0118  5C26          RB(NE)   TRAN1               TRANSLATE NEW CHARACTER IF NOT ZERO
011A  E920          CWK      A1,40               COMPARE FOR BUFFER FULL
011C  0028
011E  5002          RF(E)    CRLP                IF YES SEND CR/LF CHARS THEN PUNCH BUFFF
0120  5F72          RB       INCT                IF ZERO GO TO INPUT ROUTINE FOR NEW CHAR
0122  070A    CRLP  LDK      A7,/0A              LOAD LF CHAR
0124  46D0          CIO      A6,1,/10
0126  4710          OTR      A7,0,/10            AND SEND TO TYPEWRITER
0128  5C04          RB(4)    *-2
012A  070D          LDK      A7,/0D              LOAD CR CHAR
012C  4710          OTR      A7,0,/10            AND SEND TO TYPEWRITER
012E  5C04          RB(4)    *-2
0130  4690          CIO      A6,0,/10
0132  4CD0          SST      A4,/10
0134  5C04          RB(4)    *-2
              *
              *     THIS ROUTINE PUNCHES THE ASCII PROGRAM ON TAPE
              *
0136  0200          LDK      A2,0                ZEROISE  BUFF COUNTER
0138  46F0          CIO      A6,1,/30            START PUNCH
013A  5C04          RB(4)    *-2
013C  E348    CONT  LC       A3,BUFF,A2          LOAD CHAR FROM BUFF
013E  015E  R
0140  4330          OTR      A3,0,/30            AND PUNCH
0142  5C04          RB(4)    *-2
0144  1201          ADK      A2,1                UPDATE BUFF COUNTER
0146  1901          SUK      A1,1                SUBTRACT 1 FROM CHAR COUNT
```

```
00097  0148  5C0E              RB(4)      CONT           AND CHECK FOR LAST CHAR
00098  014A  46B0              CTO        A6,0,/30       THEN STOP PUNCH
00099  014C  4CF0              SST        A4,/30
00100  014E  5C04              RB(NA)     *-2
00101  0150  ED20              CWK        A5,/23         CHECK FOR LAST PROGRAM CHAR
       0152  0023
00102  0154  5CA8              RB(4)      OUTCT          NO?  ACCEPT MORE CHARACTERS
00103  0156  5FAF              RB         START
00104  0158         TEMBUF     RES        3              TEMPORARY BUFFER
00105  015E         BUFF       RES        25
00106                          END        START

          SYMBOL TABLE

BEGIN   0000  R   BUFF    015E  R   CONT    013C  R   CRLP    0122  R
DECIM   010A  R   INCT    00B0  R   INPT    00B2  R   OUTCT   00AE  R
SPCE    00DE  R   START   00AA  R   STORE   010C  R   TEMBUF  0158  R
TRAN    00F2  R   TRAN1   00F4  R
```

3.69 The program has 74 instructions and in the program listing the start address is shown as 00AA. This is so that if your only means of producing program tapes is by one of the examples given in this section you can include the MINI-IPL in front of the program and load it using the 4x4 IPL routine. If you have access to an Assembler and Linkage Editor you will have no trouble in producing a master tape of this program, but if you have to load it by hand from the Control Panel switches before reproducing it by one of the methods suggested in this section great care must be taken that you do not introduce errors into the program whilst it is being loaded.

3.70 Program HEXTAP

This program operates exactly the same way as the program ASC4x4 and all instructions regarding the use of that program apply to this program. The only differences are that this program operates in Echo mode and the output tape is punched in 8+8 hexadecimal format. If you include the MINI-IPL in front of the program it can then be loaded from the PTR using the normal IPL routine.

HEXTAP

```
                        IDENT   HEXTAP
                *
                *       THIS PROGRAM ALLOWS ENGINEERS TO WRITE PROGRAMS ON THE OPERATOR'S
                *       PERIPHERAL AND PUNCHES THE PROGRAM TAPE IN HEXADECIMAL 8+8 FORMAT
                *
      0000      BEGIN   EQU     *
                        RORG    BEGIN+/AA
                *
                *       THHIS ROUTINE ACCEPTS CHARACTERS FROM THE OPERATOR'S PERIPHERAL
                *       IN ECHO MODE
                *
00AA  207F      START   HLT
00AC  20BF              INH
00AE  0100      OUTCT   LDK     A1,0                  OUTPUT BUFFER COUNTER ZEROISED
00B0  0300      INCT    LDK     A3,0                  ZEROISE INPUT BUFFER COUNTER
00B2  0621      INPT    LDK     A6,/21
00B4  46D0              CIO     A6,1,/10
00B6  5C04              RB(4)   *-2
00B8  4D10              TNR     A5,0,/10
00BA  5C04              RB(4)   *-2
00BC  4690              CIO     A6,0,/10
00BE  4CD0              SST     A4,/10
00C0  5C04              RB(4)   *-2
00C2  257F              ANK     A5,/7F                MODIFIES CHARACTER TO 7 DATA BITS
                *
                *       THIS ROUTINE CHECKS FOR  ERRONEOUS CHARS AND END OF PROGRAM
                *
00C4  ED20              CWK     A5,/5E                CHECK IF LAST CHAR WAS TYPED IN ERROR
00C6  005E
00C8  5404              RF(NE)  SPCE
00CA  1B01              SUK     A3,1                  THE LAST CHAR WAS TYPED IN ERROR SO
00CC  5F1C              RB      INPT                  DOWNDATE CHAR COUNTER AND READ NEXT CHAR
00CE  ED20      SPCE    CWK     A5,/20                CHECK FOR SPACE BAR CHARACTER
00D0  0020
00D2  500E              RF(E)   TRAN                  GO TO TRAN IF 4 VALID CHARS INPUT
00D4  ED20              CWK     A5,/23                CHECK FOR END OF PROGRAM   #
00D6  0023
00D8  502C              RF(E)   CRLP
                *
                *       THIS ROUTINE STORES INPUT CHARS IN TEMPORARY BUFFER
                *
00DA  E54D              SC      A5,TEMBUF,A3          STORE CHAR IN TEMBUF
00DC  013E  R
00DE  1301              ADK     A3,1                  UPDATE INPUT BUFFER COUNT
00E0  5F30              RB      INPT                  GO TO INPUT ROUTINE
                *
                *       THIS ROUTINE TRANSLATES CHARS INTO ASCII FORMAT
                *
00E2  0300      TRAN    LDK     A3,0                  ZEROISE TEMBUF COUNTER
00E4  0404              LDK     A4,4                  LOAD SHIFT COUNTER
```

```
00E6  B208        TRAN1   XRR     A2,A2              CLEAR REGISTER
00E8  3A44                SLL     A2,4               SHIFT CHARS 4 PLACES LEFT
00EA  E54C                LC      A5,TEMBUF,A3       GET CHAR FROM TEMBUF
00EC  013E  R
00EE  250F                ANK     A5,/0F             SAVE RIGHT MOST BITS
00F0  9214                ADR     A2,A5              ADD 4 BIT CHAR FROM A5 INTO A2
00F2  1301                ADK     A3,1               UPDATE TEMBUF COUNTER
00F4  1C01                SUK     A4,1               DOWNDATE SHIFT COUNTER AND IF NOT ZERO
00F6  5C10                RB(NZ)  TRAN1+2            BRANCH AND TRANSLATE NEW CHAR
                  *
                  *       THIS ROUTINE STORES THE PROGRAM CHARS IN BUFFER
                  *       AND SENDS CR LF AFTER 40 VALID CHARS HAVE BEEN RECIEVED
                  *
00F8  8245        STORE   ST      A2,BUFF,A1         STORE TWO 8 BIT HEXADECIMAL CHARS IN BUF
00FA  0144  R
00FC  1102                ADK     A1,2               UPDATE CHAR COUNTER
00FE  E920                CWK     A1,20              COMPARE FOR BUFFER FULL
0100  0014
0102  5002                RF(E)   CRLP               IF YES SEND CR/LF CHARS THEN PUNCH BUFFE
0104  5F56                RB      INCT               IF ZERO GO TO INPUT ROUTINE FOR NEW CHAR
0106  070A        CRLP    LDK     A7,/0A             LOAD LF CHAR
0108  0600                LDK     A6,0               LOAD START OUTPUT FOR DEVICE
010A  46D0                CIO     A6,1,/10
010C  4710                OTR     A7,0,/10           AND SEND TO TYPEWRITER
010E  5C04                RB(4)   *-2
0110  070D                LDK     A7,/0D             LOAD CR CHAR
0112  4710                OTR     A7,0,/10           AND SEND TO TYPEWRITER
0114  5C04                RB(4)   *-2
0116  4690                CIO     A6,0,/10
0118  4CD0                SST     A4,/10
011A  5C04                RB(4)   *-2
                  *
                  *       THIS ROUTINE PUNCHES THE ASCII PROGRAM ON TAPE
                  *
011C  0200                LDK     A2,0               ZEROISE  BUFF COUNTER
011E  46F0                CIO     A6,1,/30           START PUNCH
0120  5C04                RB(4)   *-2
0122  E348        CONT    LC      A3,BUFF,A2         LOAD CHAR FROM BUFF
0124  0144  R
0126  4330                OTR     A3,0,/30           AND PUNCH
0128  5C04                RB(4)   *-2
012A  1201                ADK     A2,1               UPDATE BUFF COUNTER
012C  1901                SUK     A1,1               SUBTRACT 1 FROM CHAR COUNT
012E  5C0E                RB(4)   CONT               AND CHECK FOR LAST CHAR
0130  46B0                CIO     A6,0,/30           THEN STOP PUNCH
0132  4CF0                SST     A4,/30
0134  5C04                RB(NA)  *-2
0136  ED20                CWK     A5,/23             CHECK FOR LAST PROGRAM CHAR
0138  0023
013A  5C8E                RB(4)   OUTCT              NO?  ACCEPT MORE CHARACTERS
013C  5F94                RB      START
013E              TEMBUF  RES     3                  TEMPORARY BUFFER
0144              BUFF    RES     25
                          END     START
      SYMBOL TABLE

BEGIN   0000  R   BUFF    0144  R   CONT    0122  R   CRLP    0106  R
INCT    00B0  R   INPT    00B2  R   OUTCT   00AE  R   SPCE    00CE  R
START   00AA  R   STORE   00F8  R   TEMBUF  013E  R   TRAN    00E2  R
TRAN1   00E6  R
```

3.71 SUGGESTIONS FOR SOPHISTICATED PROGRAMS

As a general rule sophisticated programs perform more than one function and are longer than 20 instructions (so cannot easily be loaded by hand). An easy way to produce these programs is to take routines from simple programs (or routines you have written yourself) and link them together adding specific routines to perform the tasks you want. The two previous programs are examples of this technique. They comprise routines for the operator's device, the Paper Tape Punch, and a linking routine that translates the character codes input into either ASCII or hexadecimal character code. Make sure that when the same register is used for more than one function you do not corrupt data needed in another part of the program, that the buffer address is the same, that the displacement value of branch instructions is correct for the new program, and that all superfluous instructions from either routine have been omitted. The program examples that follow have all been produced in this way and should enable you to produce your own programs capable of performing specific and more complicated tasks even with a basic system.

3.72 MINI-IPL Routine

For programs longer than 80 characters some form of software IPL is needed to load the final part of the program into memory. You will remember that the Bootstrap is programmed to load 80 8-bit characters from the device (whose address and input mode are set up on the DATA switches) before it branches to address 84, so if we start the MINI-IPL at this address it will then load the rest of the program. The first four characters of the IPL (/FFFF and 0000) are not used by the routine but are needed to make sure that the first instruction of the routine is in location /84. This instruction loads the number of characters the IPL has to read into register A5. To calculate the character count loaded into A5 count the number of locations needed by the MINI-IPL (starting with data /FFFF) plus your program and multiply the result by two to give the number of characters. Now subtract 80 (which is the number of characters loaded by the Bootstrap) and the remainder is the number you need to load into register A5. If this number is greater than 255 you will have to change the LDK instruction into an LDKL putting the count into the second word. Make sure that the count is accurate otherwise the IPL will either not load all the instructions (count too short) or will load the blank tape of the trailing end as all zeros. In the latter case if only one or two such characters are read it should not affect the program, but if the End of tape is detected then it will almost certainly cause the program to abort. The rest of the instructions form a simple routine to read characters from the PTR and store them in the address pointed to by register A6, which is the register used by the Bootstrap to load the first 80 characters. When this routine has read all the characters (register A5's value equals zero) the PTR is stopped and, unless the first instruction of your program is Halt, your program is entered and starts to run. You can modify this routine to suit your own requirements, for example the start address of the program can be changed to any location in memory greater than /9E (which is the normal start address) by inserting a Load Constant instruction, as the first instruction of the routine, that contains the new start address of the program. You will also have to insert an RF or ABL instruction (with the new start address as the displacement value) after the RB instruction in location 9C. The remainder of the first 80 characters must all be zero otherwise the modified IPL will overwrite part of your program.

MINI-IPL

```
                     *
                     *    THIS IS A MINI IPL FOR LOADING PAPER TAPE IN 8+8 HEXADECIMAL FORMAT
                     *
0080  FFFF                DATA     /FFFF
0082  0000                DATA     0000
0084  0550                LDK      A5,80          LOAD CHAR COUNT FOR REST OF PROGRAM
0086  0101                LDK      A1,1           LOAD PARAMETERS FOR CIO START
0088  41E0                CIO      A1,1,/20       AND SEND TO CU
008A  5C04                RB(NA)   *-2
008C  4F20           INR  INR      A7,0,/20       SEND INR TO CU
008E  5C04                RB(NA)   *-2
0090  E739                SCR      A7,A6          STORE CHARACTER IN LOCATION
0092  1601                ADK      A6,1           UPDATE MEMORY ADDRESS
0094  1D01                SUK      A5,1           DECREMENT CHAR COUNTER
0096  5C0C                RB(NZ)   INR            AND READ ANOTHER CHAR IF NOT ZERO
0098  41A0                CIO      A1,0,/20       SEND STOP COMMNAD TO CU
009A  4FE0                SST      A7,/20         AND THEN SEND SST COMMAND
009C  5C04                RB(NA)   *-2
                     *
```

3.73 Program MINDUM

This program is an example of how to link the MINI-IPL to a program longer than 80 characters (in this case 82 characters), so that it can be loaded using the IPL routine. The program chosen for the exercise was DUMP and the first step was to count the number of characters, including the two dummy instructions Data /FFFF and Data 0000, which was 82 and therefore two characters too long to be loaded with the IPL routine. The number of characters needed for the MINI-IPL (excluding the two dummy instructions shown in the listing) is 26 so the total number of characters to be loaded by the MINI-IPL is 28. The program tape was produced using the program HEXTAP starting with the two dummy instructions, then the MINI-IPL and finally by the instructions for the program DUMP (excluding the two dummy instructions shown in the listing). The completed tape can now be loaded using the IPL routine and the instructions for running the programs are the same as those given for DUMP.

MINDUM

```
                            IDENT   MINDUM
           0000    BEGIN    EQU     *
                            RORG    BEGIN+/80
                   *
                   *        THIS PROGRAM ENABLES THE ENGINEER TO DUMP A SELECTED
                   *        AREA OF MEMORY ON EITHER THE ASR OR PER 3100
                   *
                   *        IT CONTAINS THE MINI-IPL AND CAN BE LOADED
                   *        USING THE IPL ROUTINE
                   *
                   *        LOAD THE STARTING ADDRESS OF THE AREA TO BE
                   *        DUMPED INTO REGISTER A7 FROM THE CONTROL PANEL
                   *
                   *        LOAD THE ENDING ADDRESS OF THE AREA INTO
                   *        REGISTER A8 FROM THE CONTROL PANEL
                   *
                   *        PUSH THE RUN BUTTON
                   *
                   *
                   *        THIS IS A MINI IPL FOR LOADING PAPER TAPE IN 8+8 HEXADECIMAL FORMAT
                   *
0080       FFFF             DATA    /FFFF
0082       0000             DATA    0000
0084       051C             LDK     A5,28              LOAD CHAR COUNT FOR REST OF PROGRAM
0086       0101             LDK     A1,1               LOAD PARAMETERS FOR CIO START
0088       41E0             CIO     A1,1,/20           AND SEND TO CU
008A       5C04             RB(NA)  *-2
008C       4F20    INR      INR     A7,0,/20           SEND INR TO CU
008E       5C04             RB(NA)  *-2
0090       E739             SCR     A7,A6              STORE CHARACTER IN LOCATION
0092       1601             ADK     A6,1               UPDATE MEMORY ADDRESS
0094       1D01             SUK     A5,1               DECREMENT CHAR COUNTER
0096       5C0C             RB(NZ)  INR                AND READ ANOTHER CHAR IF NOT ZERO
0098       41A0             CIO     A1,0,/20           SEND STOP COMMNAD TO CU
009A       4FE0             SST     A7,/20             AND THEN SEND SST COMMAND
009C       5C04             RB(NA)  *-2
                   *
                   *        THE PROGRAM STARTS HERE
                   *
009E       207F    START    HLT
00A0       208F             INH                        THE PROGRAM RUNS IN INHIBIT MODE
                   *
00A2       0404             LDK     A4,4               LOAD PARAMETERS FOR OUTPUT CHARACTER
                   *                                   WITH EVEN PARITY
00A4       44D0             CIO     A4,1,/10           SEND START COMMAND TO DEVICE
00A6       813C    WORD     LDR*    A1,A7              LOAD THE CONTENTS OF THE MEMORY ADDRESS
                   *                                   CONTAINED IN A7 INTO A1
00A8       0204             LDK     A2,4               LOAD CHARACTER COUNT FOR WORD
00AA       060F    CONT     LDK     A6,/F              LOAD MASK PATTERN
00AC       A604             ANR     A6,A1              FIND THE PRINTABLE CHAR ADDRESS IN TABLE
```

```
00050  00AE  F558             LC      A5,TABLE,A6          AND LOAD INTO REGISTER A5
       00B0  00DA  R
00051  00B2  F549             SC      A5,BUFF+1,A2         THEN STORE IN BUFF
       00B4  00FB  R
00052  00B6  39E4             SRC     A1,4                 SHIFT TO ISOLATE NEXT CHAR
00053  00B8  1A01             SUK     A2,1                 SUBTRACT ONE FROM CHAR COUNT
00054  00BA  5C12             RB(NZ)  CONT                 AND BRANCH BACK IF NOT LAST CHAR IN WORD
00055  00BC  F348     OUCHAR  LC      A3,BUFF,A2           LOAD CHAR FROM BUFF INTO A3
       00BE  00EA  R
00056  00C0  4310             OTR     A3,0,/10             AND SEND TO THE DEVICE
00057  00C2  5C04             RB(NA)  *-2
00058  00C4  1201             ADK     A2,1                 UPDATE BUFF CHAR COUNT
00059  00C6  FA20             CWK     A2,6                 AND CHECK FOR LAST CHAR SENT
       00C8  0006
00060  00CA  5C10             RB(NE)  OUCHAR               AND BRANCH BACK IF NOT LAST CHAR
00061  00CC  1702             ADK     A7,2                 UPDATE MEMORY ADDRESS COUNTER
00062  00CE  FF02             CWR     A7,A8                AND CHECK FOR LAST ADDRESS DUMPED
00063  00D0  5D2C             RB(NG)  WORD                 IF NOT LAST OUTPUT NEW WORD
00064                 *                                    IF LAST ADDRESS THEN
00065  00D2  4490             CIO     A4,0,/10             SEND STOP COMMAND TO THE DEVICE
00066  00D4  4CD0             SST     A4,/10               AND GET STATUS
00067  00D6  5C04             RB(NA)  *-2
00068  00D8  5F3C             RB      START                DUMP FINISHED GO TO START OF PROGRAM
00069  00DA  3031     TABLE   DATA    '0123456789ABCDEF'
       00DC  3233
       00DE  3435
       00E0  3637
       00F2  3839
       00E4  4142
       00E6  4344
       00E8  4546
00070  00EA  0A0D     BUFF    DATA    /0A0D
00071                         END     START

        SYMBOL TABLE

BEGIN   0000  R   BUFF    00EA  R   CONT    00AA  R   INR     008C  R
OUCHAR  00BC  R   START   009F  R   TABLE   00DA  R   WORD    00A6  R
```

3.74 Program MEM57

This is another example of how to link several routines to the simple program MEMHAN to produce a sophisticated program that will carry out a check of the entire memory without intervention from the engineer unless there is a detected error. It is more of a confidence program (after a memory fault has been repaired) than a specific test of a selected area whilst trying to locate a fault. The routines used are:

- The MINI-IPL routine so that the program can be loaded using the IPL routine. You will see that the only change necessary (from the other two listings in this section) is to load the character count register A5 with a count of 176.
- A routine to check the size of memory in an unknown system without doing a physical check on the cards. The routine assumes a maximum memory size of 32K and tries to write in the last location. If unsuccessful it subtracts 4K from the address in register A9 and tries to store the pattern in the new address. In the original routine when the size of memory had been found it branched to a Halt instruction so that the memory size could be read from register A9; so for this program the Halt instruction became the first instruction of the next routine.
- A routine to store the Start and End addresses of the area tested into a buffer called MEMADD. The previous routine has already computed the End address and this is store in the first word of the buffer; the Start address is the first location after the end of this program and it is stored in the second word of the buffer.
- Four Test Patterns and a Software Timer so that the series of test patterns will be repeated a number of times (in this case 25).
- The program MEMHAN slightly modified so that the Start and End addresses are taken from the buffer MEMADD instead of from registers, and the Test Patterns would be selected automatically.
- Routines to compute the New Start and New End addresses, to compute the New Load Area for the program, and to compute the New Addresses for the buffers for MEMADD, Test Patterns, and Timer.
- A routine to transfer the program to the New Load Area (which will be in the top end of memory) and to re-start the program MEMHAN so that it checks the bottom end of memory.

Once the program has been loaded it starts automatically and when it has checked the complete memory it stops on a Halt instruction that has been loaded into the last address of memory. If an error is detected during the running of the program it will stop on the HLT instruction in the Read routine and you can use the same technique, to check the location in error and the Write and Read patterns, as described for MEMHAN.

MEM57

```
                            IDENT    MEM57
                      *
                      *     THIS PROGRAM TESTS THE ENTIRE MEMORY
                      *
      0000            BEGIN EQU      *
                            RORG     BEGIN+/80
                      *
                      *     THIS IS A MINI IPL FOR LOADING PAPER TAPE IN 8+8 HEXADECIMAL FORMAT
                      *
0080  FFFF                  DATA     /FFFF
0082  0000                  DATA     0000
0084  05B0                  LDK      A5,176              LOAD CHAR COUNT FOR REST OF PROGRAM
0086  0101                  LDK      A1,1                LOAD PARAMETERS FOR CIO START
0088  41E0                  CIO      A1,1,/20            AND SEND TO CU
008A  5C04                  RB(NA)   *-2
008C  4F20            INR   INR      A7,0,/20            SEND INR TO CU
008E  5C04                  RB(NA)   *-2
0090  E739                  SCR      A7,A6               STORE CHARACTER IN LOCATION
0092  1601                  ADK      A6,1                UPDATE MEMORY ADDRESS
0094  1D01                  SUK      A5,1                DECREMENT CHAR COUNTER
0096  5C0C                  RB(NZ)   INR                 AND READ ANOTHER CHAR IF NOT ZERO
0098  41A0                  CIO      A1,0,/20            SEND STOP COMMNAD TO CU
009A  4FE0                  SST      A7,/20              AND THEN SEND SST COMMAND
009C  5C04                  RB(NA)   *-2
                      *
                      *     THIS ROUTINE COMPUTES THE MEMORY SIZE
                      *
009E  8720            START LDKL     A7,/5555            TEST PATTERN
00A0  5555
00A2  81A0                  LDKL     A9,/FC00            LOAD MAXIMUM MEMORY SIZE
00A4  FC00
00A6  8727            COMP  STR      A7,A9               TRY TO STORE PATTERN
00A8  FF26                  CWR*     A7,A9               CHECK IF LOCATION EXISTS
00AA  5006                  RF(0)    PARAMS              YES? GO TO PROGRAM
00AC  99A0                  SUKL     A9,/2000            NO? SUBTRACT 4K
00AE  2000
00B0  5F0C                  RB       COMP                AND TRY AGAIN
                      *
                      *     THIS ROUTINE LOADS THE START AND END ADDRESSES
                      *
00B2  81C1            PARAMS ST      A9,MEMADD           STORE END ADDRESS
00B4  00C0  R
00B6  81A0                  LDKL     A9,/180             LOAD START ADDRESS FOR WRITE/READ ROUTIN
00B8  0180
00BA  81C1                  ST       A9,MEMADD+2         AND STORE IN MEMADD BUFFER
00BC  00C2  R
00BE  570E                  RF(7)    LOAD                GO TO STRAT OF WRITE/READ ROUTINE
                      *
                      *
00C0  0000            MEMADD DATA    0                   RESERVE TO LOCATIONS
```

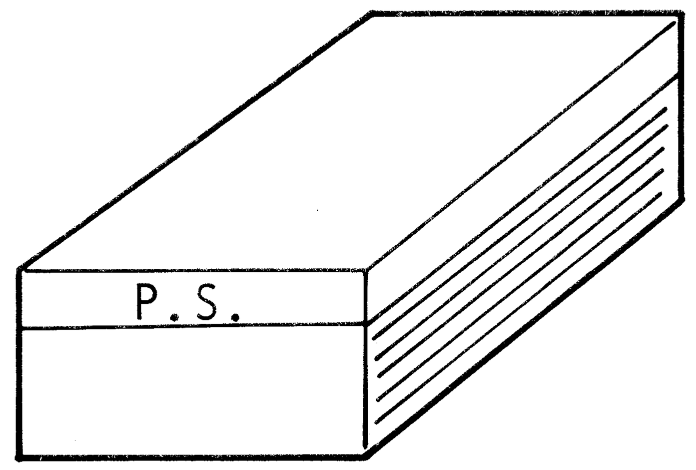
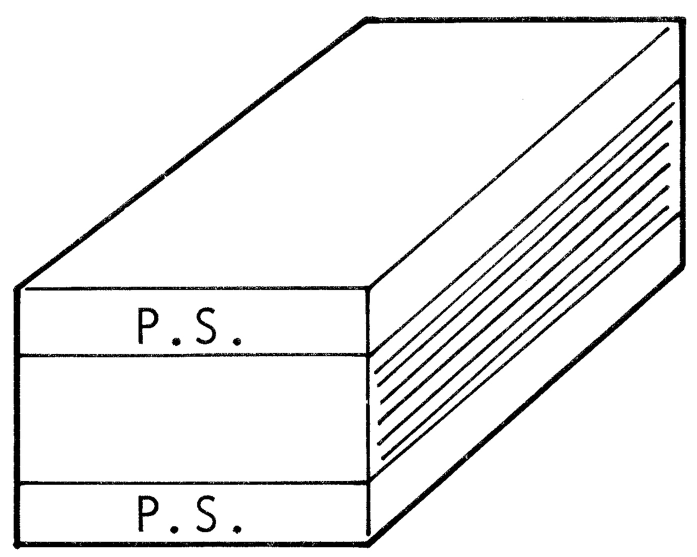
```
00044  00C2  0000                DATA    0                     FOR MEMORY ADDRESSES
00045                      *
00046  00C4  5555          PATEN DATA    /5555                 TEST
00047  00C6  AAAA                DATA    /AAAA                 PATTERNS
00048  00C8  FFFF                DATA    /FFFF                 FOR WRITE
00049  00CA  0000                DATA    /0000                 READ
00050                      *
00051  00CC  0019          TIMER DATA    25                    SOFTWARE TIMER COUNT
00052                      *
00053                      *     THESE INSTRUCTIONS PREPARE THE PROGRAM PARAMETERS
00054                      *
00055  00CE  0700          LOAD  LDK     A7,0                  ZEROISE REPEAT COUNTER
00056  00D0  0200          LOAD1 LDK     A2,0                  ZEROISE PATTERN COUNTER
00057  00D2  85C8          LDPAT LD      A13,PATEN,A2          LOAD TEST PATTERN
       00D4  00C4  R
00058                      *
00059                      *     THIS IS THE WRITE MEMORY ROUTINE
00060                      *
00061  00D6  81C0          TEST  LD      A9,MEMADD+2           LOAD START ADDRESS
       00D8  00C2  R
00062  00DA  85A7          WRITE STR     A13,A9                WRITE TEST PATTERN IN MEMORY
00063  00DC  91A0                ADKL    A9,2                  UPDATE MEMORY ADDRESS
       00DE  0002
00064  00E0  E9C0                CW      A9,MEMADD             CHECK FOR LAST ADDRESS LOADED
       00E2  00C0  R
00065  00E4  5C0C                RB(NE)  WRITE                 REPEAT ROUTINE IF NOT LAST ADDRESS
00066                      *
00067                      *     THIS IS THE READ ROUTINE
00068                      *
00069  00E6  81C0                LD      A9,MEMADD+2           LOAD START ADDRESS
       00E8  00C2  R
00070  00EA  80A6          READ  LDR*    A8,A9                 READ PATTERN FROM MEMORY
00071  00EC  E8C8                CW      A8,PATEN,A2           COMPARE MEM CONTENTS WITH PATTERN
       00EE  00C4  R
00072  00F0  5002                RB(E)   CONT                  IF EQUAL CONTINUE READ ROUTINR
00073  00F2  207F                HLT                           IF NOT EQUAL STOP ROUTINE
00074  00F4  91A0          CONT  ADKL    A9,2                  UPDATE MEMORY ADDRESS
       00F6  0002
00075  00F8  E9C0                CW      A9,MEMADD             CHECK FOR LAST ADDRESS READ
       00FA  00C0  R
00076  00FC  5C14                RB(NE)  READ                  CONTINUE IF NOT LAST WORD
00077  00FE  1202                ADK     A2,2                  UPDATE PATTERN COUNTER
00078  0100  EA20                CWK     A2,8                  CHECK FOR LAST PATTERN
       0102  0008
00079  0104  5C34                RB(NE)  LDPAT                 IF NOT LOAD NEW PATTERN
00080  0106  1701                ADK     A7,1                  UPDATE SOFTWARE TIMER COUNT
00081  0108  EF40                CW      A7,TIMER              CHECK FOR LAST TEST
       010A  00CC  R
00082  010C  5C3E                RB(NE)  LOAD1                 IF NO CONTINUE WRITE/READ ROUTINES
00083                      *
00084                      *     THIS ROUTINE COMPUTES THE NEW START AND END ADDRESSES
00085                      *
00086  010E  84C0                LD      A12,MEMADD            SAVE END ADDRESS
       0110  00C0  R
00087  0112  8140                LD      A1,MEMADD             LOAD END ADDRESS FOR COMPUTATION
       0114  00C0  R
00088  0116  024E                LDK     A2,/4E                LOAD PROGRAM SIZE
```

```
0118  9908              SUR      A1,A2                 GIVES NEW START ADDRESS
011A  8584              LDR      A13,A1                SAVE NEW START ADDRESS
011C  8141              ST       A1,NEWADD             STORE NEW LOAD ADDRESS
011E  0122  R
0120  5702              RF(7)    *+4                   BYPASS NEWADD
0122  0000    NEWADD    DATA     0                     RESERVE ONE LOCATION FOR NEW ADDRESS
0124  190A              SUK      A1,10                 GET NEW END ADDRESS
0126  8141              ST       A1,MEMADD             AND STORE IN MEMADD
0128  00C0  R
              *
              *         THIS LOADS THE NEW START ADDRESS IN MEMADD+2
              *
012A  0200              LDK      A2,0
012C  8241              ST       A2,MEMADD+2
012E  00C2  R
              *
              *         THIS LOADS THE NEW END ADDRESS IN THE WRITE/READ ROUTINES
              *
0130  110A              ADK      A1,10
0132  8141              ST       A1,WRITE+8
0134  00E2  R
0136  8141              ST       A1,CONT+6
0138  00FA  R
              *
              *         THIS LOADS THE NEW ADDRESS OF MEMADD+2
              *
013A  1102              ADK      A1,2
013C  8141              ST       A1,TEST+2
013E  00D8  R
0140  8141              ST       A1,READ-2
0142  00F8  R
              *
              *THIS LOADS THE NEW ADDRESS OF PATEN
              *
0144  1102              ADK      A1,2
0146  8141              ST       A1,LDPAT+2
0148  00D4  R
014A  8141              ST       A1,READ+4
014C  00FE  R
              *
              *         THIS LOADS THE NEW ADDRESS OF TIMER
              *
014E  1108              ADK      A1,8
0150  8141              ST       A1,CONT+22
0152  010A  R
              *
              *         THIS ROUTINE DOES THE TRANSFER
              *
0154  83A0              LDKL     A11,40
0156  0028
0158  8D40    REP       MI       10,MEMADD
015A  00C0  R
015C  86A0              LDKL     A14,20
015E  0014
0160  96C1              ADS      A14,REP+2
0162  015A  R
0164  8D37              MSR      10,A13
```

```
00132  0166  95A0          ADKL      A13,20
       0168  0014
00133  016A  9BA0          SUKL      A11,10
       016C  000A
00134  016E  5C1A          RB(NZ)    REP
00135  0170  83A0          LDKL      A11,/207F
       0172  207F
00136  0174  9DA0          SUKL      A13,2
       0176  0002
00137  0178  83B7          STR       A11,A13
00138  017A  9DA0          SUKL      A13,64
       017C  0040
00139  017E  8F16          ABR       A13
00140                      END       START
```

SYMBOL TABLE

BEGIN	0000	R	COMP	00A6	R	CONT	00F4	R	INR	008C	R
LDPAT	00D2	R	LOAD	00CE	R	LOAD1	00D0	R	MEMADD	00C0	R
NEWADD	0122	R	PARAMS	00B2	R	PATEN	00C4	R	READ	00EA	R
REP	0158	R	START	009E	R	TEST	00D6	R	TIMER	00CC	R
WRITE	00DA	R									

Chassis	Card Slots	Models	Maximum Memory Size (16-bit words)	Width - 19 inches (483mm) Depth - 21.65 in. (550mm)	Height
M4	10	P856	32K		6 U 267mm
		P857 MMU FPP optional	32K		
		P857 MMU standard FPP optional	64K		
OPTIONAL					
M5	17	P856	32K		11 U 489mm
		P857 MMU, FPP	128K		

- The Memory Management Unit (MMU) is required for all systems with more than 32k words of memory.
- The Floating Point Processor (FPP) is available with any P857M system.
- 1 U = 44.45 mm (1.75 inches)

Figure 4-1 P856M/P857M Chassis Configurations

SECTION IV

MECHANICAL

4.1 GENERAL

The P856M/P857M System is available in either of two basic chassis (M4 or M5), with two different extension chassis (E1, E2) available for additional control-unit cards. The different basic-chassis configurations are shown in Figure 4-1. Each chassis contains a power supply, ventilation, printed circuits, and I/O cable connectors, as well as the logic-card slots. Each chassis is slide mounted in a 19-inch (483mm) rack. Each basic chassis includes a control panel (either complete or simplified) mounted on the front face. The chassis dimensions and installation data are provided in Figure 4-2.

4.2 WIRING AND CABLING

4.3 General

The cable connections on the P857M basic chassis and the extension chassis, are shown on Figure 4-3. The uses of the different connectors on the cards are shown on Figure 4-4. Lists of signals and pin numbers for the card connectors are provided in the following tables :

Table	Connector
4-1	GP Bus connector, IOM-IOB
4-2	CPU-A connector-1 (V24 CU)
4-3	Connector-3 (CPU, Memory, IOP, CU)
4-4	CPU-A connector-5
4-5	Control Panel Connector
4-6	IOP Connectors 4, 5 (Break)
4-7	Extension Connectors AIE/TAIE

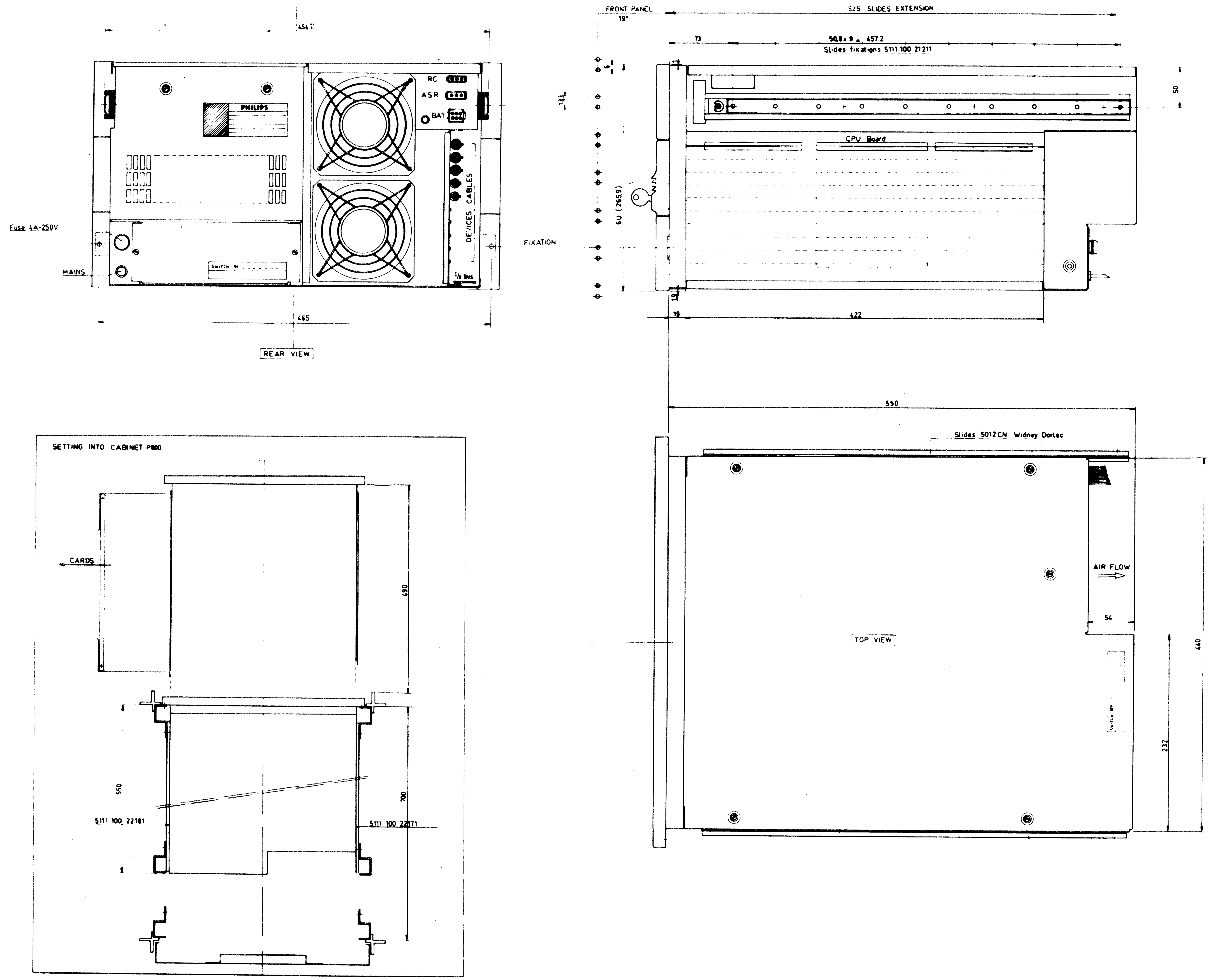


Figure 4-2A M4 Chassis Installation Data

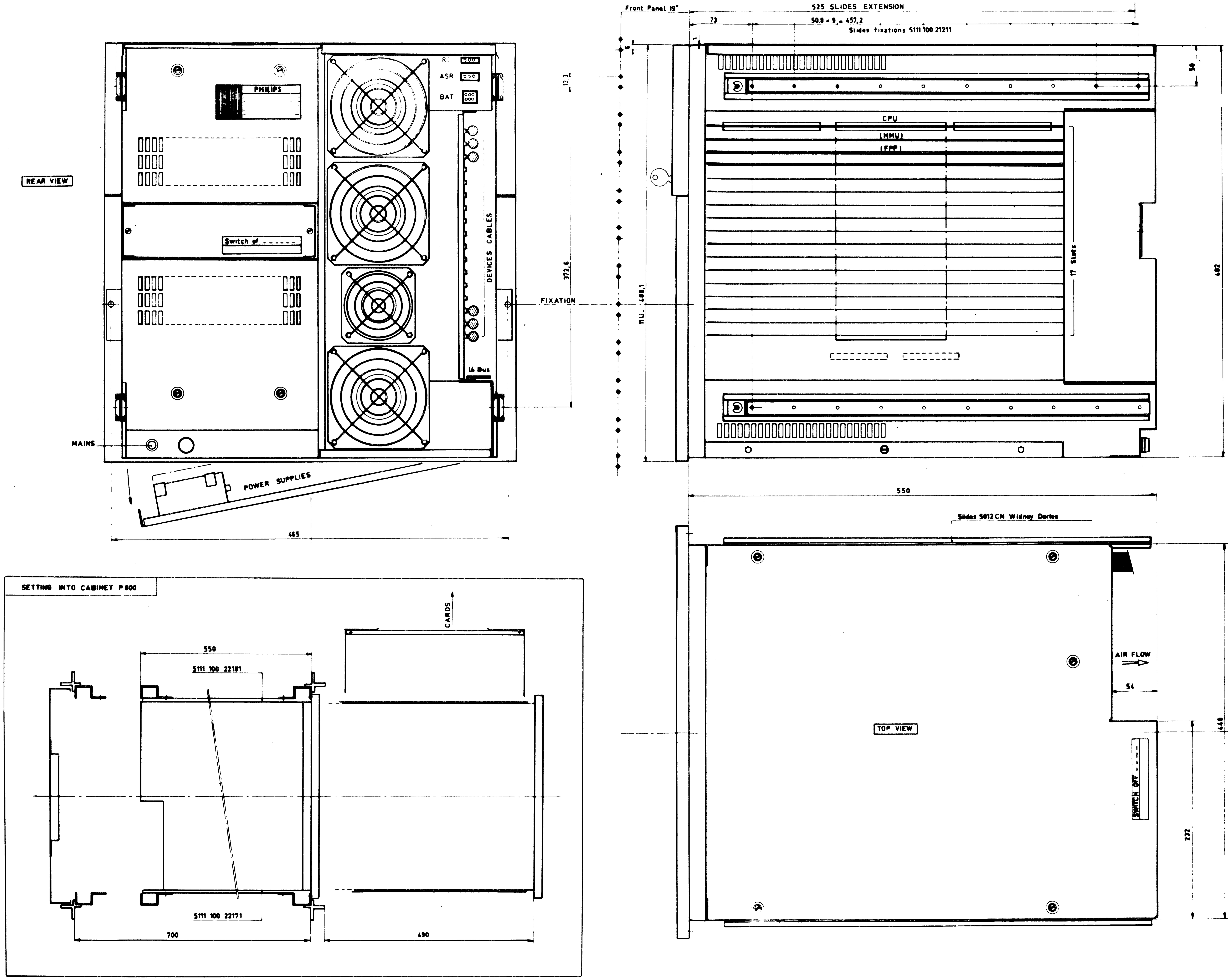
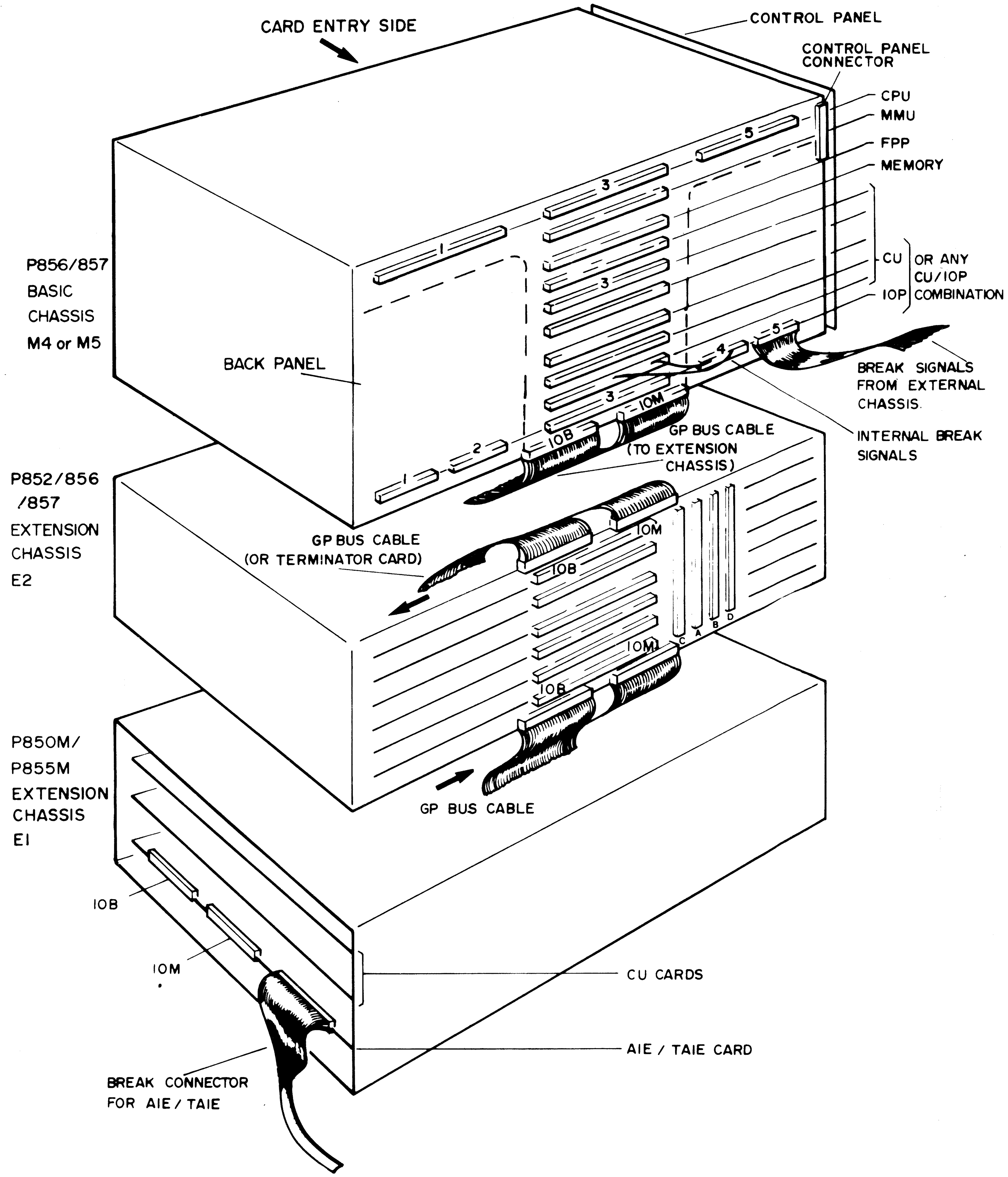


Figure 4-2B M5 Chassis Installation Data



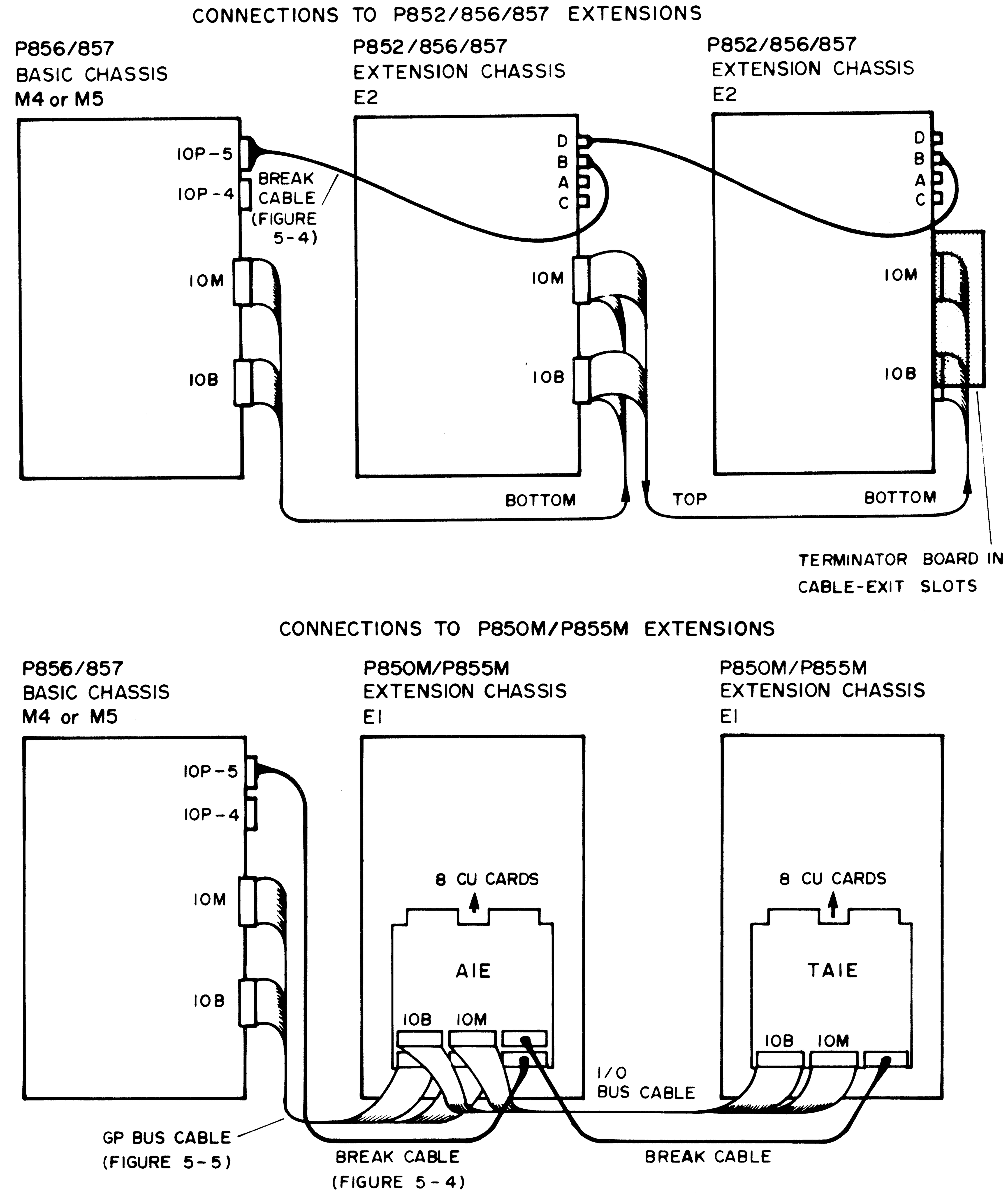


Figure 4-3 P856M/857M Basic/Extension Chassis Connections

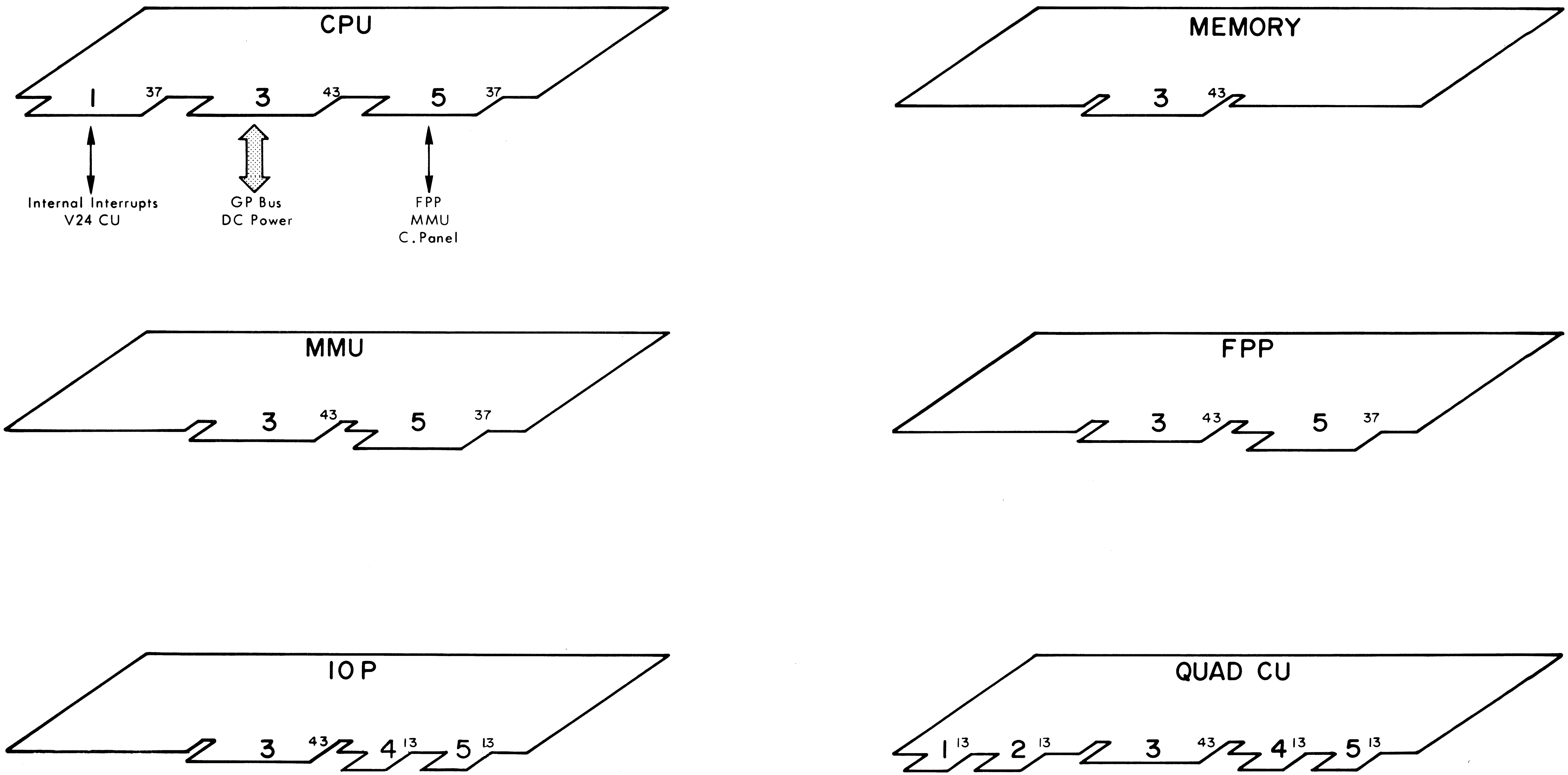
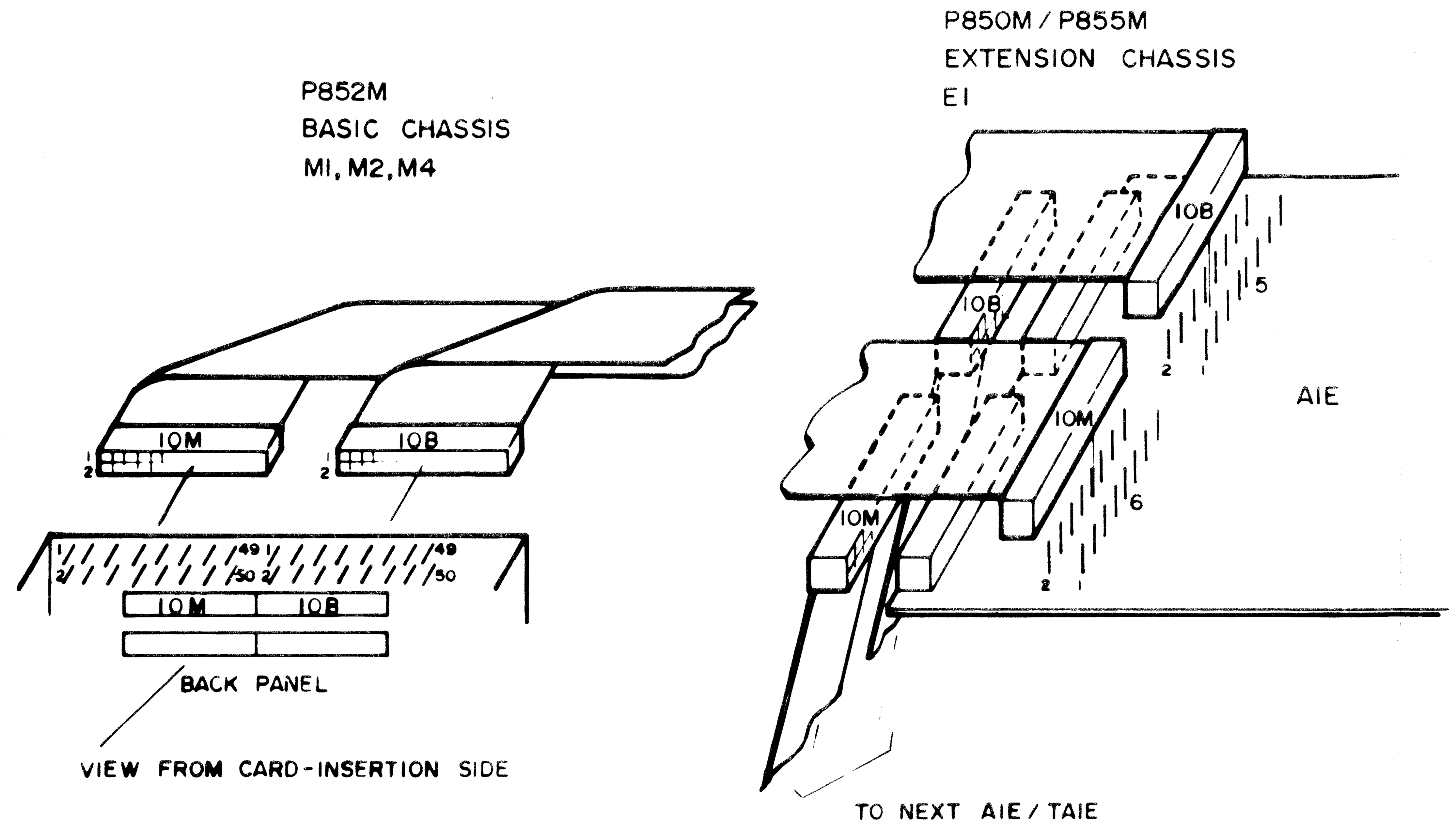
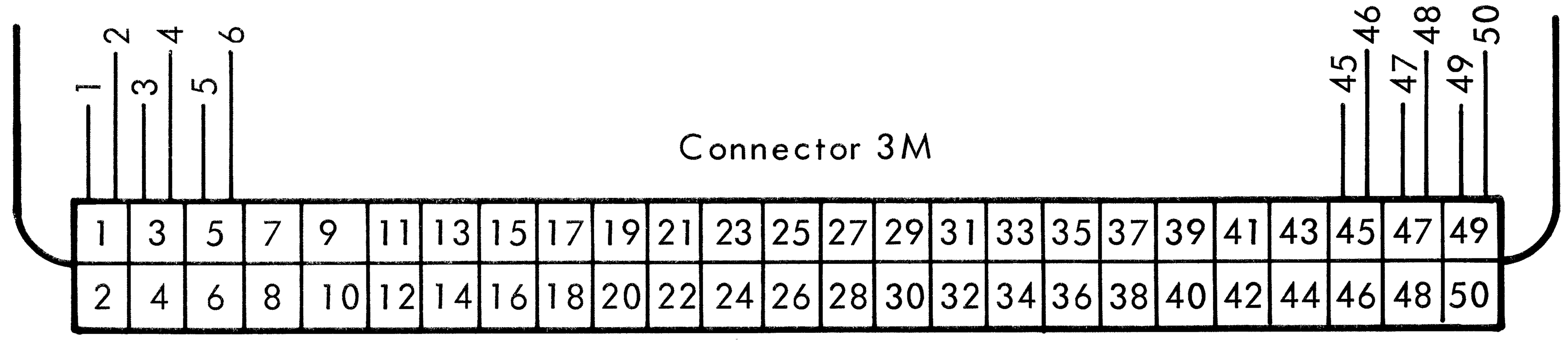


Figure 4-4 Circuit-Card Connector Uses

Table 4-1 GP Bus Connector IOM, IOB





Connector viewed from cable side

Connector I O M				Connector I O B			
N° Pin	Signal	N° Pin	Signal	N° Pin	Signal	N° Pin	Signal
1	M A	26	M C	1	M C	26	BIO 05N
2	MAD 04	27	CLEARN	2	RSLN	27	M 3
3	M A	28	M C	3	M C	28	BIO 04N
4	MAD 03	29	M C	4	PWFN	29	M B
5	M A	30	TPMN	5	M B	30	BIO 03N
6	MAD 08	31	M C	6	BIO 15N	31	M B
7	M A	32	M C	7	M B	32	BIO 02N
8	MAD 09	33	TMPN	8	BIO 14N	33	M B
9	M A	34	M C	9	M B	34	BIO 01N
10	MAD 10	35	M C	10	BIO 13N	35	M B
11	M A	36	TMEN	11	M B	36	BIO 00N
12	MAD 11	37	M C	12	BIO 12N	37	M B
13	M A	38	M C	13	M B	38	BIEC5
14	MAD 12	39	TRMN	14	BIO 11N	39	M C
15	M A	40	M C	15	M B	40	SCEIN
16	MAD 13	41	M C	16	BIO 10N	41	M C
17	M A	42	Spare	17	M B	42	BIEC3
18	MAD 14	43	M C	18	BIO 09N	43	M C
19	M A	44	Spare	19	M B	44	BIEC4
20	MAD 15	45	M C	20	BIO 08N	45	M C
21	M A	46	Spare	21	M B	46	BIEC1
22	ACN	47	M C	22	BIO 07N	47	M C
23	M C	48	Spare	23	M B	48	BIEC2
24	ARN	49	M D	24	BIO 06N	49	M C
25	M C	50	5V	25	M B	50	BIEC0

4.4 Operator I/O Device

The operator's input/output device is connected to the CPU-integral Serial Control Unit via CPU connector 1. The maximum cable length between the device and the CPU is 20 meters. The operator's interrupt does not use the encoded BIEC lines, but is connected from the Serial Control Unit directly to the internal-interrupt inputs (see Figure 1-3A).

4.5 Interface Signals

All interface signals between the CPU and the units (Serial CU, MMU, FPP, and interrupts) are listed in Table 4-8. The GP Bus signals are described in Section II. All these signals are also included in the signal lists for the connectors where they are used.

4.6 Interrupts and Breaks

The use of interrupts and breaks, and their interconnections, are described in Section I and shown in Figure 1-2.

4.7 CARDS

The complete CPU and the V24 Serial Control Unit are mounted on a single, multi-layer printed circuit card. The circuit-card locations within the chassis are shown in Figure 4-3. The CPU card fits in a dedicated slot at the top of the chassis. The MMU and FPP (P857 only), and memory cards fit in the next three dedicated slots. The MMU and FPP must be close to the CPU card. Both the MMU and the FPP have some discrete wiring connections to the CPU which are used to increase operating speed. If either the MMU or FPP are not used, a memory card may be placed in that dedicated slot. Card connector uses are shown in Figure 4-4. Circuit locations and parts lists are provided at the end of this section.

4.8 INTEGRATED CIRCUITS

A complete list of integrated circuits (ICs) and a guide which shows the IC symbols, input and output polarities, control codes, and pin numbers is provided in Appendix A.

4.9 Read Only Memories (ROMS) and PLA

The ROMs and the Programmable Logic Array (PLA) circuits are pre-loaded with special contents, or codes, which cannot be shown on the general IC-diagram drawing. The following list is a directory to the logic descriptions that use these circuits :

Circuit	Logic		Paragraph
6200 1024-bit ROM	LL	IPL	2.66
7488A 256-bit ROM	FF	A,D,L Command	2.47
	NN	CR code selection	2.76
8205 4096-bit ROM	BB	Microcommand Control Store	2.37
8576 96-code PLA	AA	Instruction Decode	2.42

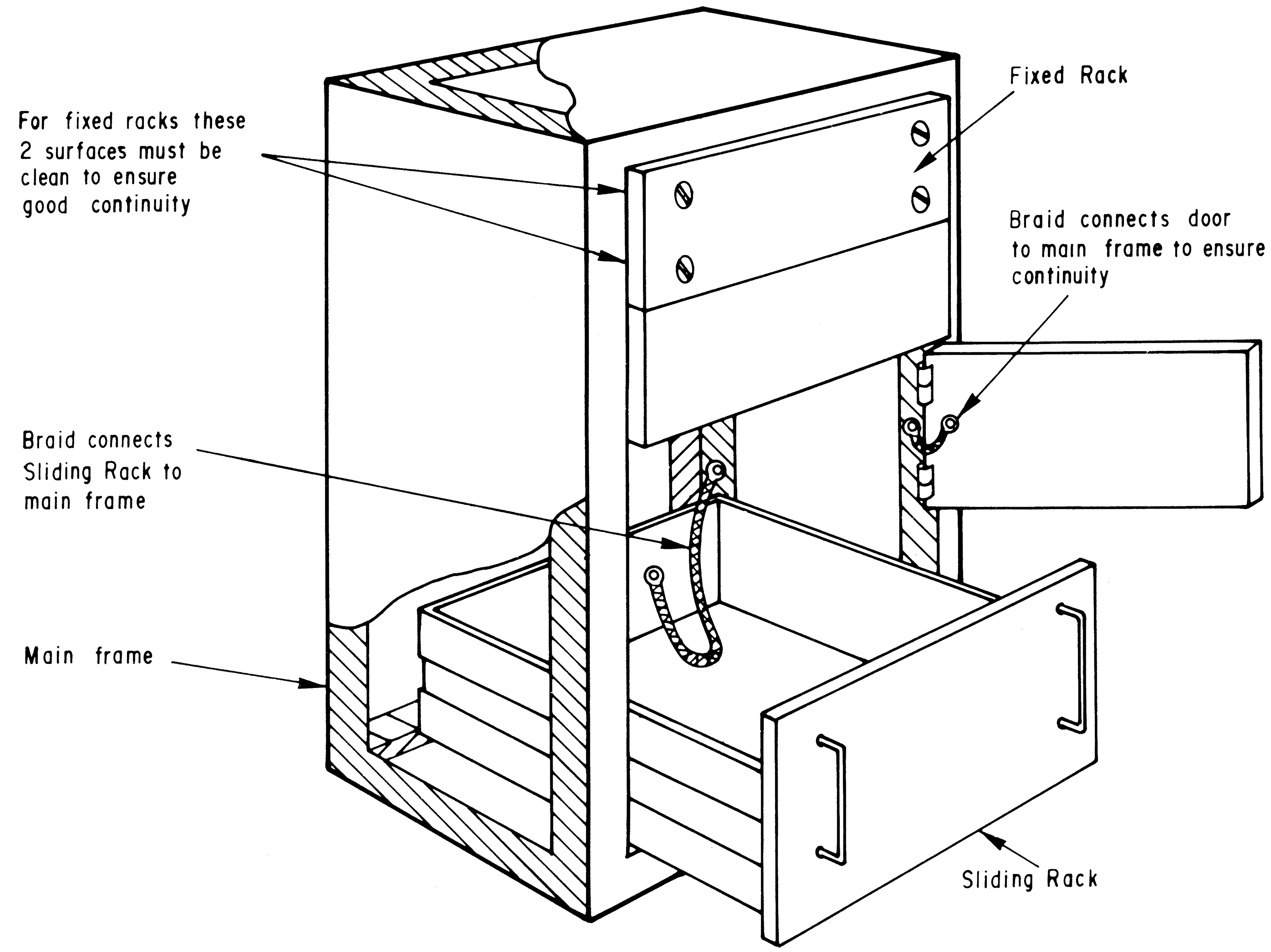
4.10 RULES FOR CONNECTING GROUNDS IN A SYSTEM

These rules must be observed for all installations to ensure that external interference is reduced to a minimum and that the electrical safety regulations are complied with.

4.11 Grounding for Cabinets and Racks

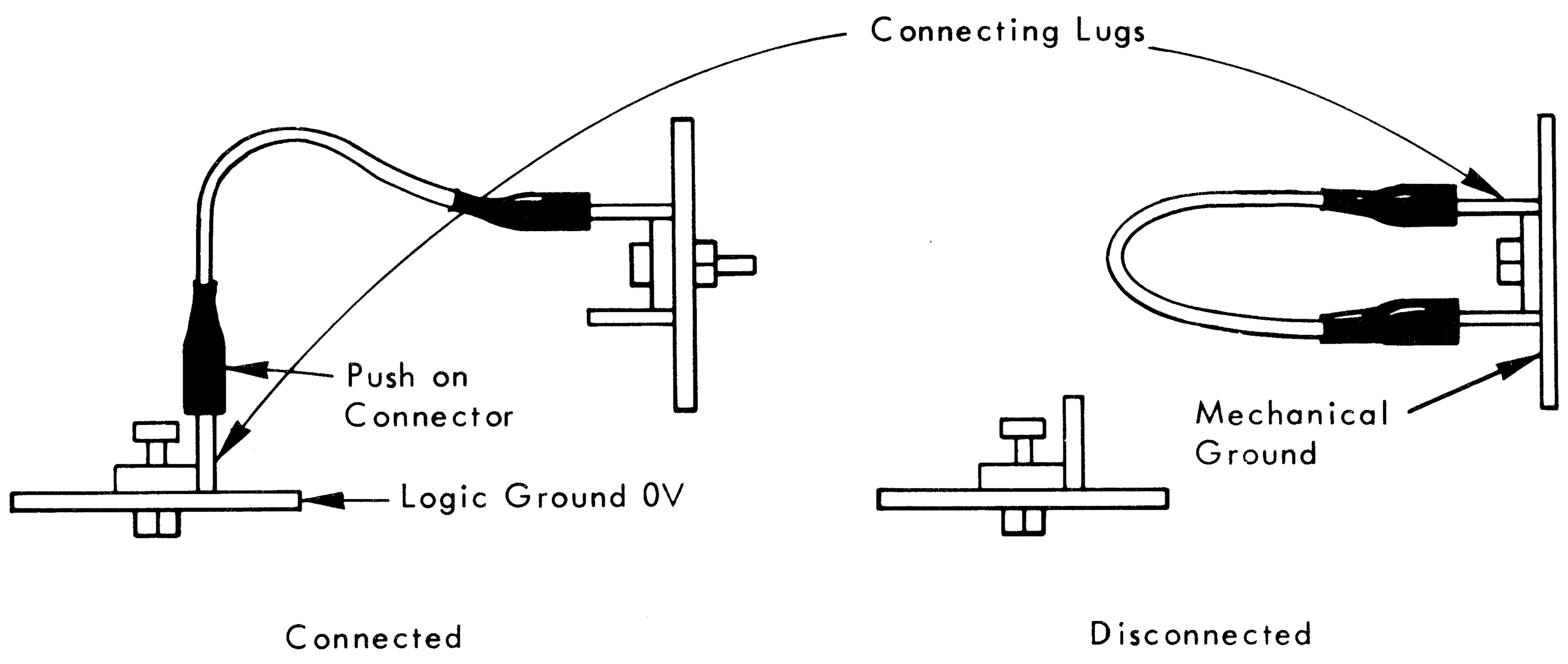
The following rules should ensure that the cabinets and racks have good ground continuity.

- Fixed Racks — The rack is fitted to the cabinet mainframe by screws. A good continuity between cabinet and rack is ensured by keeping the mating surfaces clean.
- Sliding Racks — A rack mounted on telescopic rails does not have a reliable ground path so it is essential that an electrical link is made using metal braid. Note that it is not acceptable to use the ground conductor of the mains lead or cable shielding for the ground connection.
- Cabinet Panels and Doors — All doors and panels must have a good electrical link with the mainframe and in particular the doors must be connected with metal braid.

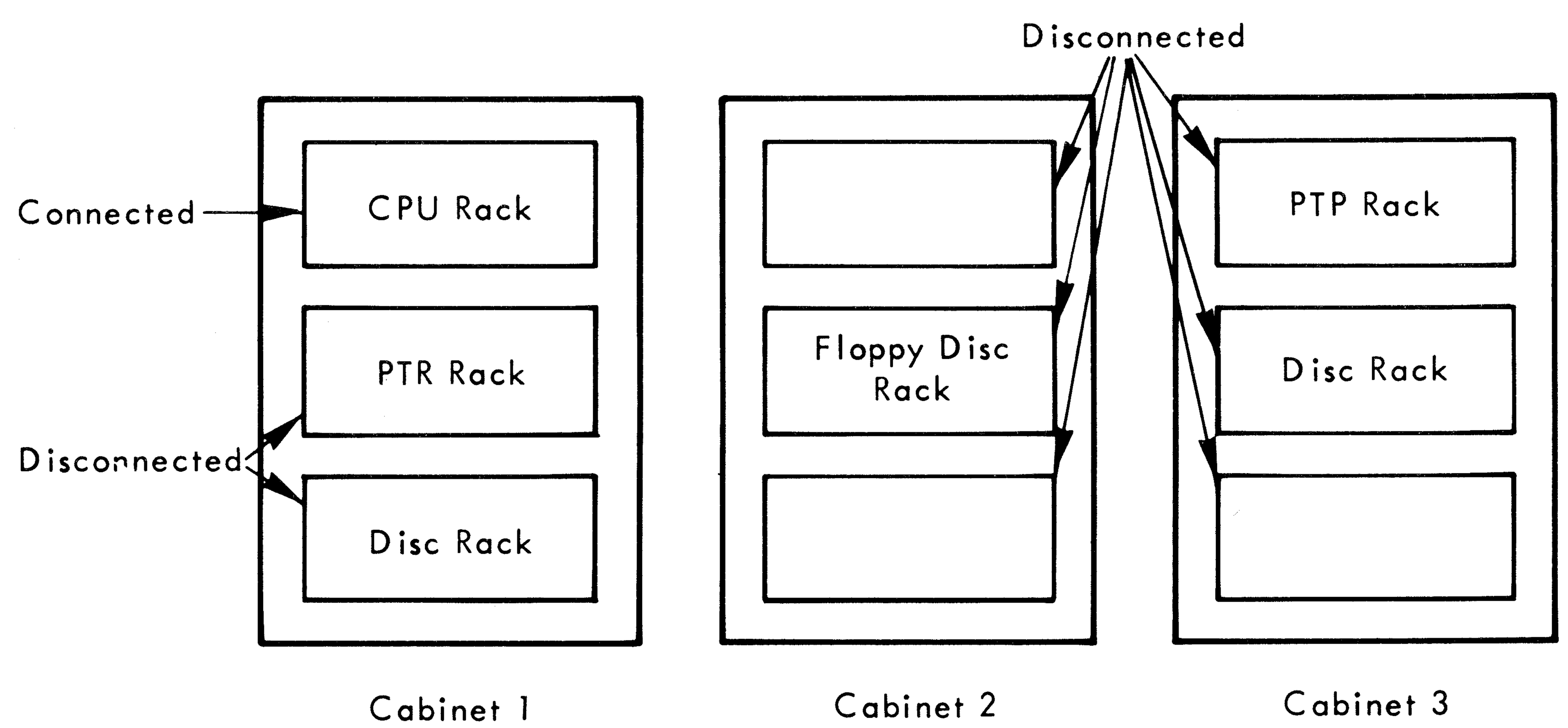


4.12 Logic Ground and Mechanical Ground

Logic ground and mechanical ground points are provided for each rack. They are situated close to each other and may be connected or unconnected.



A system may use several cabinets each containing a number of racks but only one rack may have this link connected. All other rack links are unconnected.

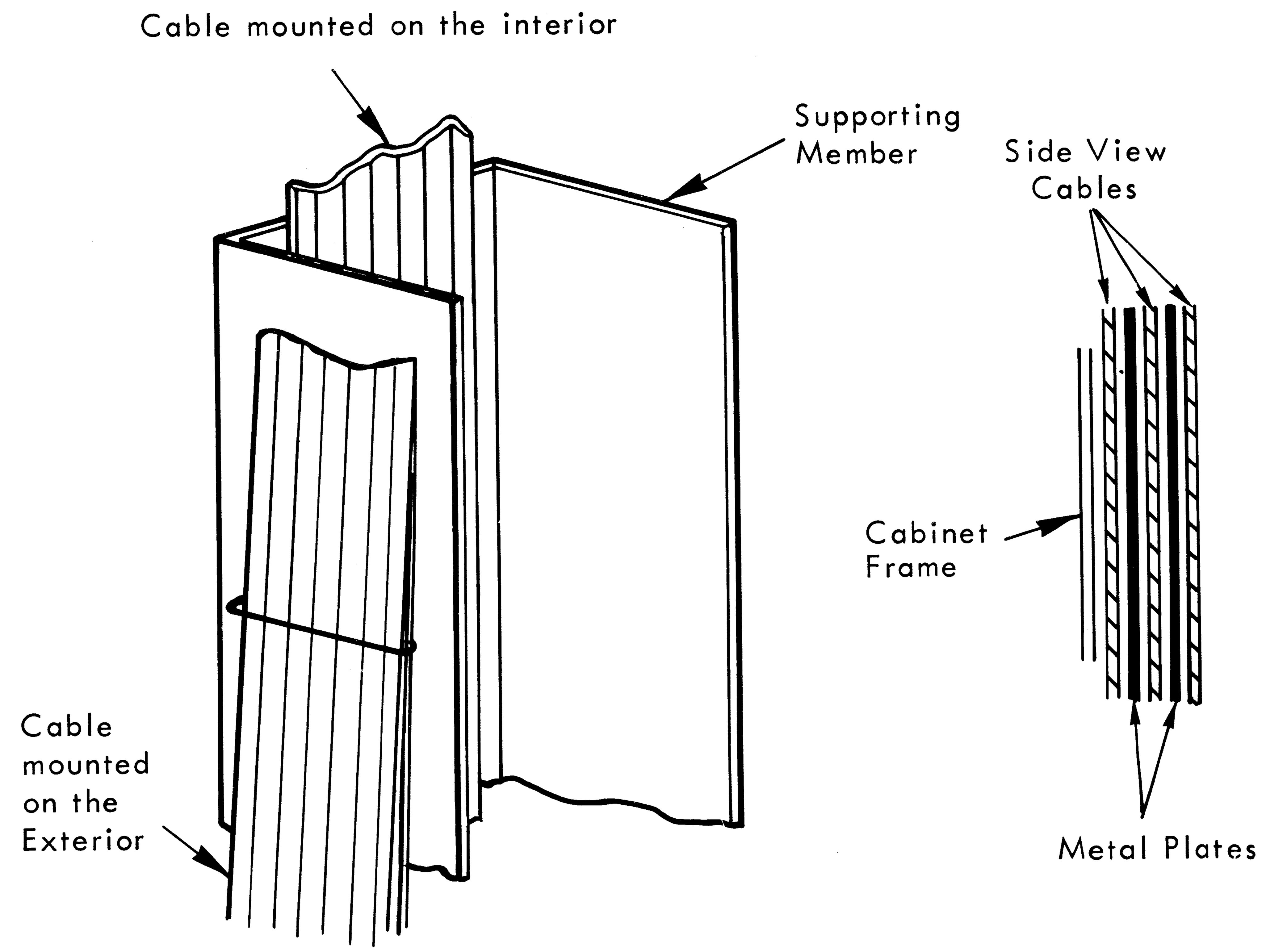


In a normal configuration the CPU rack is connected and all other racks are unconnected. If more than one CPU is being used in the system then the user decides on which CPU rack to have the link connected.

4.13 Flat Cables

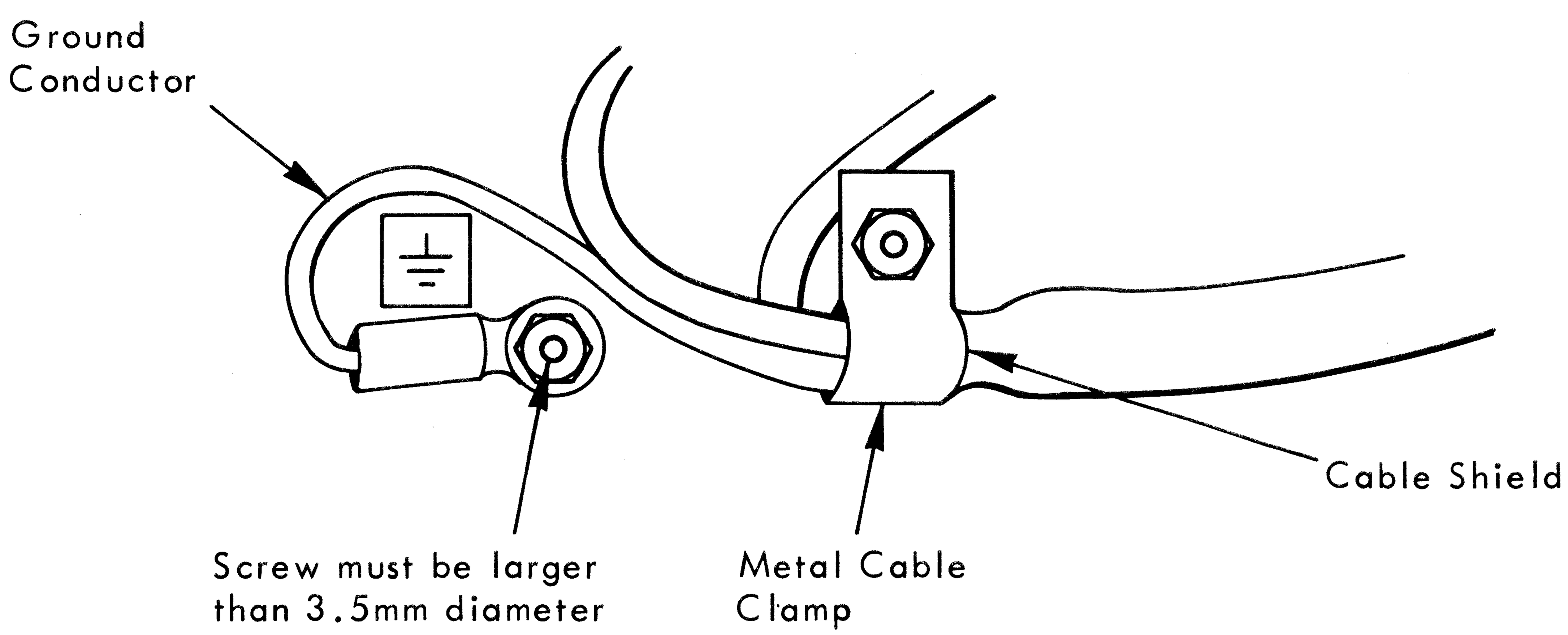
Flat cables without shielding may cause electromagnetic fields to be set up in the equipment so the use of non-shielded flat cables should be avoided. If it is essential to use non-shielded flat cables then special precautions must be taken to avoid these fields being set up.

- If channel type supporting members are in the cabinet then they should be used as they provide a shield around the cable.
- If these supporting members are not available then the cables should be run separately, either clamped flat against the panels or against the outside of other supporting members.
- Mounting flat cables one on top of the other should always be avoided. If this method must be used then a metal plate should be inserted between the flat cables, suitably supported and electrically connected to the cabinet.



4.14 Mains Cables

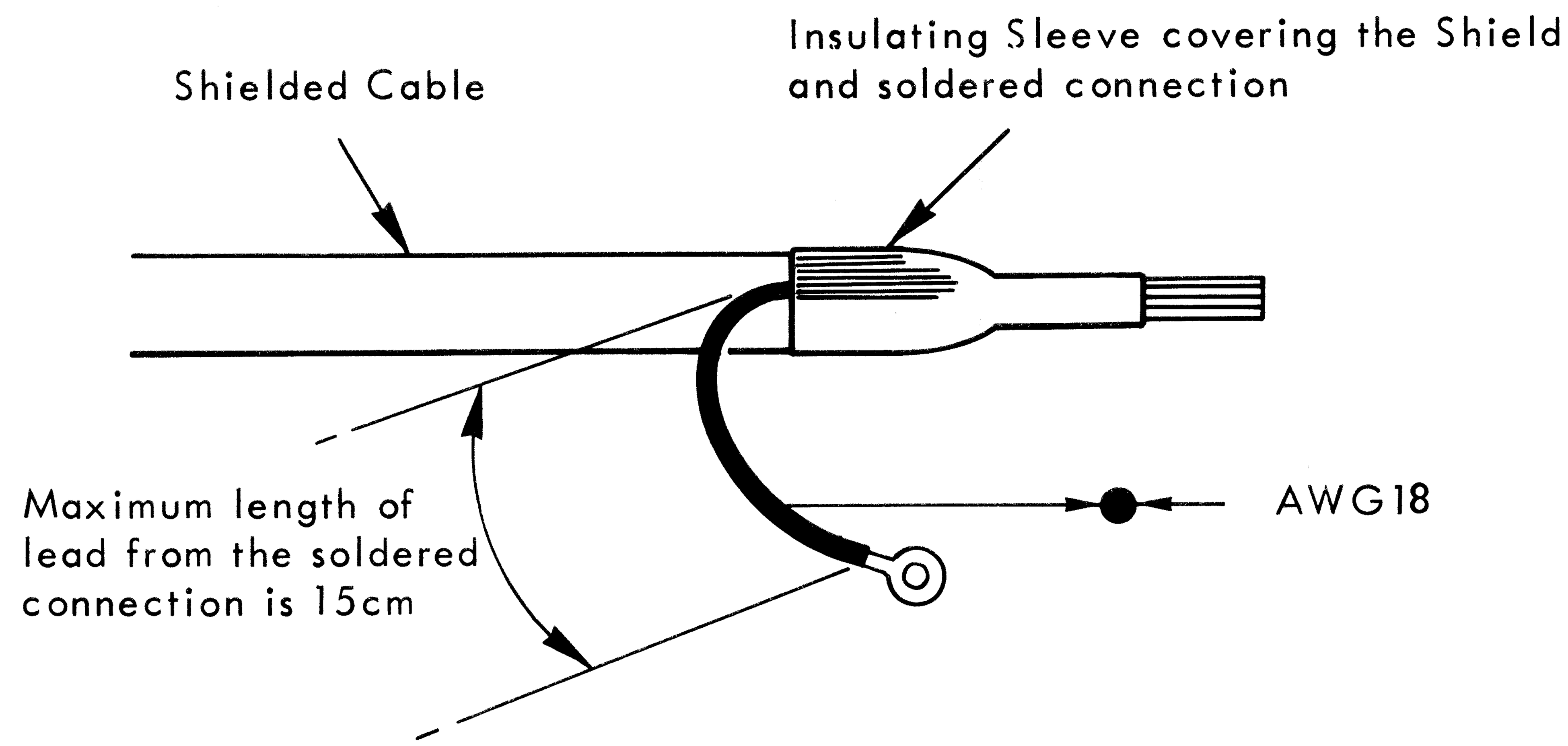
When planning an installation an area along the mainframe should be reserved for the mains cables and all other cables should be kept as far as possible from the mainframe. It is recommended that the cable be shielded either before or after filtering and that this shielding be connected to a mechanical ground at each end of the cable. The mains cable must be secured to the chassis by a metal clamp at the point where the shielding extends out of the cable sheath. Note that the screw fixing the ground conductor to the chassis must be as close as possible to the metal cable clamp.



4.15 General Rules for Connecting Shielded Cables

All twisted pairs must be contained inside a shielded cable and this shield must be connected to mechanical ground at each end. Specific lead dimensions are given for some peripherals but as a general rule the following dimensions should be adhered to.

- the lead must be as short as possible and never exceed a maximum length of 15cm.
- Connection to the shielding must be soldered and connection to the mechanical earth must be with either an eye-lug (for a 3mm screw) or a push-on type connector (Faston 6.35 or equivalent).
- The lead cross section must have a minimum guage of AWG18.



4.16 Special Rules for Connecting Shielded Cables

For some devices the lead dimensions for connecting the shielded cables at the device and at the CPU are different to the general rules. These devices are listed in the following table.

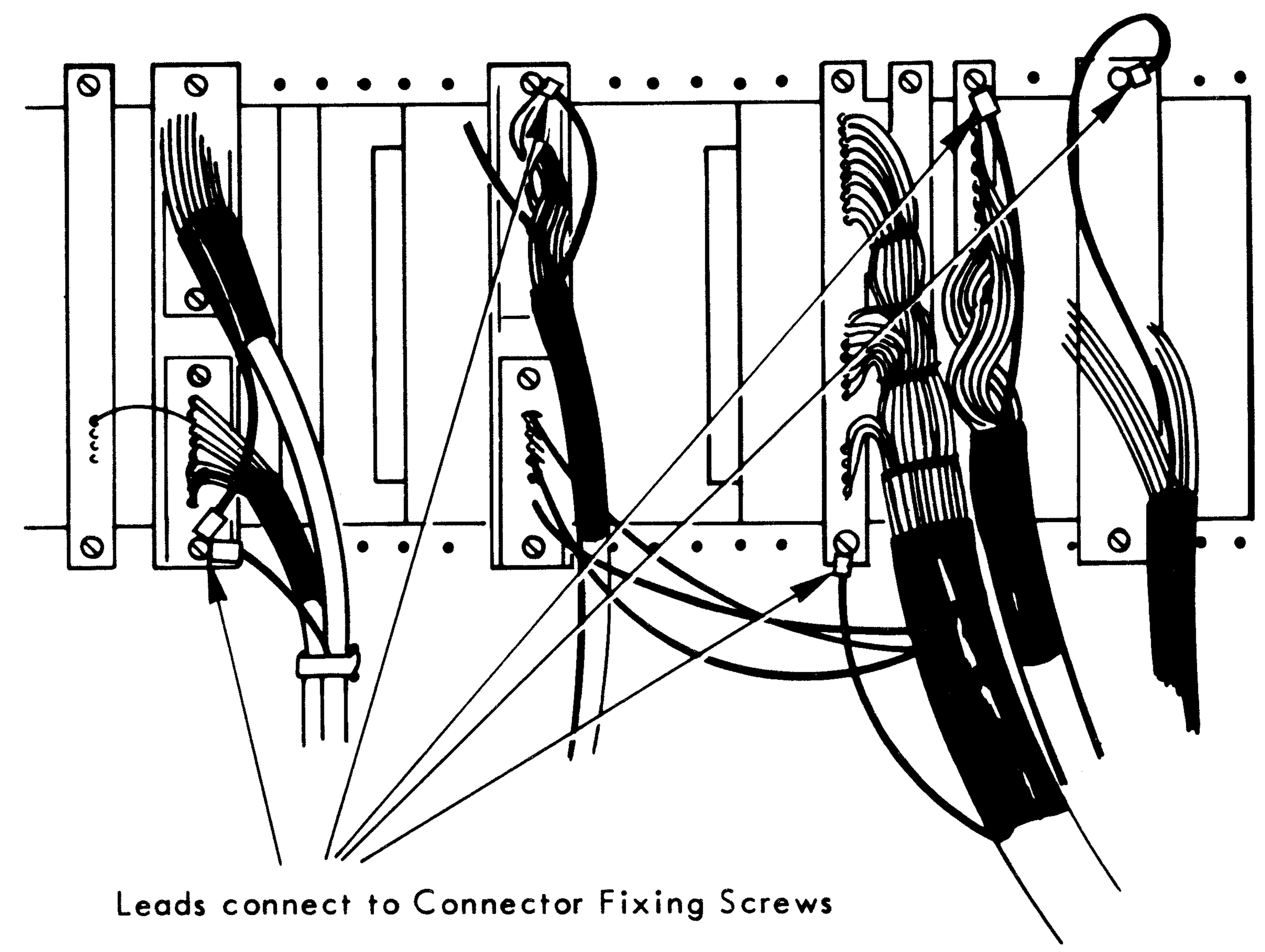
Table of Special Lead Dimensions

	At the Device			At the CPU		
	AWG	Max Length	Eye Lug	AWG	Max Length	Eye Lug
Mag Disc X1215/16	16	15cm	4mm	16	15cm	3mm
Mag Disc 9760/9762	16	08	3	16	10	3
Line Printer X1415/25	16	10	3	16	10	3
PER 3100 (V24 Interface)	16	15	4	16	10	3
PER 3100 (Curr. Loop Int.)	16	15	4	18	03	Molex pin 3
ASR33 (V24 Interface)	16	10	3	16	10	3
ASR33 (Curr. Loop Int.)	18	03	Molex pin 6	18	03	Molex pin 3
Tape Reader 2540EP	16	10	3	16	10	3
Card Reader CM300L	16	15	4	16	10	3
Tape Punch 4070	16	10	4	16	10	3
Display P817	16	10	4	16	10	3
	At the Terminal Box			At the CPU		
Remote Control Cables	18	10	-	18	03	Molex centre pin
	At the Extension Rack			At the CPU		
Break Cables	16	10	3	16	15	3
	At the Cassette Rack			At the CPU		
Break Cables	16	10	3	16	15	3

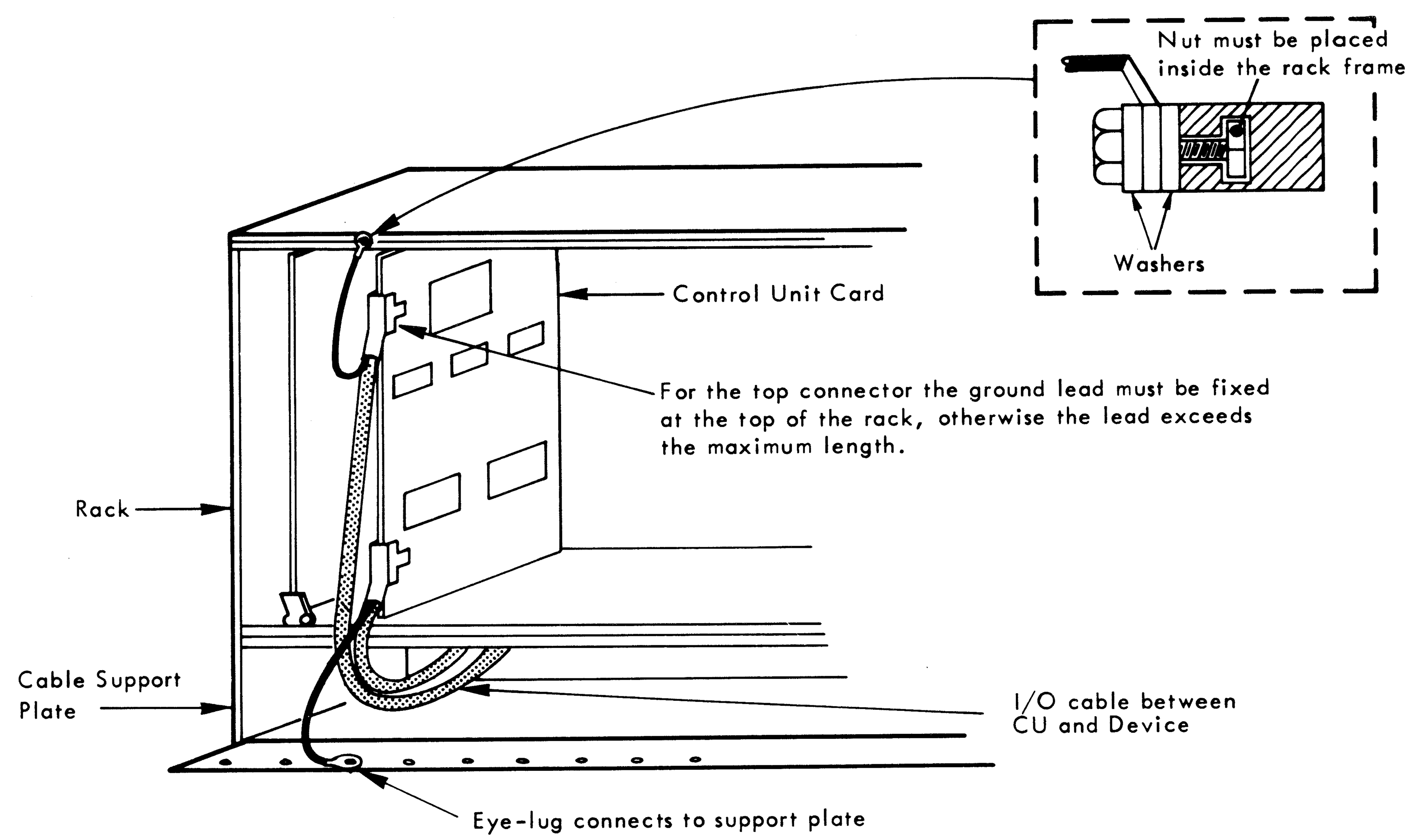
4.17 Connecting the Ground Lead in a Cabinet or Rack

The connection of ground leads for a Cabinet or Rack depends on the type of installation. In the following examples different types of ground connections are given.

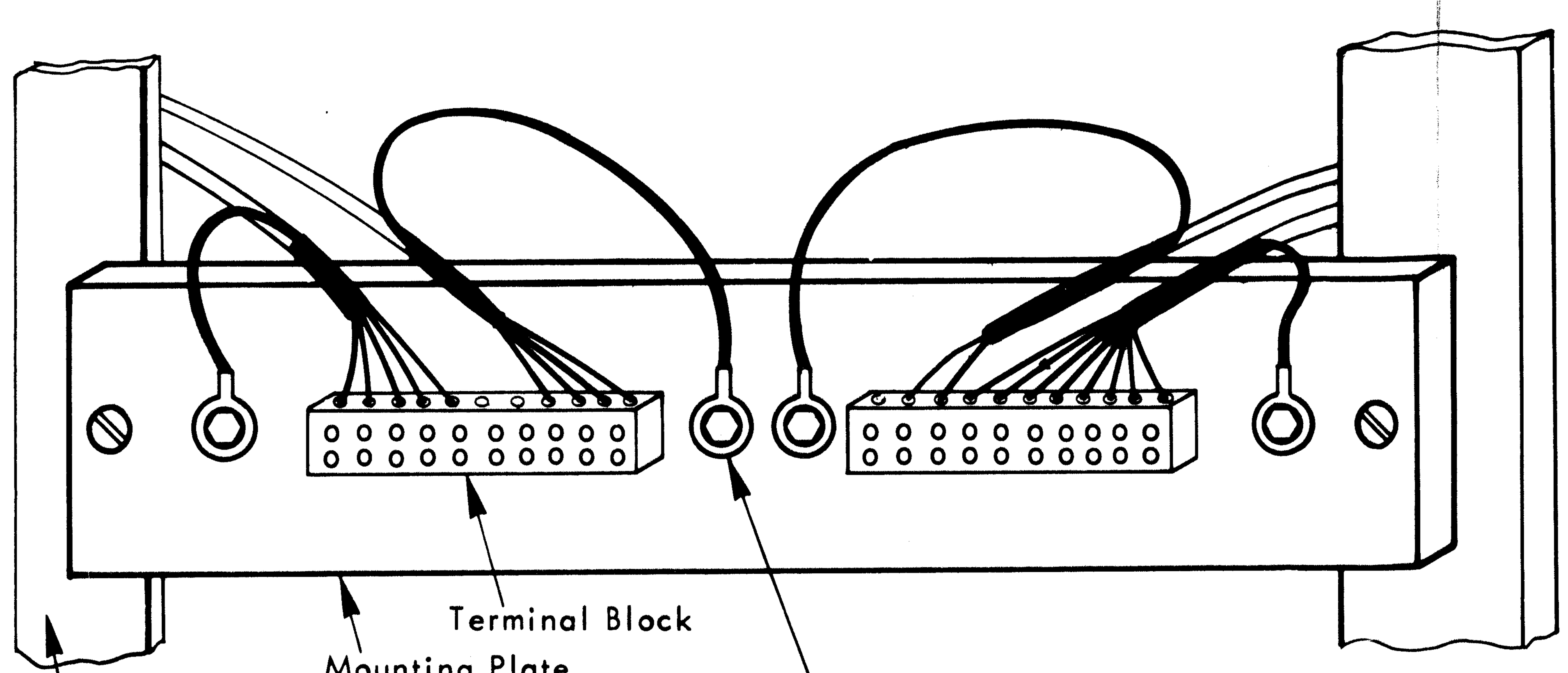
Example using Connector Fixing Screws



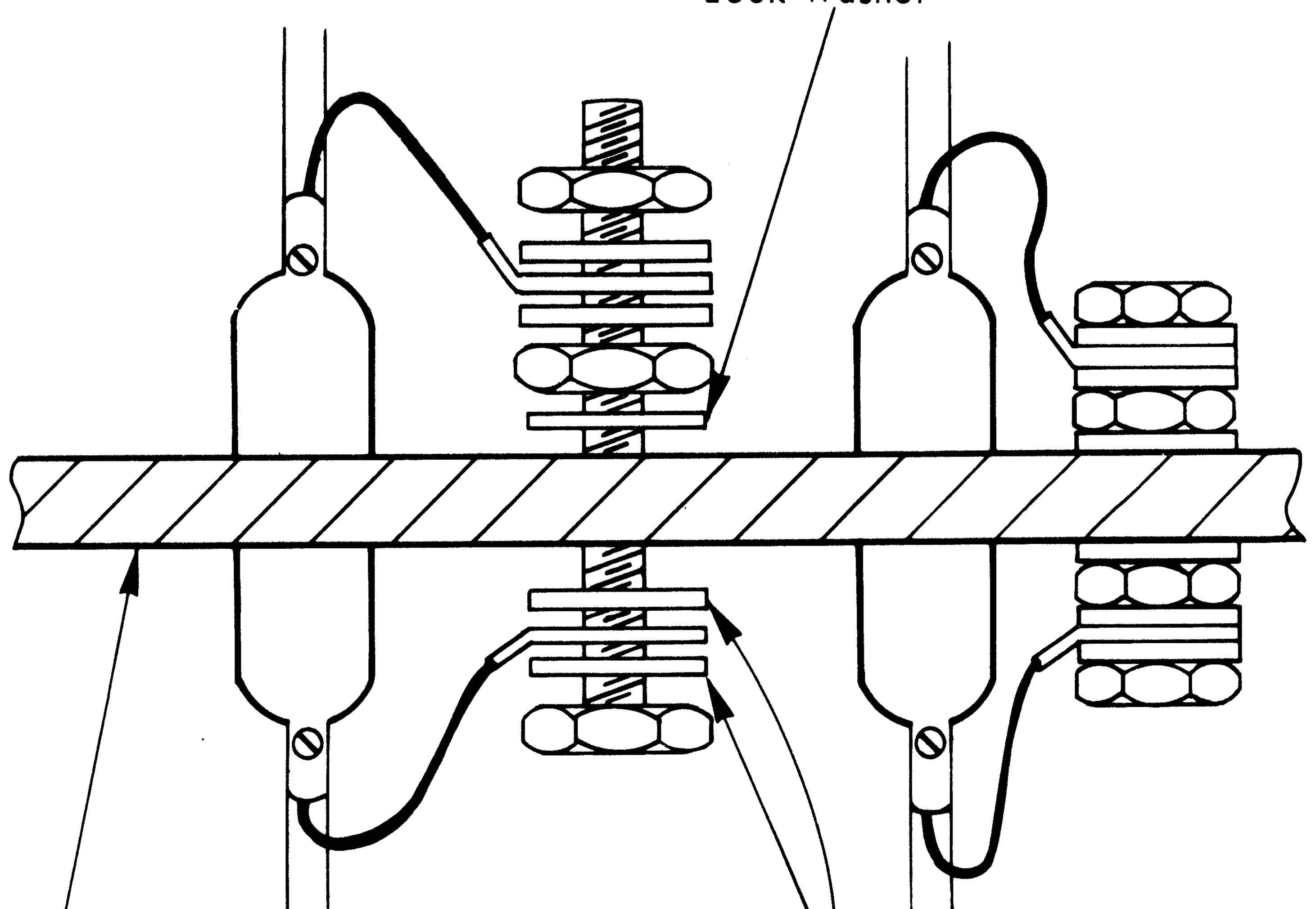
Example using Cable Support Plate



Example using Terminal Blocks



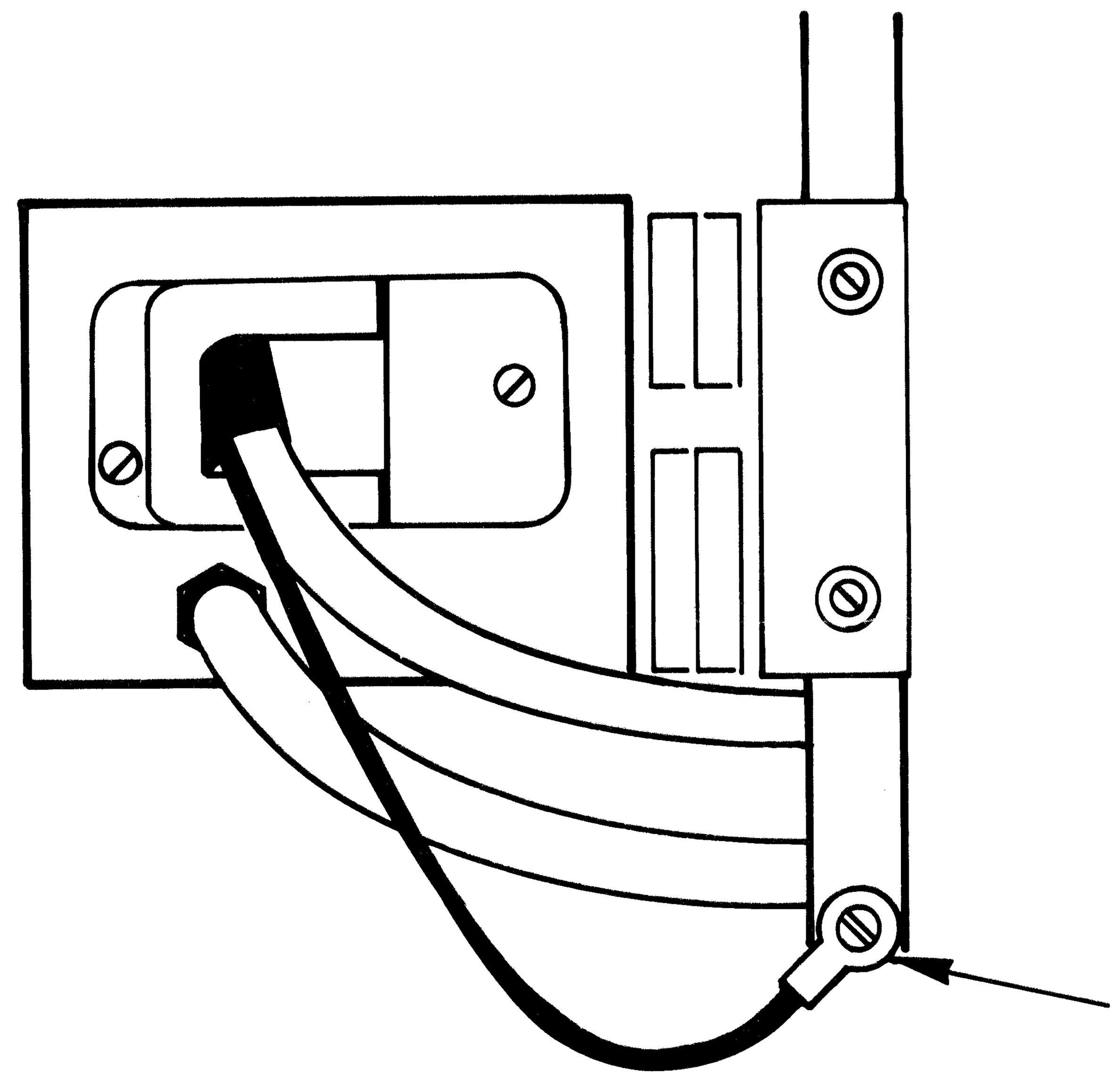
Example using Connectors (Top View)



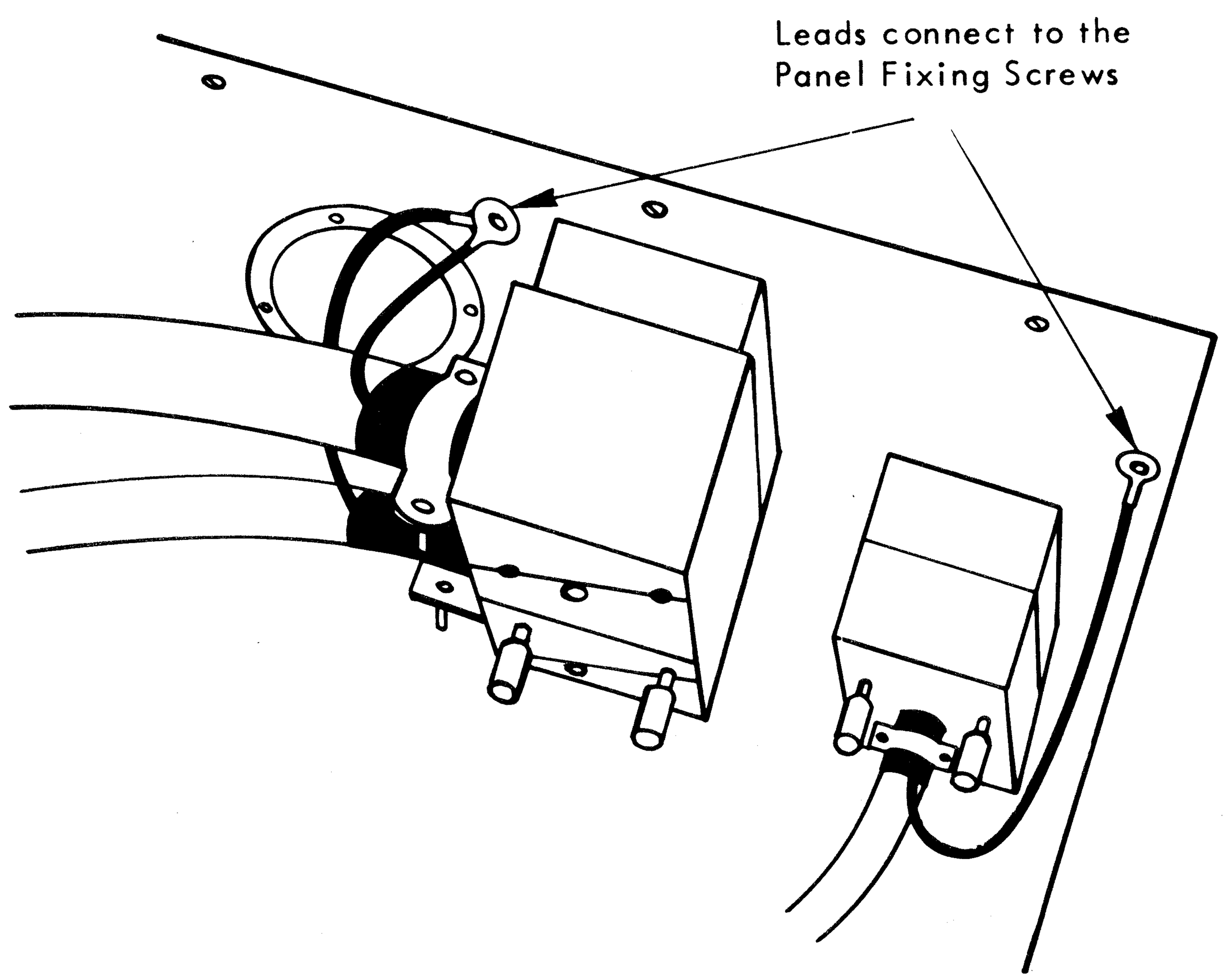
4.18 Connecting a Ground Lead at the Devices

The following illustrations show how a ground lead is connected at some peripheral devices.

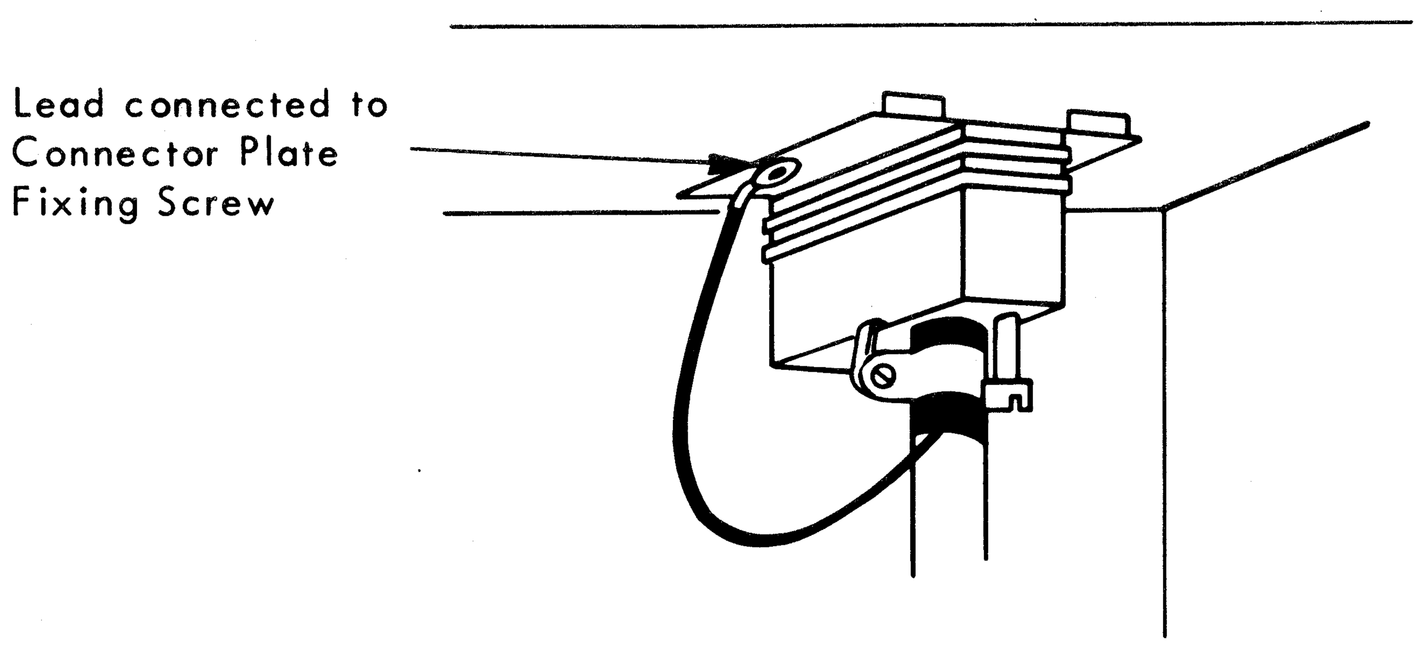
Connecting to the Mag. Disc Control Units (CDD X1215 and X1216)



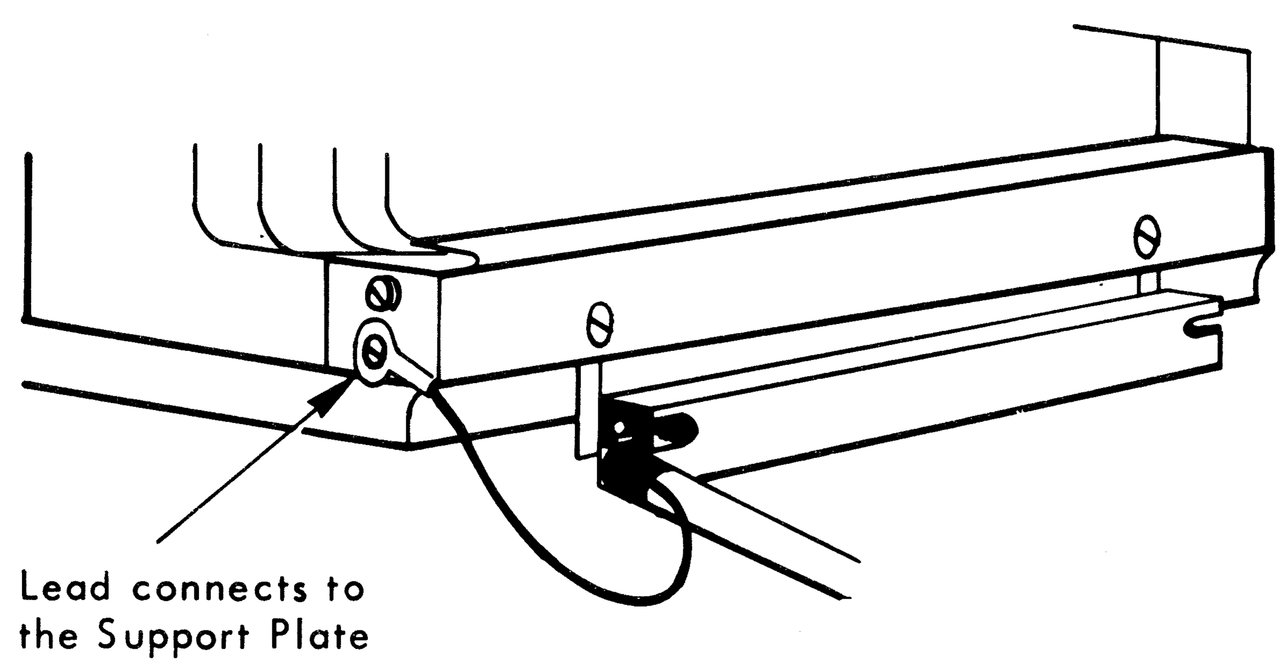
Connecting to the Mag Disc Control Units (CDD CDC 9760 and 9762)



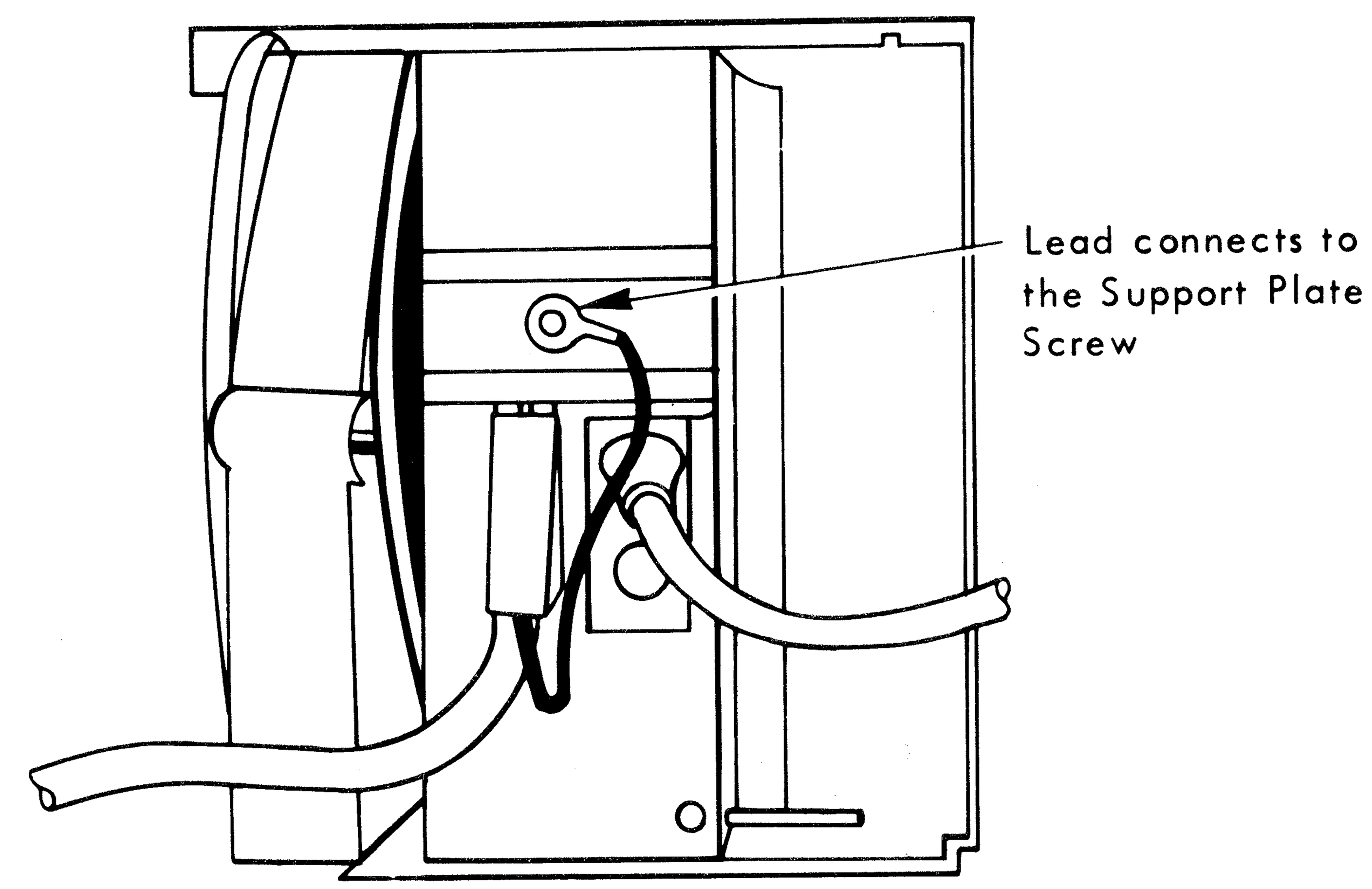
Connecting to the Line Printers (X1415 and X1425)



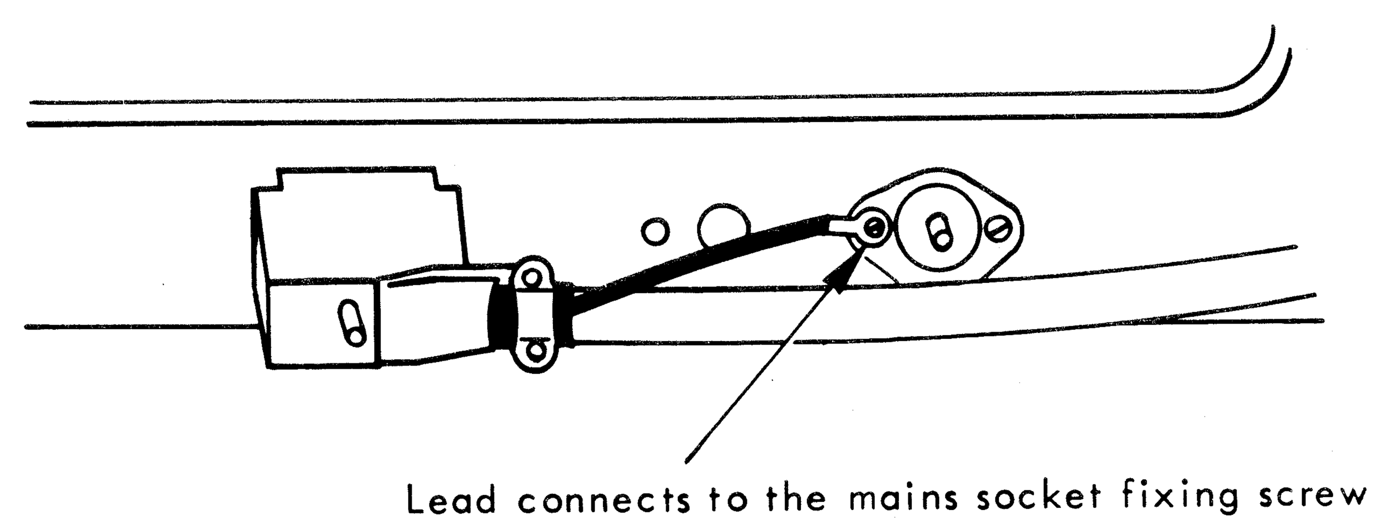
Connecting to the Paper Tape Reader (Digitronics 2540 EP)



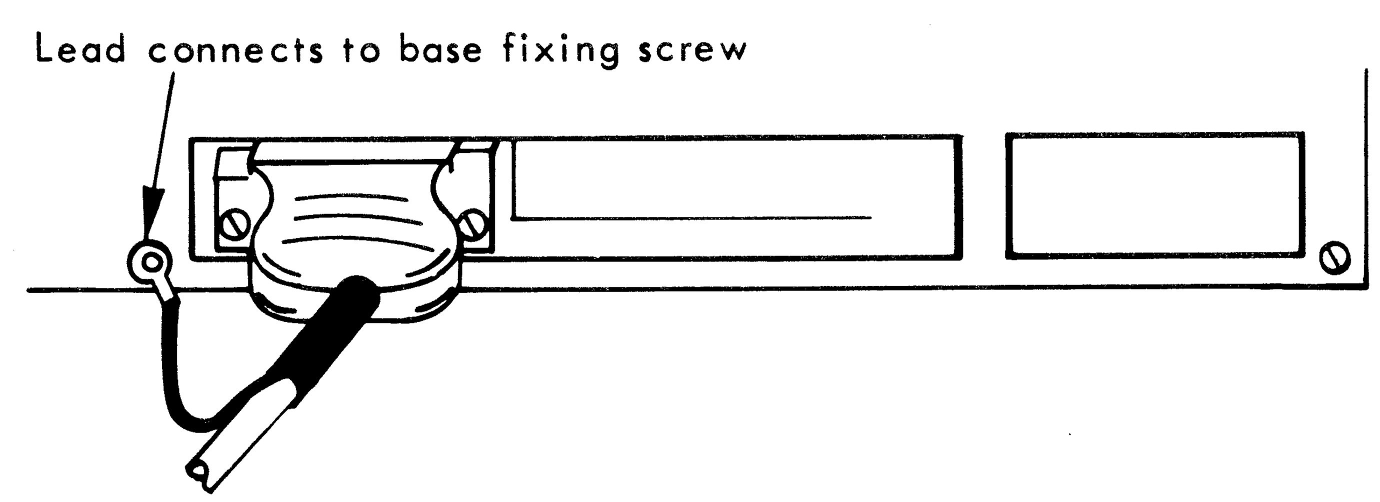
Connecting to the Paper Tape Punch (Facit 4070)



Connecting to the Card Reader (CM300L)

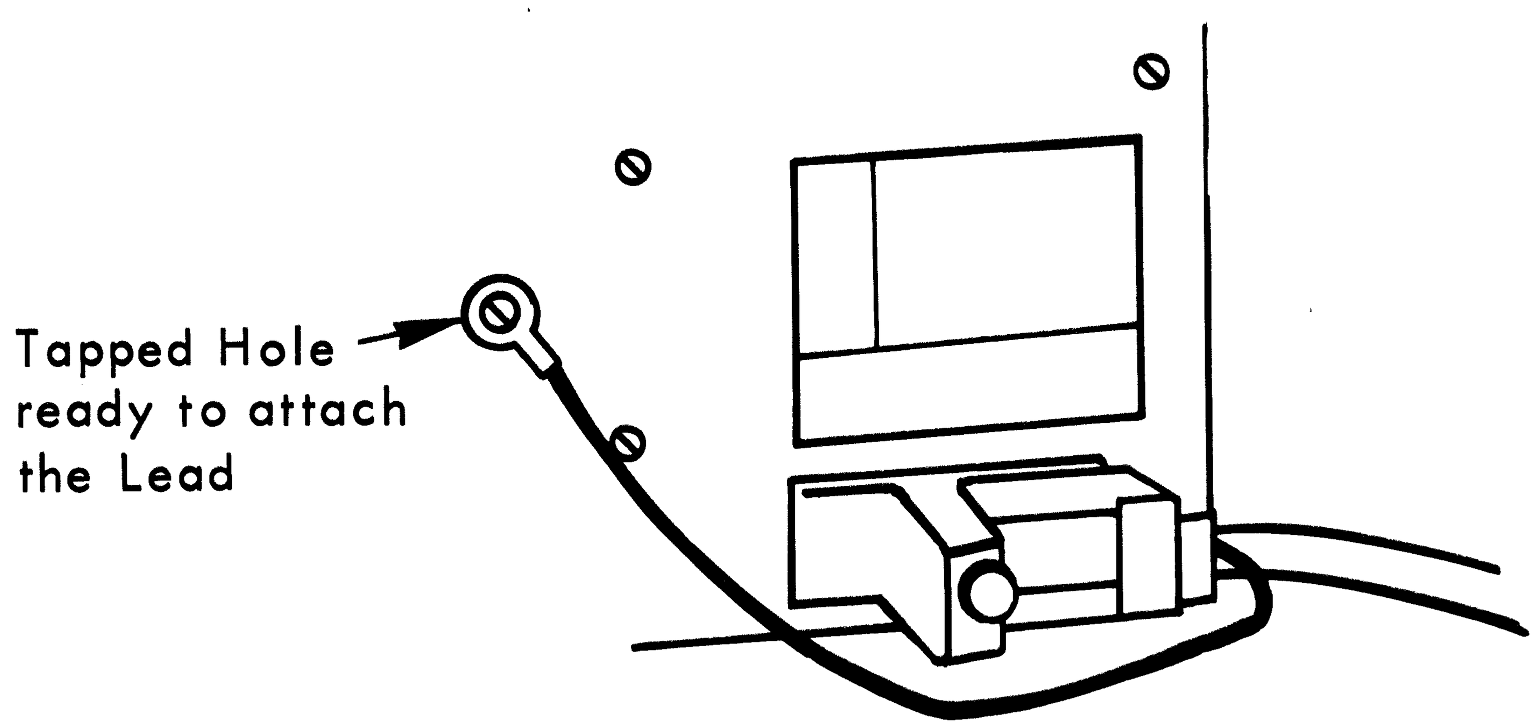


Connecting to the Display Console (P817)



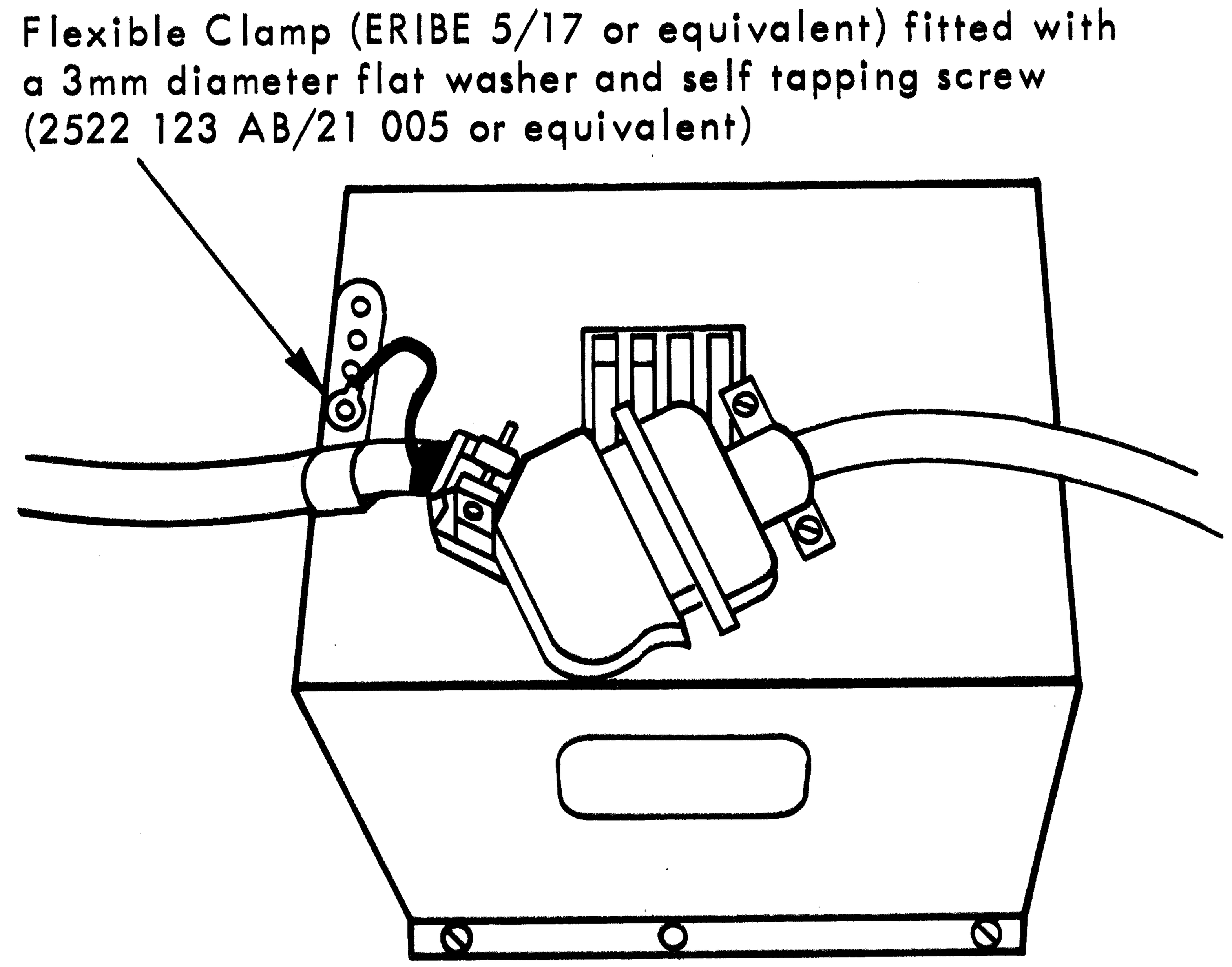
Connecting to the Serial Control Unit PER3100

V24 Interface: At the device a tapped hole is near the connector which should be used to attach the lead

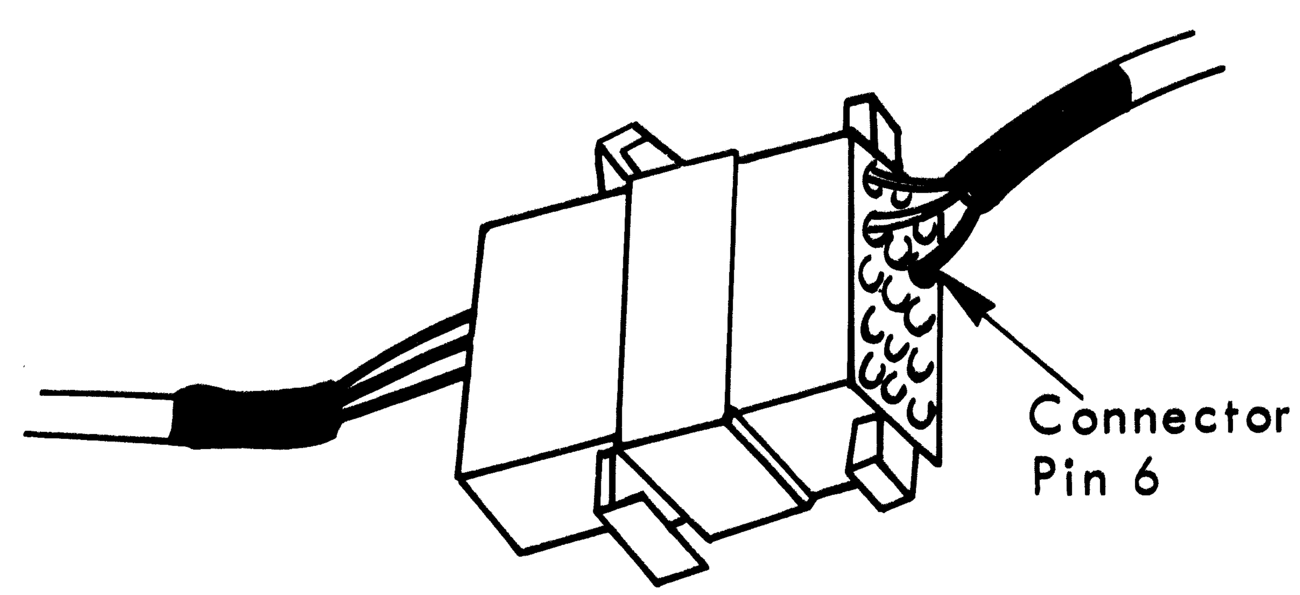


Connecting to the Serial Control Unit ASR33

V24 Interface: At the device the cable is attached to the metal cover of the transformer situated in the base of the ASR.

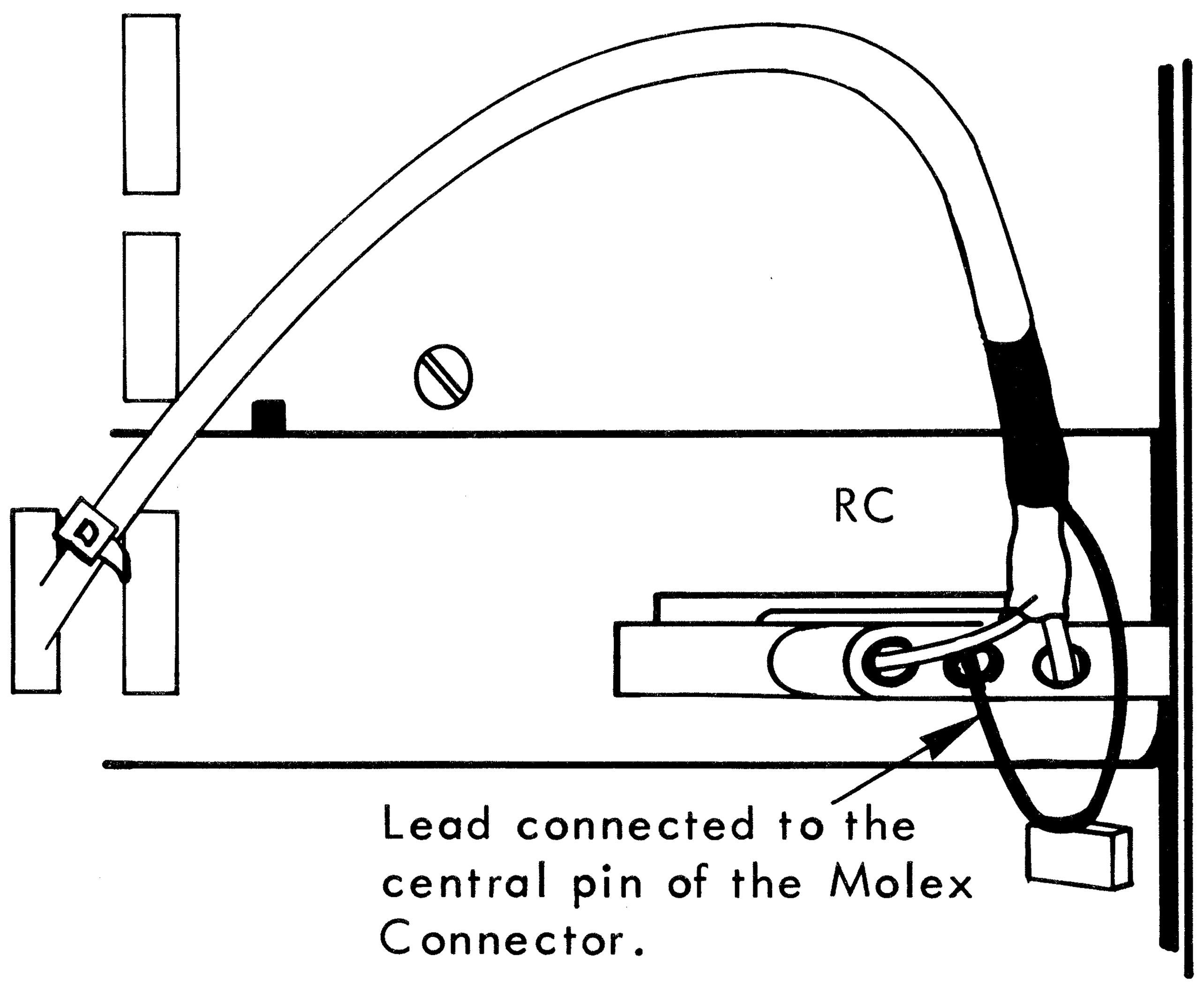


Current Loop Interface: At the device when using the extension cable the connection between Molex connector pin 6 and the ASR mechanical ground is made with the cable shielding. Note that under no circumstances must the Molex connectors be situated outside the ASR base.

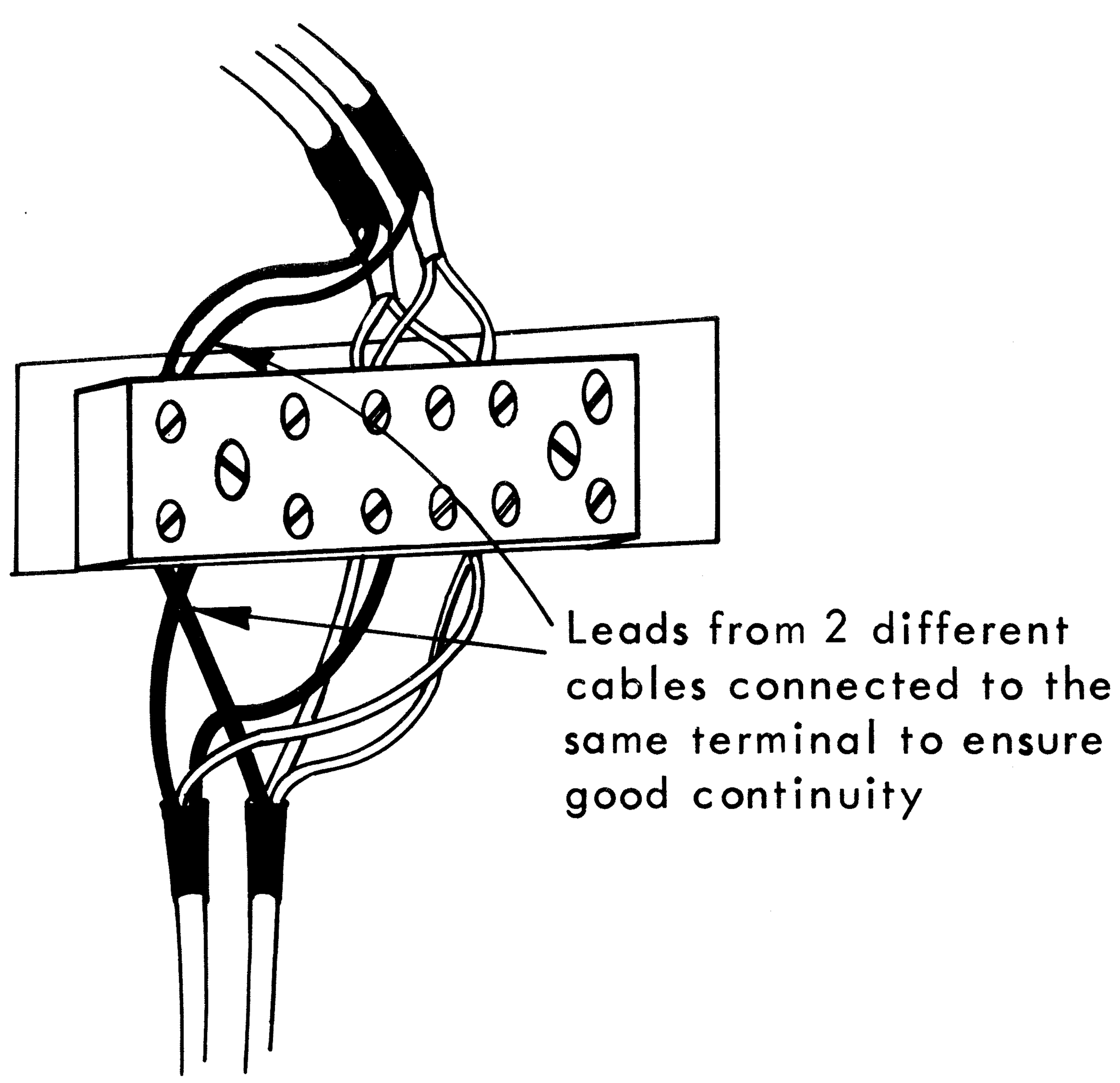


Connecting the Remote Control Cables

Rack Side: The shielding is connected to the central pin of the Molex connector. Note that for racks M1, M2, M4, M5, E2 and K7 a link connects the mechanical ground to the central pin of the Molex connector.

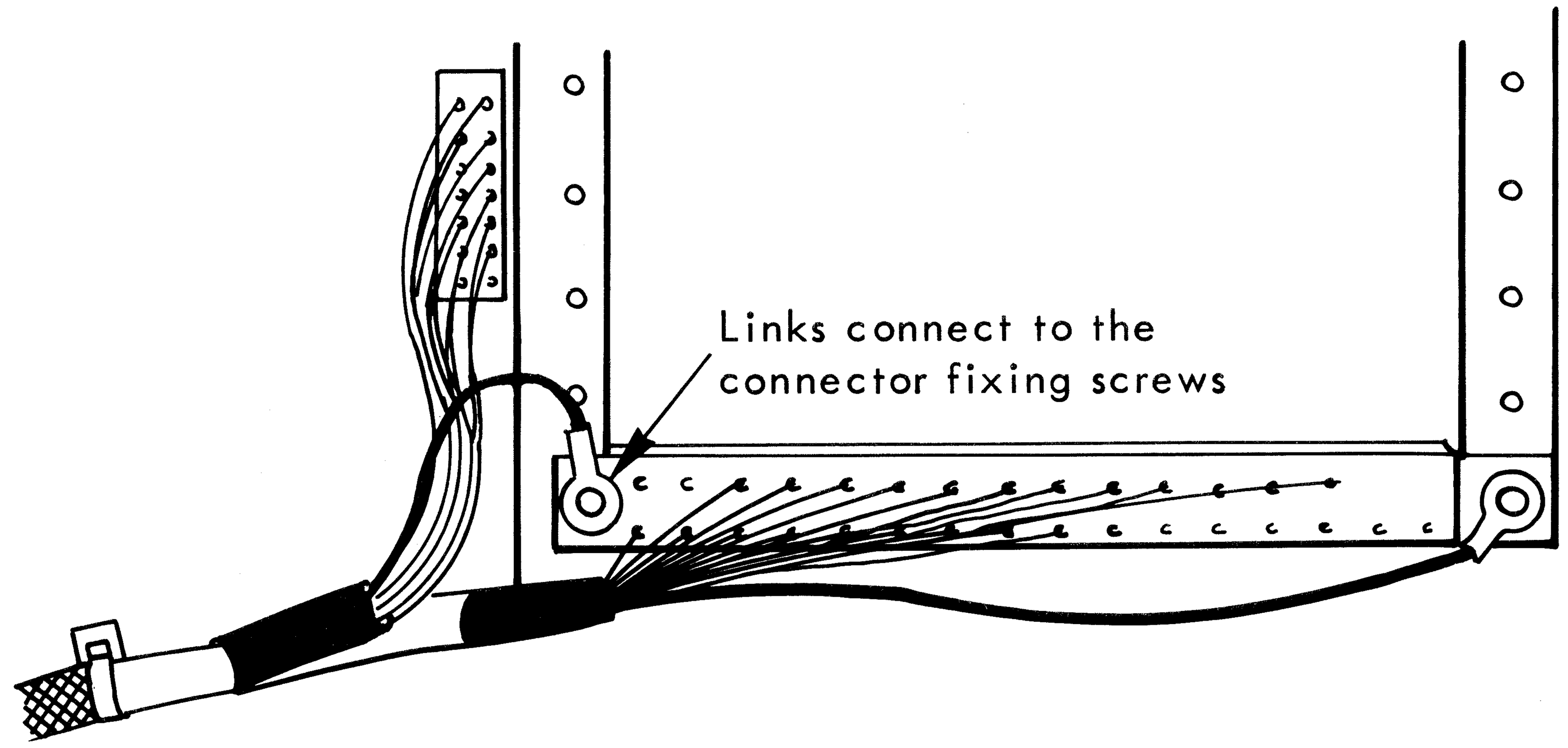


Connection at the Cabinet Terminal Box: When connecting the shielding of two cables at the Terminal Box connect to the same terminal to ensure good continuity.



Connecting the Break Cables

At the Extension Rack:



At the Cassette Rack:

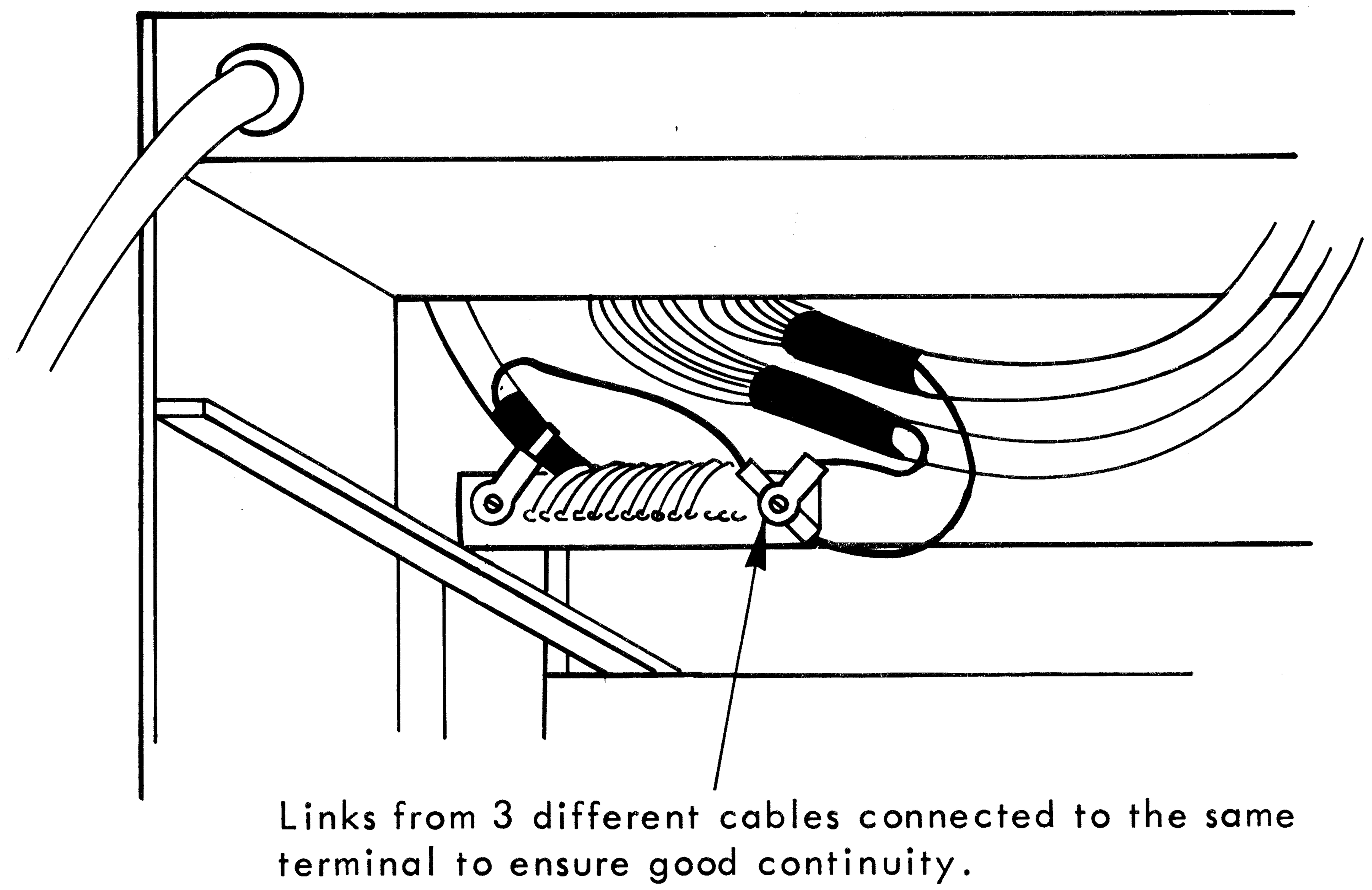


Table 4-2 CPU-A, Connector 1 (V24 CU)

1A01	0V	1B01	ASR LINE
1A02	PIFN	1B02	RTC AN
1A03	CPFN	1B03	SCEIN
1A04	IS02N	1B04	IS05N
1A05	PFFN	1B05	IS04N
1A06	BIEC4	1B06	RTCFZ1N
1A07	BIEC2	1B07	BIEC1
1A08	BIEC5	1B08	BIEC3
1A09	IS06N	1B09	BIEC0
1A10	IS03N	1B10	IS07N
1A11	IS01N	1B11	IS00N
1A12	INTASRN (INTSERN)	1B12	
1A13		1B13	
1A14		1B14	
1A15		1B15	
1A16		1B16	
1A17		1B17	
1A18		1B18	
1A19		1B19	
1A20		1B20	
1A21		1B21	
1A22		1B22	
1A23		1B23	
1A24		1B24	
1A25		1B25	
1A26		1B26	
1A27	0V	1B27	0V
1A28	5V	1B28	5V
1A29		1B29	
1A30	Mech. Ground	1B30	
1A31	CT103 2	1B31	0V
1A32	CT104 3	1B32	0V
1A33	CT106 5	1B33	0V
1A34	CT107 6	1B34	0V
1A35	CT1082 20	1B35	0V
1A36	CT109 8	1B36	0V
1A37	CT133 25	1B37	0V

Table 4-3 Connector 3 (CPU, Mem, IOP, CU)

3A01	+18V		c	3B01	-18V		c
3A02	BIEC0		a c	3B02	Chassus Ground		c
3A03	BIEC2		a c	3B03	BIEC1		a c
3A04	BIEC4		a c	3B04	BIEC3		a c
3A05	SCEIN		a c	3B05	BIEC5		a c
3A06	+16V		b	3B06	+16V		b
3A07	0V			3B07	0V		
3A08	BIO 00N			3B08	BIO 01N		
3A09	BIO 02N			3B09	BIO 03N		
3A10	BIO 04N			3B10	BIO 05N		
3A11	BIO 06N			3B11	BIO 07N		
3A12	BIO 08N			3B12	BIO 09N		
3A13	BIO 10N			3B13	BIO 11N		
3A14	BIO 12N			3B14	BIO 13N		
3A15	BIO 13N			3B15	BIO 15N		
3A16	OKO	*	a c d	3B16	OKI	*	c d
3A17	PWFN		a c	3B17	-5V		a b c
3A18	0V			3B18	-5V		b
3A19	+5V			3B19	+5V		
3A20	+5V			3B20	+5V		
3A21	0V			3B21	0V		
3A22	0V			3B22	0V		
3A23	BR (CU - 4)		c	3B23	+5V Battery		
3A24	0V			3B24			
3A25	0V			3B25	+16VM		b
3A26	WRITE	*		3B26	MAD 15		
3A27	CHA	*		3B27	MAD 14		
3A28	TRMN	*		3B28	MAD 13		
3A29	TMRN		a b c	3B29	MAD 12		
3A30	TMEN		a c d	3B30	MAD 11		
3A31	TMPN		a c d	3B31	MAD 10		
3A32	TPMN		a c d	3B32	MAD 09		
3A33	0V			3B33	MAD 08		
3A34	ACN		a c	3B34	MAD 07	*	
3A35	SPYC	*	a c d	3B35	MAD 06	*	
3A36	BUSRN	*	a c d	3B36	MAD 05	*	
3A37	MSN	*	a c d	3B37	MAD 04		
3A38	BSYN	*	a c d	3B38	MAD 03		
3A39	CLEARN		a c d	3B39	MAD 02	*	
3A40	0V			3B40	MAD 01	*	
3A41	BR (CU-2)		c	3B41	MAD 00	*	
3A42	BR (CU-3)		c	3B42	MAD 64	*	
3A43	BR (CU-1)		c	3B43	MAD 128		

a- CPU only
b- Memory only
c- Control Unit only
d- IOP only
* CU use only on DMA (main chassis)

Table 4-4 CPU-A Connector-5

5A01	
5A02	
5A03	
5A04	
5A05	
5A06	
5A07	
5A08	
5A09	
5A10	* SP03
5A11	* FLOACTN
5A12	* BSYCPUAN
5A13	* GFETCH
5A14	* DONEFN
5A15	* FLOCR1
5A16	
5A17	* OSCF0
5A18	
5A19	* MMUABS
5A20	* DONEMN
5A21	* BOMFN
5A22	* FU
5A23	* S01
5A24	* S03
5A25	* SP02
5A26	
5A27	* 0V
5A28	* 5V
5A29	BIOEKEY
5A30	UNLOCKN
5A31	RUNN
5A32	RCP00N
5A33	LOADRN
5A34	READSTN
5A35	RCP03N
5A36	LOADMN
5A37	READRN

5B01	* SP05
5B02	* GBCPFN
5B03	* PREQN
5B04	* CPBABS
5B05	* TESTN
5B06	
5B07	
5B08	
5B09	
5B10	* SP04
5B11	* SP01
5B12	* TMFN
5B13	* BOFFN
5B14	* PLOCR0
5B15	* FPPABS
5B16	
5B17	* MOSCFL0
5B18	
5B19	
5B20	* MFAULTN
5B21	
5B22	* S00
5B23	* S02
5B24	* TMMM
5B25	* TMMU
5B26	
5B27	* 0V
5B28	* 5V
5B29	CPMCN
5B30	IPL
5B31	START
5B32	CPINT
5B33	RUNFA
5B34	RCP01N
5B35	RCP02N
5B36	READMN
5B37	INSTN

Table 4-5 Control Panel Connector

A01	BIO15N
A02	BIO14N
A03	BIO13N
A04	BIO12N
A05	BIO11N
A06	BIO10N
A07	BIO09N
A08	BIO08N
A09	BIO07N
A10	BIO06N
A11	BIO05N
A12	BIO04N
A13	BIO03N
A14	BIO02N
A15	BIO01N
A16	BIO00N
A17	
A18	READRN
A19	0V

B01	CPMN
B02	BIOEKEY
B03	IPL
B04	UMLOCKN
B05	START
B06	RUNN
B07	CPINT
B08	RCP0N
B09	RUNFA
B10	LOADRN
B11	RCP1N
B12	READSTN
B13	RCP2N
B14	RCP3N
B15	READMN
B16	INSTN
B17	LOADMN
B18	LOCK
B19	+5V

Table 4-6 IOP Connectors 4, 5 (Break)

4A01	IR07N
4A02	IR05N
4A03	IR03N
4A04	IR01N
4A05	
4A06	BREX07N
4A07	BREX06N
4A08	BREX05N
4A09	BREX04N
4A10	BREX03N
4A11	BREX02N
4A12	BREX01N
4A13	BREX00N

4B01	IR06N
4B02	IR04N
4B03	IR02N
4B04	IR00N
4B05	
4B06	BR07N
4B07	BR06N
4B08	BR05N
4B09	BR04N
4B10	BR03N
4B11	BR02N
4B12	BR01N
4B13	BR00N

5A01	
5A02	
5A03	
5A04	
5A05	
5A06	BREX07N
5A07	BREX06N
5A08	BREX05N
5A09	BREX04N
5A10	BREX03N
5A11	BREX02N
5A12	BREX01N
5A13	BREX00N

5B01	
5B02	
5B03	
5B04	
5B05	
5B06	Ground (0V)
5B07	Ground (0V)
5B08	Ground (0V)
5B09	Ground (0V)
5B10	Ground (0V)
5B11	Ground (0V)
5B12	Ground (0V)
5B13	Ground (0V)

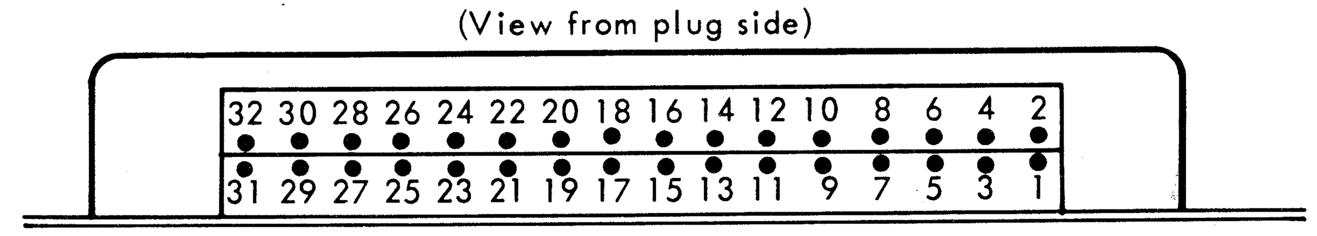
Table 4-7 Extention Connectors AIE/TAIE

TAIE Connectors

1A01	+ 5V	1B01	0V	2A01	BOF02N	2B01	IR04N
1A02		1B02	BIN00N	2A02	EOR	2B02	IR02N
1A03	BIN00N	1B03	BIN01N	2A03	BOF01N	2B03	IR06N
1A04	BIN01N	1B04	BIN02N	2A04	BOF00N	2B04	IR07N
1A05	BIN02N	1B05	BIN03N	2A05	BOU14	2B05	IR00N
1A06	BIN03N	1B06	BIN04N	2A06	BOU13	2B06	IR01N
1A07	BIN04N	1B07	BIN05N	2A07	BOU12	2B07	IR03N
1A08	BIN05N	1B08	BIN06N	2A08	BOU09	2B08	IR05N
1A09	BIN06N	1B09	BIN07N	2A09	BOU10	2B09	
1A10	BIN07N	1B10		2A10	BOU11	2B10	
1A11	BR00	1B11	ACCN	2A11	BOU15	2B11	
1A12	BR01	1B12	ACCN	2A12	MCN	2B12	
1A13	ACCN	1B13	AREN	2A13	BAD03N	2B13	
1A14	AREN	1B14	AREN	2A14	DAVN	2B14	
1A15		1B15	SP4	2A15	BAD04N	2B15	
1A16		1B16	SP3	2A16	BAD05N	2B16	
1A17	BR02	1B17	SP2	2A17	BOU08	2B17	
1A18	BR03	1B18	SP1	2A18	BOU07	2B18	
1A19	BR06	1B19		2A19	BOU06	2B19	
1A20	BR07	1B20		2A20	BOU05	2B20	
1A21	BIN08N	1B21	BIN08N	2A21	BOU04	2B21	
1A22	BIN14N	1B22	BIN14N	2A22	BAD00N	2B22	
1A23	BR04	1B23		2A23	BOU03	2B23	
1A24	BR05	1B24	S	2A24	BAD01N	2B24	
1A25		1B25		2A25	BAD02N	2B25	
1A26	BIN13N	1B26	BIN13N	2A26	BOU02	2B26	
1A27	BIN15N	1B27	BIN15N	2A27	BOU01	2B27	
1A28	BIN09N	1B28	BIN09N	2A28	BOU00	2B28	
1A29	BIN10N	1B29	BIN10N	2A29	+ 5V TELEC	2B29	PWFMOD1
1A30	BIN11N	1B30	BIN11N	2A30		2B30	PWFMOD2
1A31	BIN12N	1B31	BIN12N	2A31	+ 5V	2B31	0V

Break Connector for AIE/TAIE Cards

(View from plug side)



1	BR00N (I)	17	BR04N (I)
2	BR00N (O)	18	BR04N (O)
3	M	19	M
4	M	20	M
5	BR01N (I)	21	BR05N (I)
6	BR01N (O)	22	BR05N (O)
7	M	23	M
8	M	24	M
9	BR02N (I)	25	BR06N (I)
10	BR02N (O)	26	BR06N (O)
11	M	27	M
12	M	28	M
13	BR03N (I)	29	BR07N (I)
14	BR03N (O)	30	BR07N (O)
15	M	31	M
16	M	32	M

Reference Berg 65268 - 005

BR U-LINK

(shown in internal position)

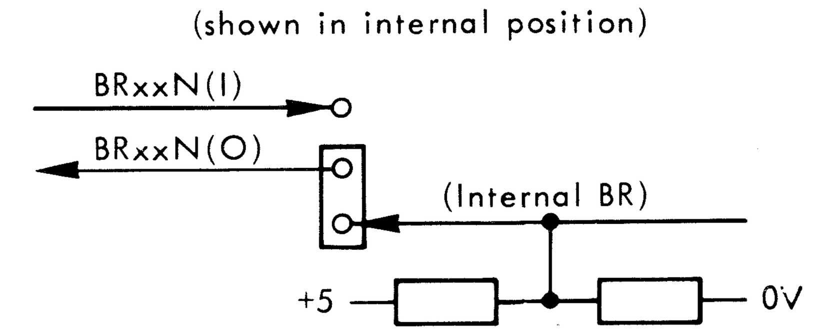


Table 4-8 Interface Signal List

GP BUS (Section II)		
Input to CPU	Input/Output	Output from CPU
BUSRN	ACN	CHA
MSN	BIO00-15N	WRITE
PWFN	BSYN	CLEARN
RSLN	TMEN	MAD128,64,00-15
BIEC0-5	TMPN	OK0/OK1
	TMRN	SCEIN
	TPMN	SPYC
	TRMN	

CONTROL PANEL (Section III)		
Input to CPU		Output from CPU
* CPBABS	RCP0-3N	BIOEKEY
CPINT	READMN	* BSYCPUAN
CPMCN	READRN	* GBCPFN
INSTN	READSTN	RUNFA
IPL	RUNN	
LOADMN	START	* = Extended (Address) half of panel, P857 only
LOADRN	TESTS	
* PREQN	UNLOCKN	

V24 SERIAL CONTROL UNIT (Section VI)		
Input from Device	Output from Device	Ground Signals
CT103	CT104	CT101
CT1082	CT106	CT102
CT133	CT107	
	CT109	

CPU/V24 SERIAL CU (CPU Logic)			
Input to CPU	Output from CPU		
AREDELA (TT)	BUSFDET (TT)	K04,08,10-15 (AA)	µQ1 (BB)
ASR0-7 (JJ)	CPGFZON (PP)	K04,08-15 (AA)	RSLAN (RR)
BRGFN (PP)	D10,15 (HH)	L08-15 (HH)	RSLCN (RR)
TYAC (TT)	FNU (DD)	MCL, MCLN (RR)	T5N,T7 (EE)
			TC810 (EE)

V24 CU to CPU Interrupt
INTSERN

MEMORY MANAGEMENT UNIT (MMU)		
Input to CPU	Output from CPU	
DONEMN	BOMFN	OSCFLO
MFAULTN	BSYCPUAN	S00-03
MMUABS	FU	TMMN
	GFETCH	TMMU

FLOATING POINT PROCESSOR (FPP)		
Input to CPU	Output from CPU	
DONEFN	BOFFN	GFETCH
FPPABS	BSYCPUAN	OSCFLO
FLOCR0, 1	FLOACT	TMFN

INTERNAL INTERRUPTS (LEVELS 0-7)		
Input to CPU	Output from CPU	Input to CPU
(P.Supply) PWFN	PFFN	ISO0N
(C.Panel) CPINT	CPINTFN	ISO1N
(CPU program)	PIFN	ISO2N
(P.Supply) RTCFZ1N	RTCFAN	ISO3N
		ISO4N
(V24 CU) INSERN		ISO5N
		ISO6N
		ISO7N

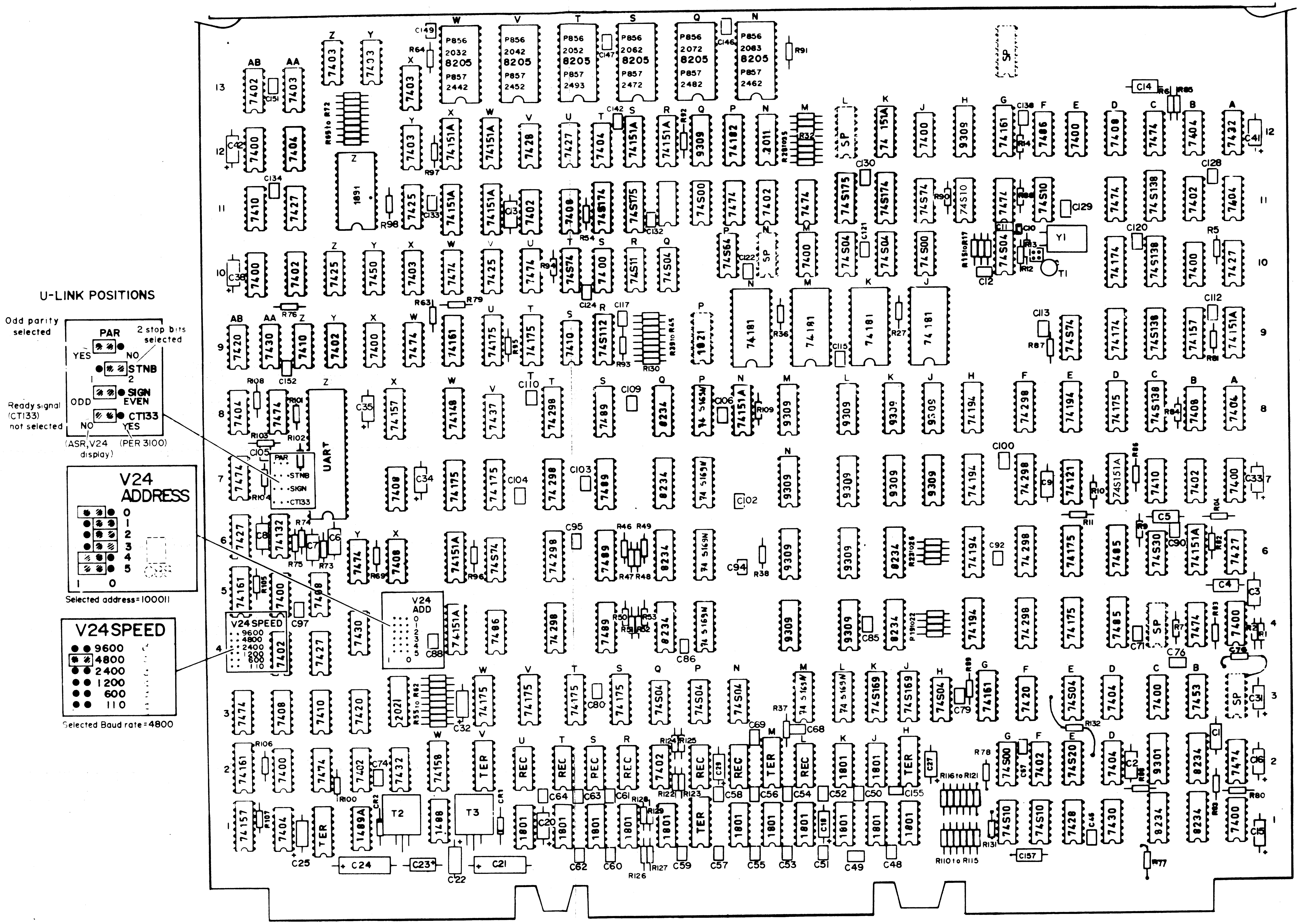


Figure 4-5 CPU Card Layout

Table 4-9A P856M CPU Parts List

Reference	Description	12NC Code
	Printed Circuit	5111 100 05701
	Heat Sink	
	Right Spring	
	Left Spring	
	Integrated Circuit 7400	
	Integrated Circuit 7402	
	Integrated Circuit 7403	
	Integrated Circuit 7404	
	Integrated Circuit 7408	
	Integrated Circuit 7410	
	Integrated Circuit 7420	
	Integrated Circuit 7425	
	Integrated Circuit 7427	
	Integrated Circuit 7428	
	Integrated Circuit 7430	
	Integrated Circuit 7432	
	Integrated Circuit 7437	
	Integrated Circuit 7450	
	Integrated Circuit 7453	
	Integrated Circuit 7474	
	Integrated Circuit 7485	
	Integrated Circuit 7486	
	Integrated Circuit 7489	
	Integrated Circuit 74121	
	Integrated Circuit 74132	
	Integrated Circuit 74148	
	Integrated Circuit 74151A	
	Integrated Circuit 74157	
	Integrated Circuit 74158	
	Integrated Circuit 74161	
	Integrated Circuit 74174	
	Integrated Circuit 74175	
	Integrated Circuit 74181	
	Integrated Circuit 74182	
	Integrated Circuit 74194	
	Integrated Circuit 74298	
	Integrated Circuit 74S00	
	Integrated Circuit 74S10	
	Integrated Circuit 74S11	
	Integrated Circuit 74S20	
	Integrated Circuit 74S30	
	Integrated Circuit 74S04	
	Integrated Circuit 74S64	

Table 4-9A Contd.

Reference	Description	12NC Code
	Integrated Circuit 74S74	
	Integrated Circuit 74S112	
	Integrated Circuit 74S138	
	Integrated Circuit 74S169	
	Integrated Circuit 74S174	
	Integrated Circuit 74S175	
T3.	Integrated Circuit 7812 (TO220)	
T2.	Integrated Circuit 7912 (TO220)	
	Integrated Circuit 1488	
	Integrated Circuit 1489A	
	Integrated Circuit 1801	
	Integrated Circuit 1891 (8576)	
	Integrated Circuit 8234	
	Integrated Circuit 9301	
	Integrated Circuit 9309	
	Integrated Circuit ADL 2011	
	Integrated Circuit REC 0612	
	Integrated Circuit 2502 (UART)	
	Integrated Circuit CR 2021	
	Integrated Circuit ROM 2032 (8205)	
	Integrated Circuit ROM 2042 (8205)	
	Integrated Circuit ROM 2052 (8205)	
	Integrated Circuit ROM 2062 (8205)	
	Integrated Circuit ROM 2072 (8205)	
	Integrated Circuit ROM 2083 (8205)	
	Integrated Circuit SN 74LS169N	
C10.	Capacitor 1nF, 100V, 10%, ceramic.	
C68.	Capacitor 200pF, 500V, 1%.	
C8,76.	Capacitor 390pF, 250V, 1%.	
C157.	Capacitor 510pF, 250V, 1%.	
C1,5,14.	Capacitor 620pF, 250V, 1%.	
C9.	Capacitor 1000pF, 125V, 1%.	
C7,6.	Capacitor 1.3nF, 63V, 1%.	
C2,4.	Capacitor 430pF, 250V, 1%.	
C13.	Capacitor 2 nF, 63V, 1%	
C158.	Capacitor 150pF, 63V, 2%.	
C21,24.	Capacitor 10μF, 63V, FITCO.	
C15,18,20,22,23,25,27,29,31-35,38, 41,42.	Capacitor 10μF, 25V, FITCO.	
C3.	Capacitor 1,8nF, 125V, 1%.	

Table 4-9A Contd.

Reference	Description	12NC Code
C12,46,49-64,67,69,71,74,79,80,85, 86,88,90,92,94,97,100, 102, 104-106, 109,110,112,113,115,117,120-122,124, 128-130,133,134,138,142,146,147,149, 151,152,155.	Capacitor 10nF, ceramic.	
R3,8,37,54,74,75.	Resistor 100Ω, 1/8W, 1%.	
R27,36,65-72.	Resistor 470Ω, 1/4W, 5%.	
R53.	Resistor 560Ω, 1/4W, 5%.	
R14,19-26,28-35,39-52,55-62,64,76, 80-109,130.	Resistor 1KΩ, 1/4W, 5%.	
R04-06,07,38,63,77,79,131,132.	Resistor 10KΩ, 1/4W, 5%.	
R16.	Resistor 464Ω, 1/8W, 1%.	
R17.	Resistor 681Ω, 1/8W, 1%.	
R12.	Resistor 2.15KΩ, 1/8W, 1%.	
R11,13.	Resistor 3.16KΩ, 1/8W, 1%.	
R110-115,122,123,126,127.	Resistor 220Ω, 1/4W, 5%.	
R116-121,124,125,128,129.	Resistor 390Ω, 1/4W, 5%.	
R15.	Resistor 1.47KΩ, 1/8W, ± 1%.	
R1,2,10,73.	Resistor 110Ω, 0.125W, 1%.	
R9,78.	Resistor 147Ω, 0.125W, 1%.	
CR1,2.	IC. TERNET Resistors.	
Y 1.	Quartz QA 55A22.22 Mhz.	
T 1.	Transistor BSX20	
	U-Link DCW06.	
	Mica 56325.	

Table 4-9B P857M CPU Parts List

Reference	Description	12 NC Code
	Printed Circuit	5111 100 05701
	Heat Sink	
	Right Spring	
	Left Spring	
	Integrated Circuit 7400	
	Integrated Circuit 7402	
	Integrated Circuit 7403	
	Integrated Circuit 7404	
	Integrated Circuit 7408	
	Integrated Circuit 7410	
	Integrated Circuit 7420	
	Integrated Circuit 7425	
	Integrated Circuit 7427	
	Integrated Circuit 7428	
	Integrated Circuit 7430	
	Integrated Circuit 7432	
	Integrated Circuit 7437	
	Integrated Circuit 7450	
	Integrated Circuit 7453	
	Integrated Circuit 7474	
	Integrated Circuit 7485	
	Integrated Circuit 7486	
	Integrated Circuit 7489	
	Integrated Circuit 74121	
	Integrated Circuit 74132	
	Integrated Circuit 74148	
	Integrated Circuit 74151A	
	Integrated Circuit 74157	
	Integrated Circuit 74158	
	Integrated Circuit 74161	
	Integrated Circuit 74174	
	Integrated Circuit 74175	
	Integrated Circuit 74181	
	Integrated Circuit 74182	
	Integrated Circuit 74194	
	Integrated Circuit 74298	
	Integrated Circuit 74S00	
	Integrated Circuit 74S10	
	Integrated Circuit 74S11	
	Integrated Circuit 74S20	
	Integrated Circuit 74S30	
	Integrated Circuit 74S04	
	Integrated Circuit 74S64	
	Integrated Circuit 74S74	

Table 4-9B Contd.

Reference	Description	12NC Code
	Integrated Circuit 74S112	
	Integrated Circuit 74S138	
	Integrated Circuit 74S169	
	Integrated Circuit 74S174	
	Integrated Circuit 74S175	
T3.	Integrated Circuit 7812 (TO220)	
T2.	Integrated Circuit 7912 (TO220)	
	Integrated Circuit 1488	
	Integrated Circuit 1489A	
	Integrated Circuit 1801	
	Integrated Circuit 1891 (8576)	
	Integrated Circuit 8234	
	Integrated Circuit 9301	
	Integrated Circuit 9309	
	Integrated Circuit 2502 (UART)	
	Integrated Circuit ADL 2011	
	Integrated Circuit REC 0612	
	Integrated Circuit CR 2021	
	Integrated Circuit ROM 2442 (8205)	
	Integrated Circuit ROM 2452 (8205)	
	Integrated Circuit ROM 2462 (8205)	
	Integrated Circuit ROM 2472 (8205)	
	Integrated Circuit ROM 2482 (8205)	
	Integrated Circuit ROM 2493 (8205)	
	Integrated Circuit SN74LS169N	
C10.	Capacitor 1nF, 100V, 10%, ceramic.	
C68.	Capacitor 200pF, 500V, 1%.	
C8,76.	Capacitor 390pF, 250V, 1%.	
C157.	Capacitor 510pF, 250V, 1%.	
C1,5,14.	Capacitor 620pF, 250V, 1%.	
C9.	Capacitor 1000pF, 125V, 1%.	
C6,7.	Capacitor 1.3nF, 63V, 1%.	
C2,4.	Capacitor 430pF, 250V, 1%.	
C13.	Capacitor 2nF, 63V, 1%.	
C12,46,49-64,67,69,71,74,79,80,85, 86,88,90,92,94,97,100,102,104-106, 109,110,112,113,115,117,120-122,124, 128-130,133,134,138,142,146,147,149, 151,152,155.	Capacitor 10nF, ceramic.	
C21,24.	Capacitor 10µF, 63V, FITCO.	
C15,18,20,22,23,25,27,29,31-35,38, 41,42.	Capacitor 10µF, 25V, FITCO.	
C158.	Capacitor 150pF, 63V, 2%. ceramic.	
C3.	Capacitor 1.8nF, 125V, 1%.	
C159	Capacitor 100pF 2%	

Table 4-9B Contd.

Reference	Description	12NC Code
R3,8,37,54,74,75.	Resistor 100Ω, 1/8W, 1%.	
R27,36,65-72.	Resistor 470Ω, 1/4W, 5%.	
R53.	Resistor 560Ω, 1/4W, 5%.	
R14,19-26,28-35,39-52,55-62,64,76, 80-109,130.	Resistor 1KΩ, 1/4W, 5%.	
R4,5,6,7,38,63,77,79,131,132.	Resistor 10KΩ, 1/4W, 5%.	
R16.	Resistor 464Ω, 1/8W, 1%.	
R17.	Resistor 681Ω, 1/8W, 1%.	
R12.	Resistor 2.15KΩ, 1/8W, 1%.	
R11,13.	Resistor 3.16KΩ, 1/8W, 1%.	
R110-115,122,123,126,127.	Resistor 220Ω, 1/4W, 5%.	
R116-121,124,125,128,129.	Resistor 390Ω, 1/4W, 5%.	
R15.	Resistor 1.47KΩ, 1/8W, 1%.	
R1,2,10,73.	Resistor 110Ω, 0,125W, 1%.	
R9,78.	Resistor 147Ω, 0,125W, 1%.	
CR1,2.	Diode AAZ 18.	
Y1.	Quartz QA 55A 22.22 Mhz.	
T1.	Transistor BSX 20.	
	IC TERNET Resistors.	
	U-Link DCW06.	
	Mica 56325.	
R133	Resistor 100Ω 1/4W 5%	

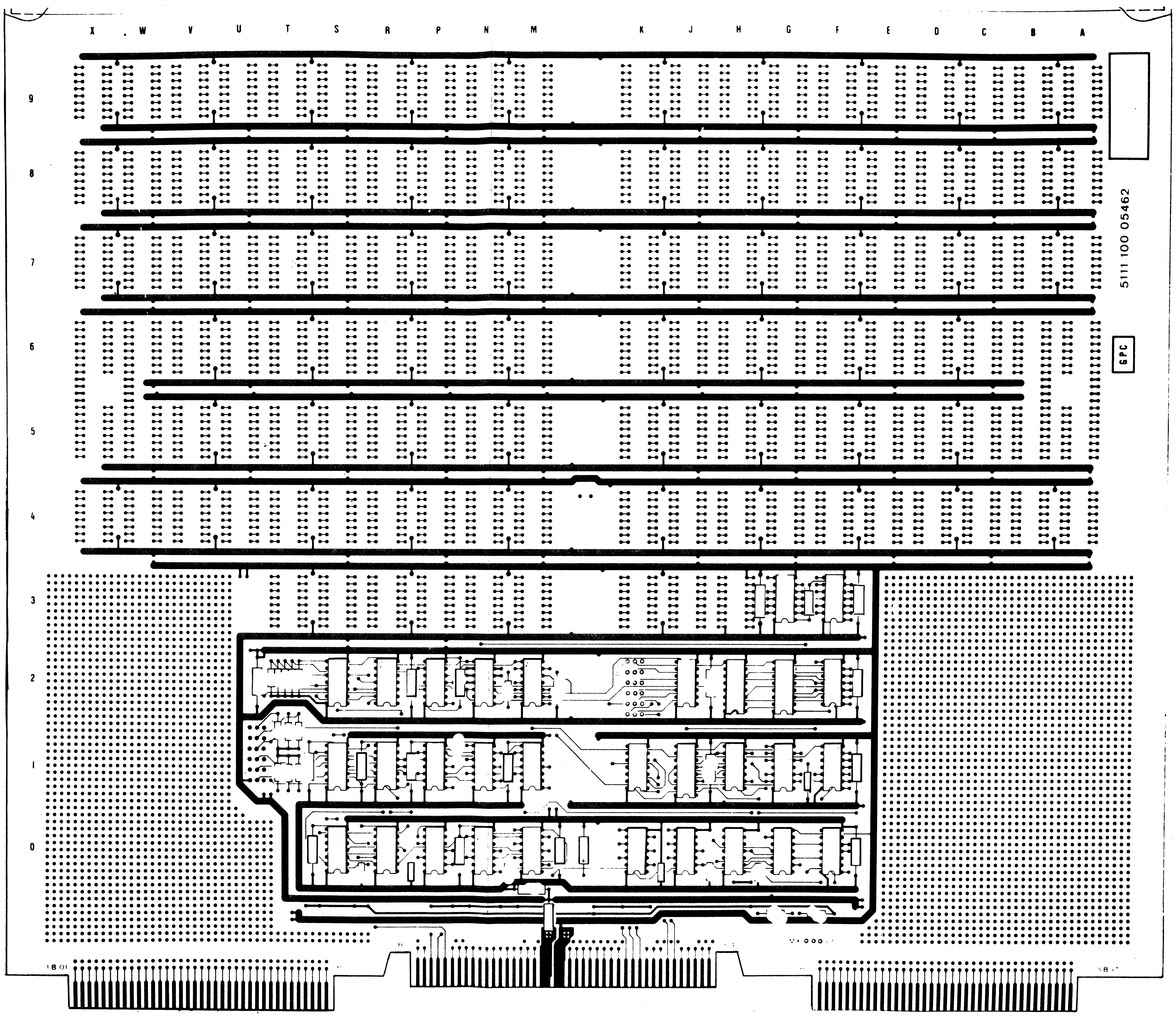
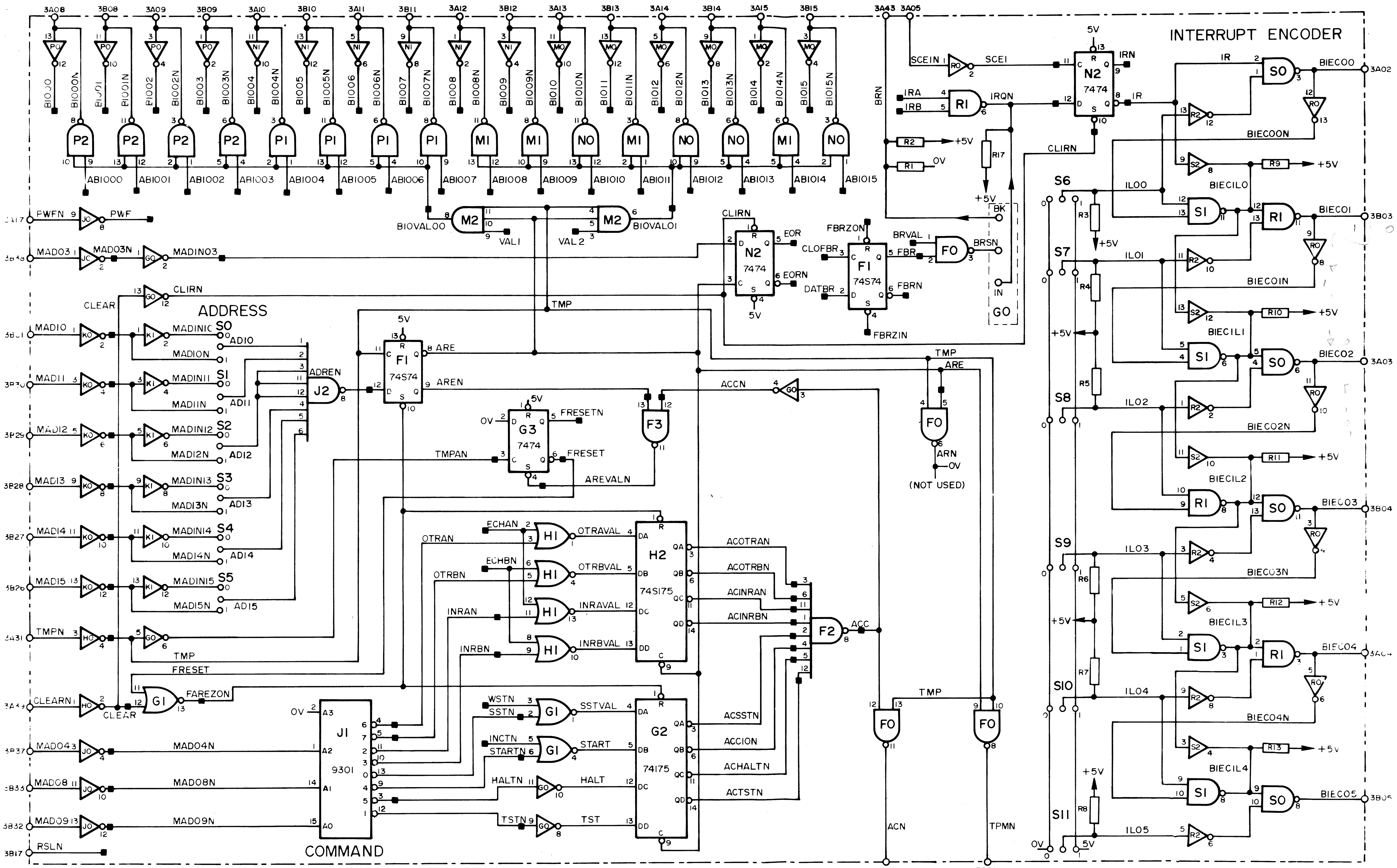


Figure 4-6 General Purpose Card Layout



NOTE:
CUSTOMER CONNECTIONS (—■—) ARE SHOWN HERE ADJACENT TO THEIR ACTUAL LOCATION ON THE CARD.

Figure 4-7 General Purpose Card Schematic

Table 4-10 General Purpose Card Parts List

Reference	Description	12NC Code
	Printed circuit	5111 100 05462
IC 1,4,10,12,13,15,23.	Integrated Circuit 1801	
IC 9,16,22.	Integrated Circuit 7404	
IC 11.	Integrated Circuit 7438	
IC 18,19.	Integrated Circuit 7402	
IC 20.	Integrated Circuit 74S74	
IC 21.	Integrated Circuit 7417	
IC 24,30.	Integrated Circuit 7474	
IC 25.	Integrated Circuit 74S11	
IC 26,29.	Integrated Circuit 7430	
IC 28.	Integrated Circuit 74175	
IC 31.	Integrated Circuit 7400	
IC 27.	Integrated Circuit 74S175	
IC 17.	Integrated Circuit 9301	
IC 2,3,5,6,7,8,14.	Integrated Circuit REC 0613	
C 7,8.	Capacitor 68µF, 16V, CTS.	
C 2,5,12-15,17-26.	Capacitor 3.3µF, 16V, CTS.	
C 3,4,6,9,10,11,16.	Capacitor 10µF, ± 20%, ceramic.	
R 1.	Resistor 220Ω, 0.250W, ± 5%.	
R 2,9,10,11,12,13.	Resistor 390Ω, 0.250W, ± 5%.	
R 3-8,14-17.	Resistor 1KΩ, 0.250W, ± 5%.	
L 1,2.	Inductance.	
S 0-12.	U-Link.	

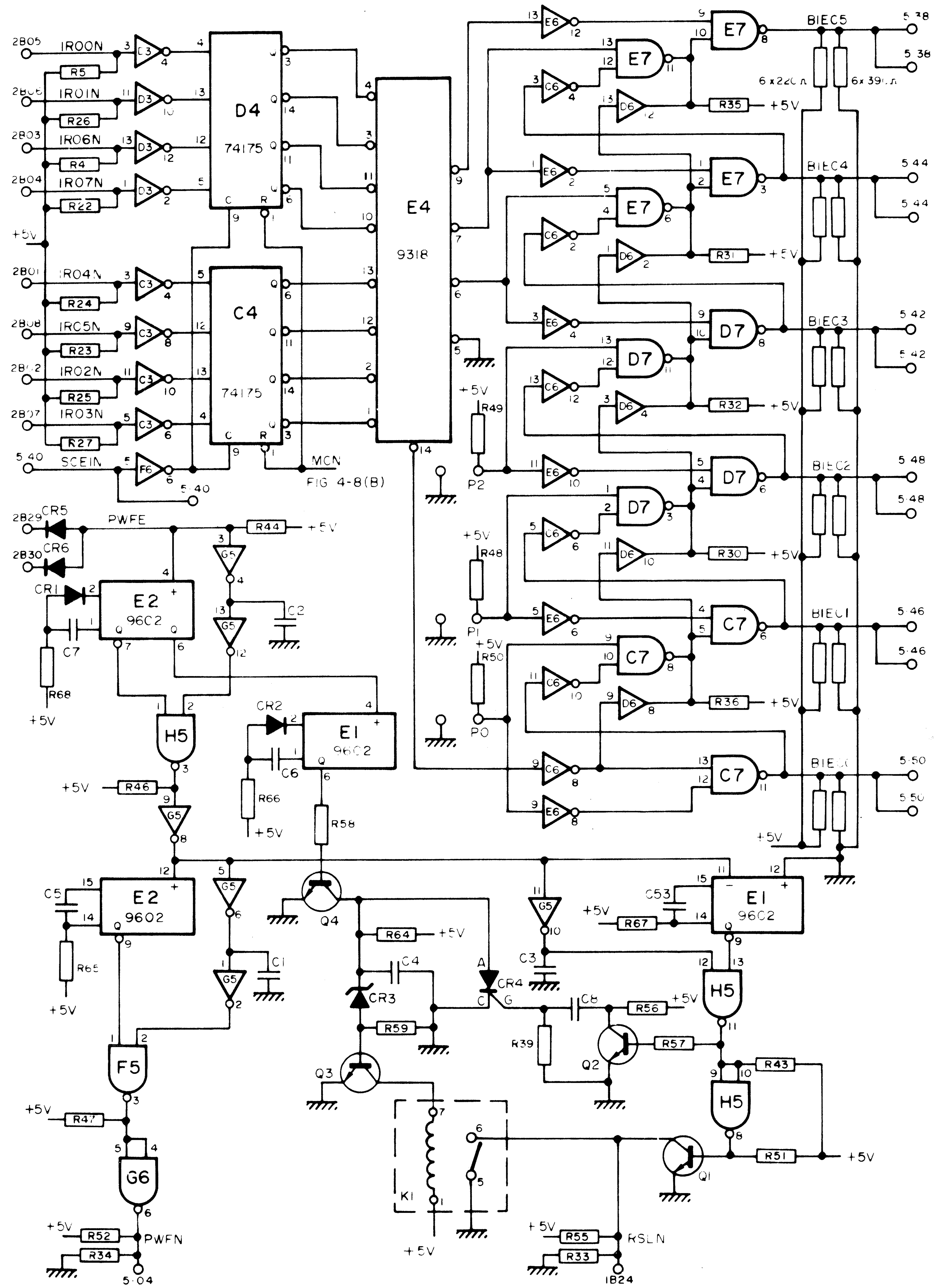


Figure 4-8(A) TAIE Card Schematic

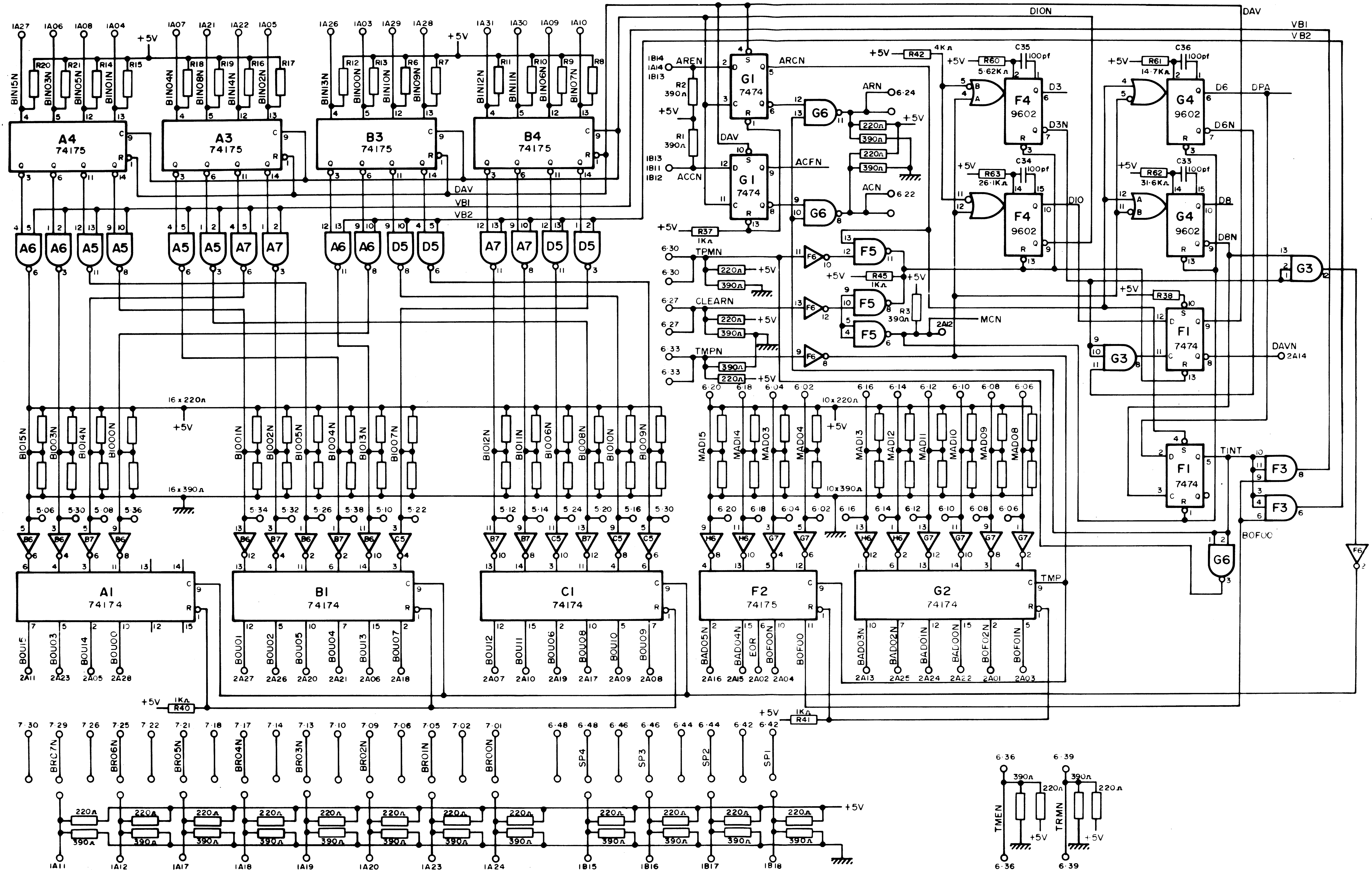


Figure 4-8(B) TAIE Card Schematic

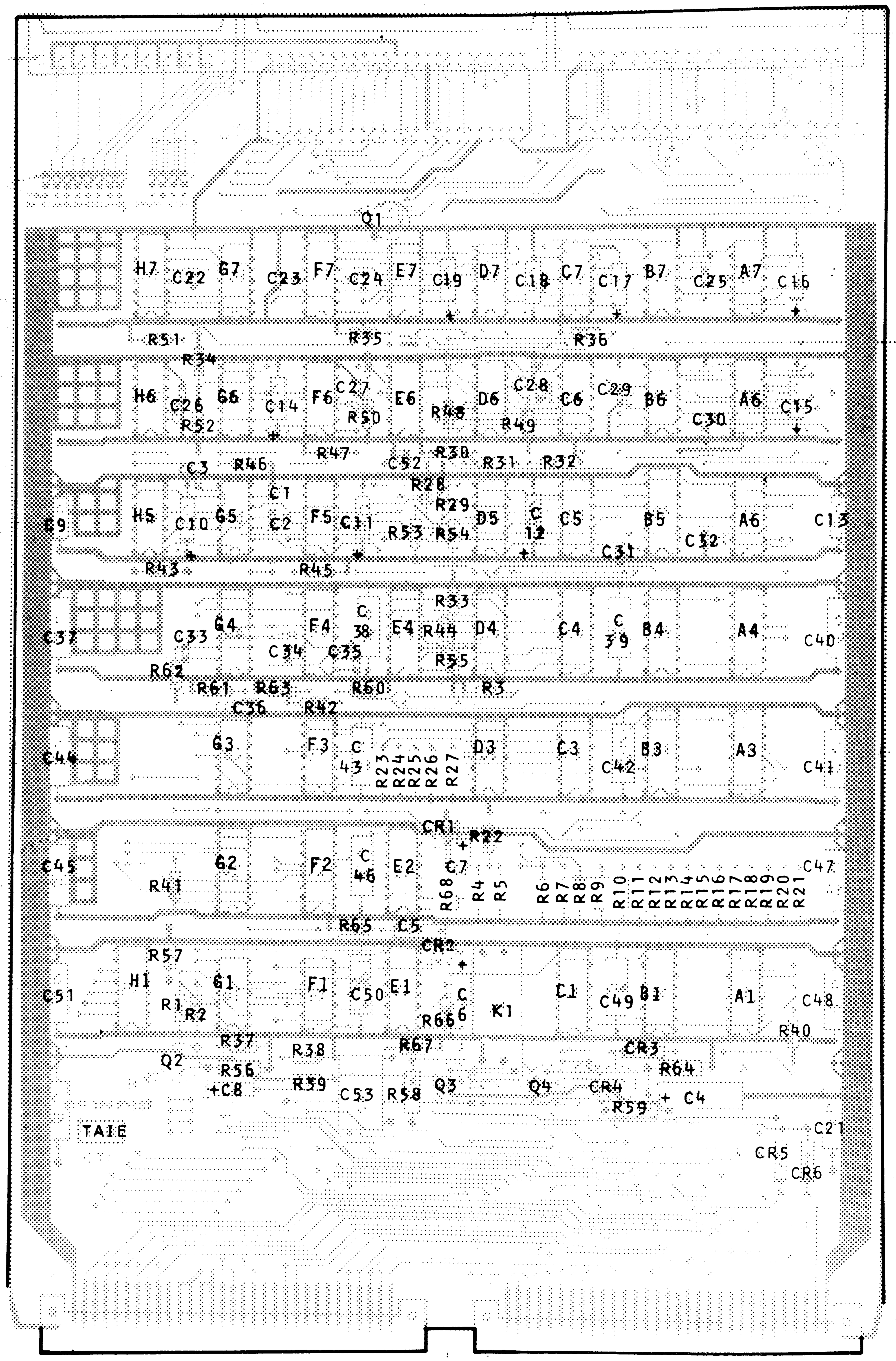


Figure 4-9 TAIE Card Layout

Table 4-11 TAIE Parts List

Reference	Description	12NC-Code
	TAIE CARD	5111 199 81440
	U-link	
A5,A6,A7,C7,D5,D7,E7,F5,G6,H5	IC 1801	
A1,B1,C1,G2	IC 74174	
G1,F1	IC 7474	
A3,A4,B3,B4,C4,D4,F2	IC 74175	
F3,G3	IC 74H11	
E1,E2,F4,G4	IC 9602	
E4	IC 9318	
B5,F7,H1,H7	IC Ternet Resistors	
C3,D3,E6,G5	IC 7404	
D6	IC 7417	
B6,B7,C5,C6,F6,G7,H6	IC REC 0612	
R37 thru R51	Resistor, 1KΩ, ±5%, 0,25W	
R1 thru R36	Resistor, 390Ω, ±5%, 0,25W	
R60	Resistor, 5,62KΩ, ±1%, 0,125W	
R61	Resistor, 14,7KΩ, ±1%, 0,125W	
R62	Resistor, 31,6KΩ, ±1%, 0,125W	
R63,R68	Resistor, 26,1KΩ, ±1%, 0,125W	
R57	Resistor, 1,6KΩ, ±5%, 0,25W	
R65	Resistor, 12KΩ, ±5%, 0,25W	
R66	Resistor, 24KΩ, ±5%, 0,25W	
R56,R58,R59	Resistor, 300Ω, ±5%, 0,5W	
R64	Resistor, 100Ω, ±5%, 0,5W	
R67	Resistor, 10KΩ, ±1%, 0,125W	
R52 thru R55	Resistor, 220Ω, ±5%, 0,25W	
Q1	Transistor BSX60	
Q2,Q3,Q4	Transistor 2N2219	
K1	Relay MRMD 15005	
CR1,CR2	Diode BAX 13	
CR3	Diode 1N746A	
CR5,CR6	Diode AAZ 18	
CR4	Thyristor 2N1595	
C53	Capacitor, 0,33µf, 10%, 100V, MPR	
C1,C2,C3	Capacitor, 470pf, ±10%, 100V, cer.plat	
C4	Capacitor, 47µf, 25V, FITCO	
C5	Capacitor, 0,1µf, 10%, 100V, MPR	
C6	Capacitor, 22µf, 16V, CTS13	
C7	Capacitor, 33µf, 10V, CTS13	
C8	Capacitor, 1µf, 35V, CTS13	
C9 thru C19	Capacitor, 3,3µf, 16V, CTS13	
C20, C21	Capacitor, 10µf, 15V, FITCO	
C22 thru C32,C52	Capacitor, 10000pf, 40V, cer.plat	
C33 thru C36	Capacitor, 100pf, 2%, 63V, cer.plat	
C37 thru C51	Capacitor, 0,01µf, -10%, 100V, MAC	

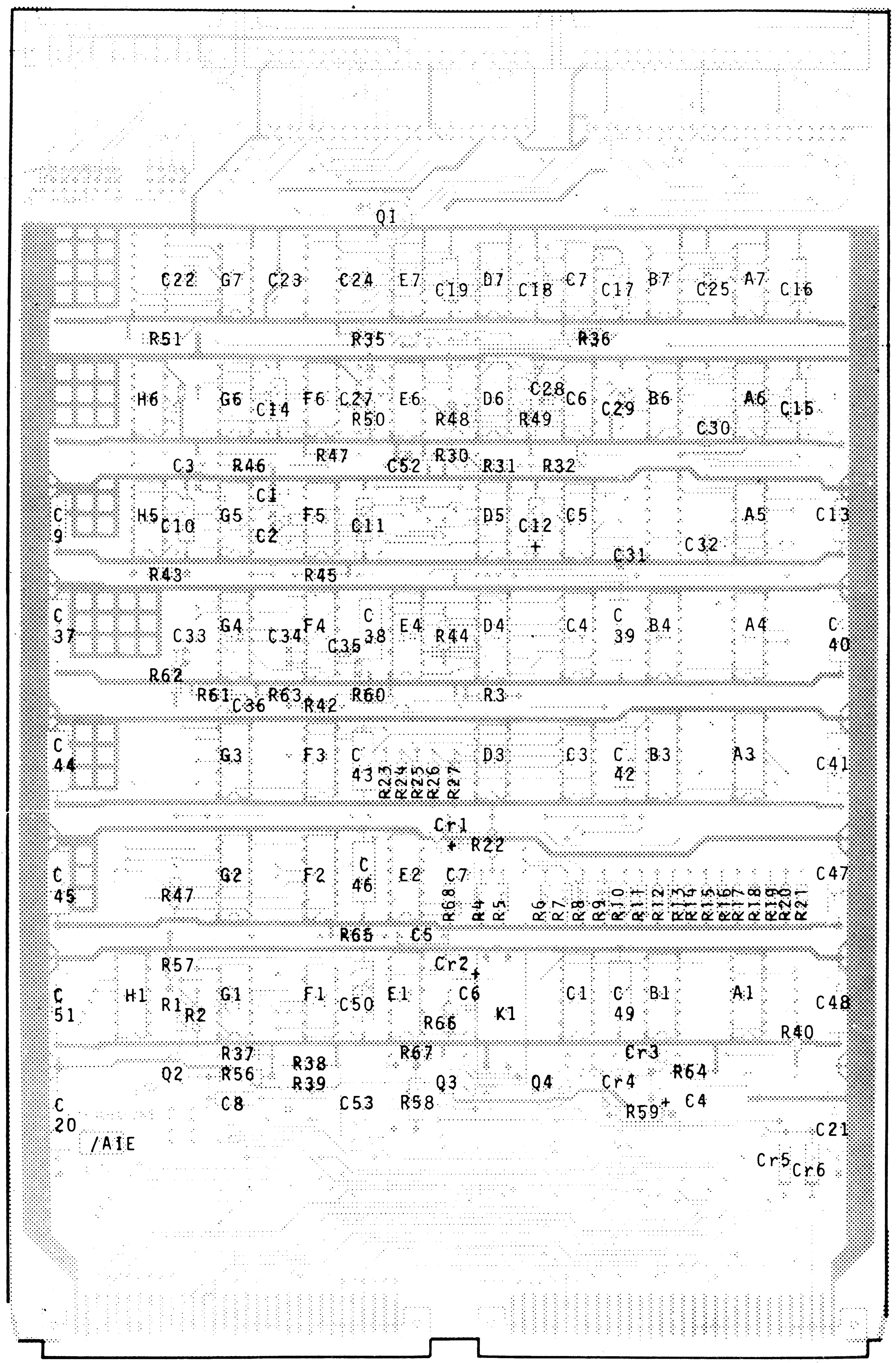


Figure 4-10 AIE Card Layout

Table 4-12 AIE Parts List

Reference	Description	12NC-Code
	AIE CARD	5111 199 78450
A5,A6,A7,C7,D5,D7,E7,F5,G6,H5	IC 1801	
A1,B1,C1,G2	IC 74174	
G1,F1	IC 7474	
A3,A4,B3,B4,C4,D4,F2	IC 74175	
F3,G3	IC 74H11	
E1,E2,F4,G4	IC 9602	
E4	IC 9318	
H1	IC Ternet Resistors	
C3,D3,E6,G5	IC 7404	
D6	IC 7417	
B6,B7,C5,C6,F6,G7,H6	IC REC 0612	
R37 thru R51	Resistor, 1KΩ, ±5%, 0,25W	
R1 thru R27,R30,R31,R32,R35,R36	Resistor, 390Ω, ±5%, 0,25W	
R60	Resistor, 5,62KΩ, ±1%, 0,125W	
R61	Resistor, 14,7KΩ, ±1%, 0,125W	
R62	Resistor, 31,6KΩ, ±1%, 0,125W	
R63,R68	Resistor, 26,1KΩ, ±1%, 0,125W	
R57	Resistor, 1,6KΩ, ±5%, 0,25W	
R65	Resistor, 12KΩ, ±5%, 0,25W	
R66	Resistor, 24KΩ, ±5%, 0,25W	
R56,R58,R59	Resistor, 300Ω, ±5%, 0,5W	
R64	Resistor, 100Ω, ±5%, 0,5W	
R67	Resistor, 10KΩ, ±1%, 0,125W	
Q1	Trnasistor BSX 60	
Q2,Q3,Q4	Trnasistor 2N2219	
K1	Relay MRMD 15005	
CR1,CR2	Diode BAX 13	
CR3	Diode 1N746A	
CR5,CR6	Diode AAZ 18	
CR4	Thyristor 2N1595	
C53	Capacitor, 0,33μF, 10%, 100V, MPR	
C1,C2,C3	Capacitor, 470pf, ±10%, 100V, cer.plat	
C4	Capacitor, 47μF, 25V, FITCO	
C5	Capacitor, 0,1μf, 10%, 100V, MPR	
C6	Capacitor, 22μf, 16V, CTS13	
C7	Capacitor, 33μf, 10V, CTS13	
C8	Capacitor, 1μf, 35V, CTS13	
C9 thru C19	Capacitor, 3,3 μf, 16V, CTS13	
C20,C21	Capacitor, 10μf, 25V, FITCO	
C22 thru C32-C52	Capacitor, 10000pf, 40V, cer. plat	
C33 thru C36	Capacitor, 100pf, 2%, 63V, cer. plat	
C37 thru C51	Capacitor, 0,01μf, ±10%, 100V, MAC	

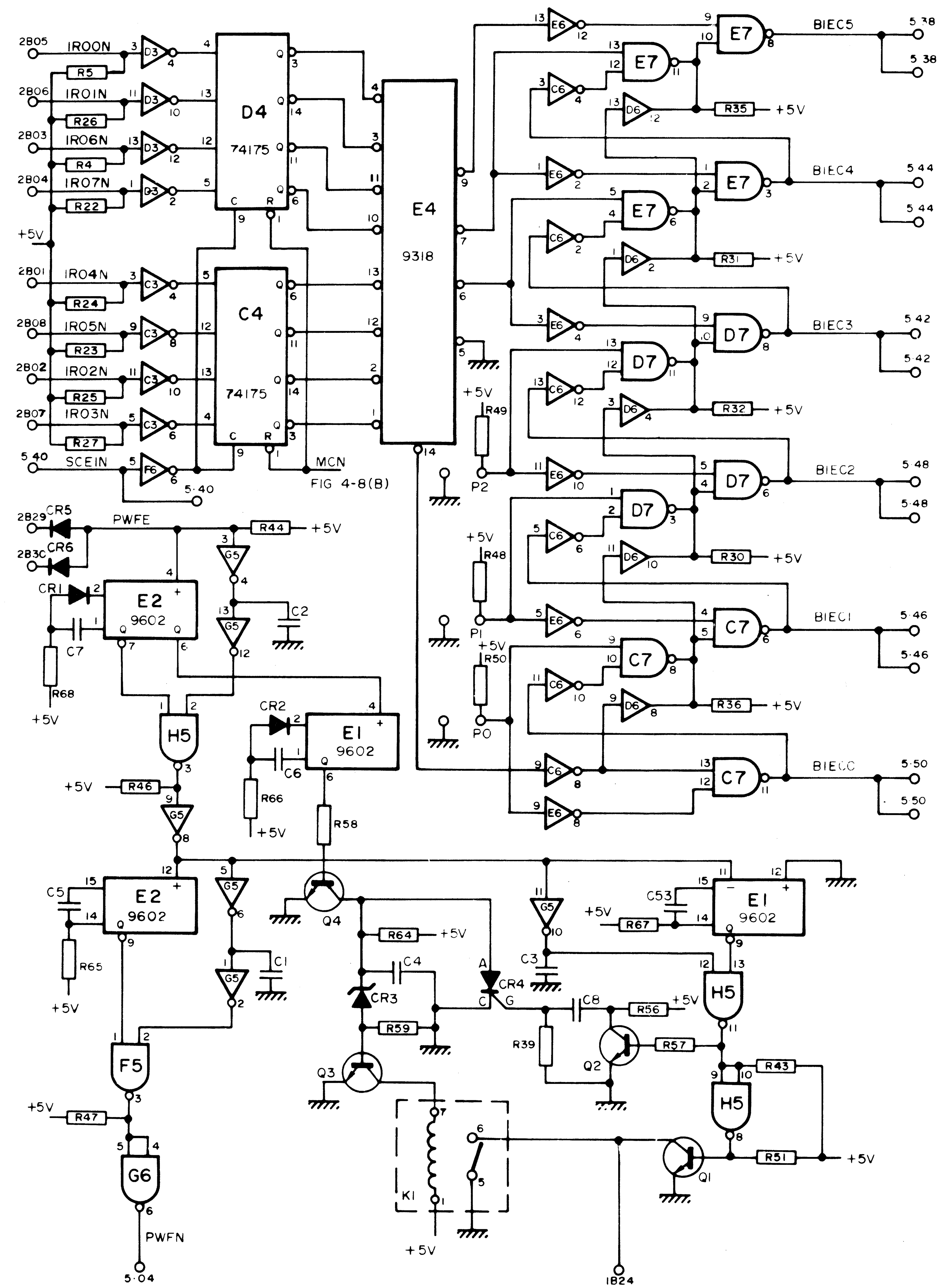


Figure 4-11(A) AIE Card Schematic

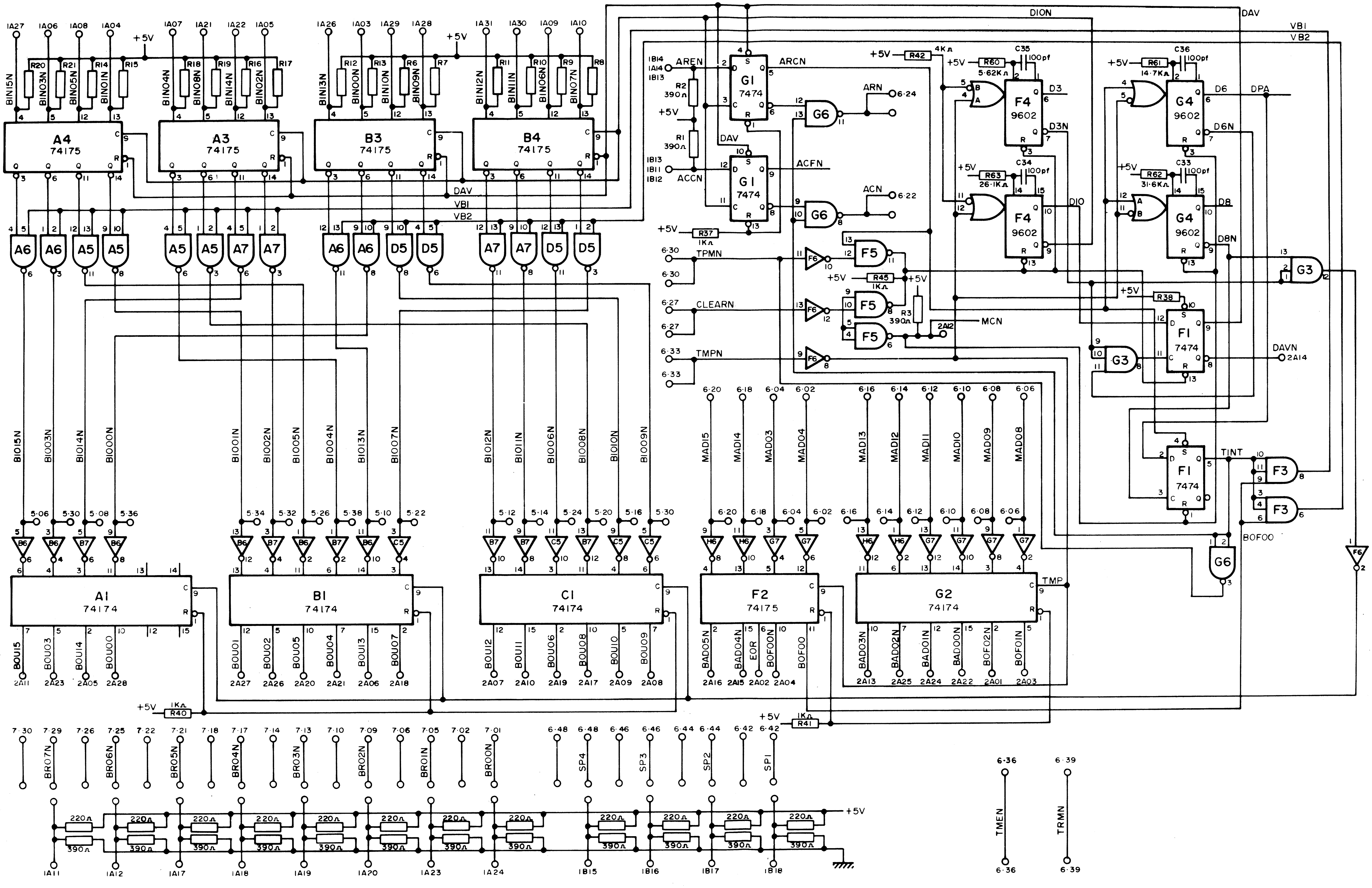
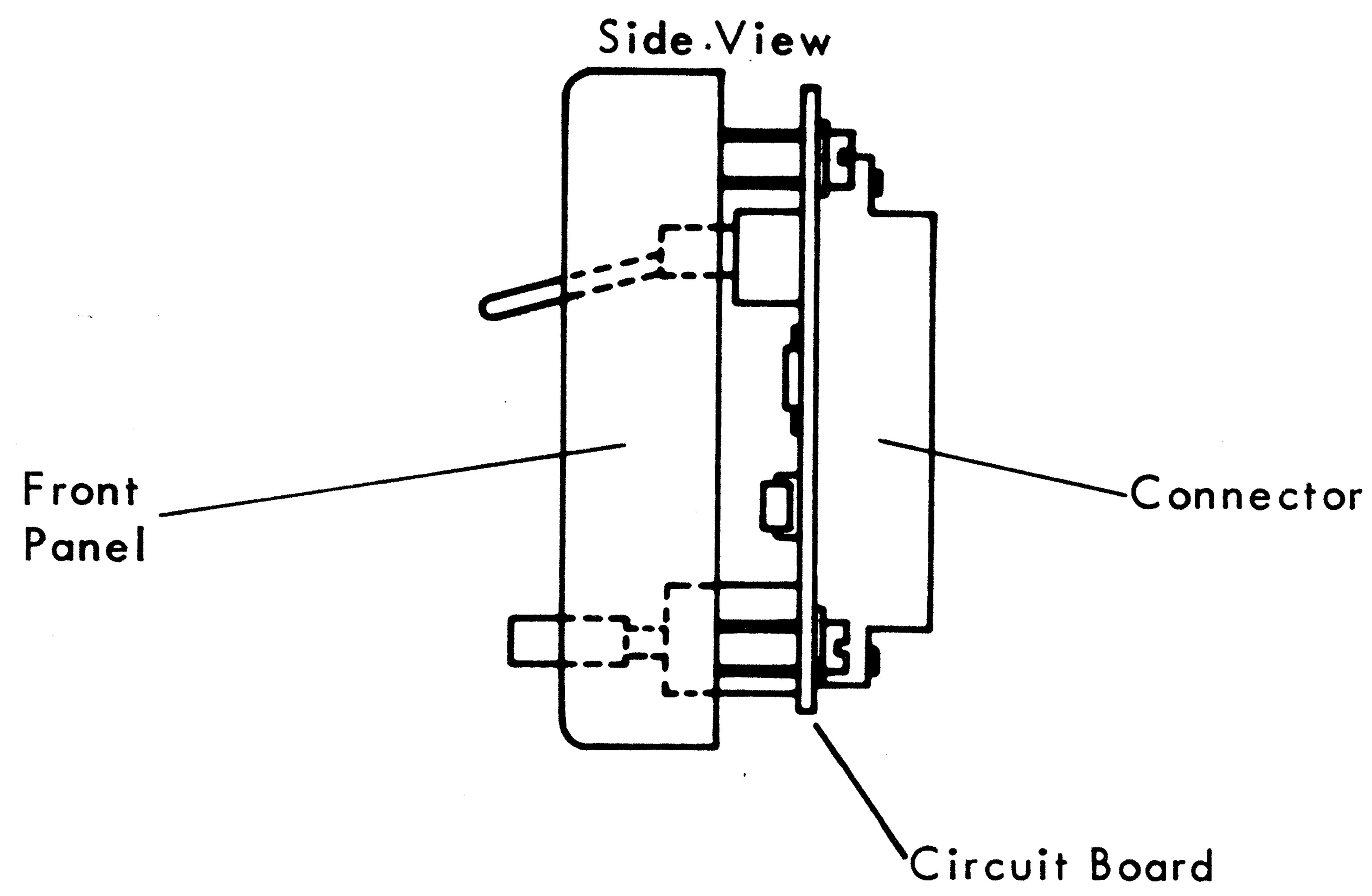
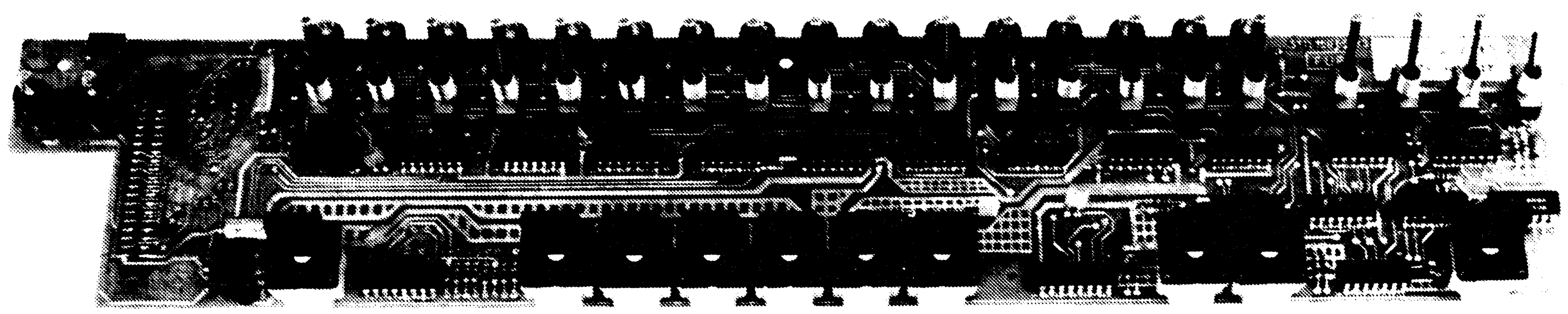


Figure 4-11(B) AIE Card Schematic





Control Panel Parts List

Reference	Description	12NC Code
	Control Panel	5111 199 81450
	Front Cover	79700
	Printed Circuit, equiped	82070
	Circuit Board	100 05213
IC18, IC19	Integrated Circuit 74729	
IC17	7410	
IC13	7404	
IC12	9602	
IC11, IC15	7432	
IC1, IC7, IC8	7416	
IC3, IC4, IC5, IC10	1801	
IC14	7420	
IC16	7400	
IC2, IC6, IC9	REC 0612	
C19, C20	Capacitor, 47µF, 20V, CTS13	
C1	Capacitor, 4700pF, 100V, ±10 cer plat	
C2 thru C18, C21	Capacitor, 3900pF, 100V, ±10 cer plat	
L1, L2, L3	Inductor	
R1	Resistor, 10KΩ, 0.250W, ±5	
R2	Resistor, 46.4KΩ, 0.125W, ±1	
R23 thru R38	Resistor, 330Ω 0.250W, ±5	
R3 thru R22, R39, R40	Resistor, 1KΩ, 0.250W, ±5	
	Toggle Switch 7101 LYCG	
	Push Button Switch 2RT - TFB	
	Lamp 6V 30mA ref. 2306	
	Jumper Block F088	

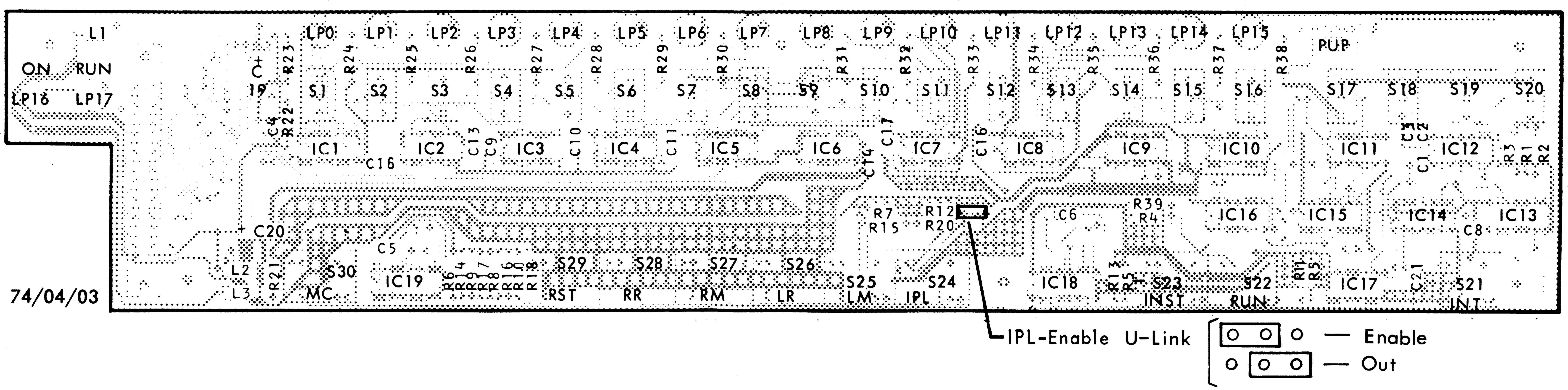


Figure 4-12A Standard Control Panel Layout

Extended Control Panel Parts List

Reference	Description	12NC Code
	Printed Circuit	5111 100 05764
U1,3,5,6,7,29.	Integrated circuit 1801	
U2,4,8,26.	Integrated circuit REC 0613	
U28.	Integrated circuit 74S11	
U27.	Integrated circuit 74S00	
U9,18,25.	Integrated circuit 7417	
U10,13,16,19,22.	Integrated circuit 74157	
U11,14,17,20,23.	Integrated circuit 74S169	
U12,15,21,24.	Integrated circuit 9324	
C1-17,19-39.	Capacitor 10nF, ceramic.	
C18.	Capacitor 560pF, 10%, ceramic.	
	Capacitor 22µF, 10V, FITCO.	
R18,19,20,22,23,24.	Resistor 1KΩ, 0.25W, 5%.	
R1-17.	Resistor 330Ω, 0.25W, 5%.	
R21.	Resistor 100Ω, 0.25W, 5%.	
LP1-17.	Lamp 6V 30 mA ref. 2306.	

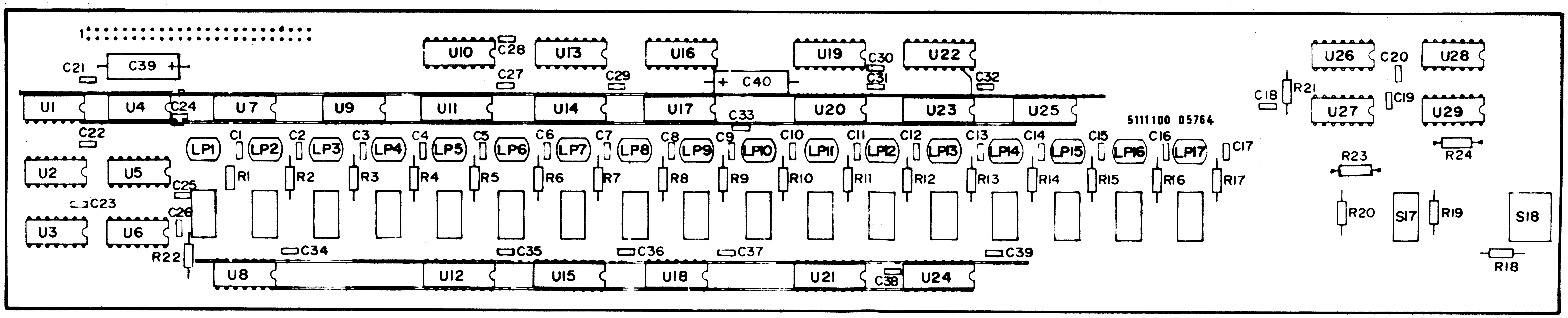


Figure 4-9B Extended Control Panel Layout

SECTION V

POWER SUPPLIES

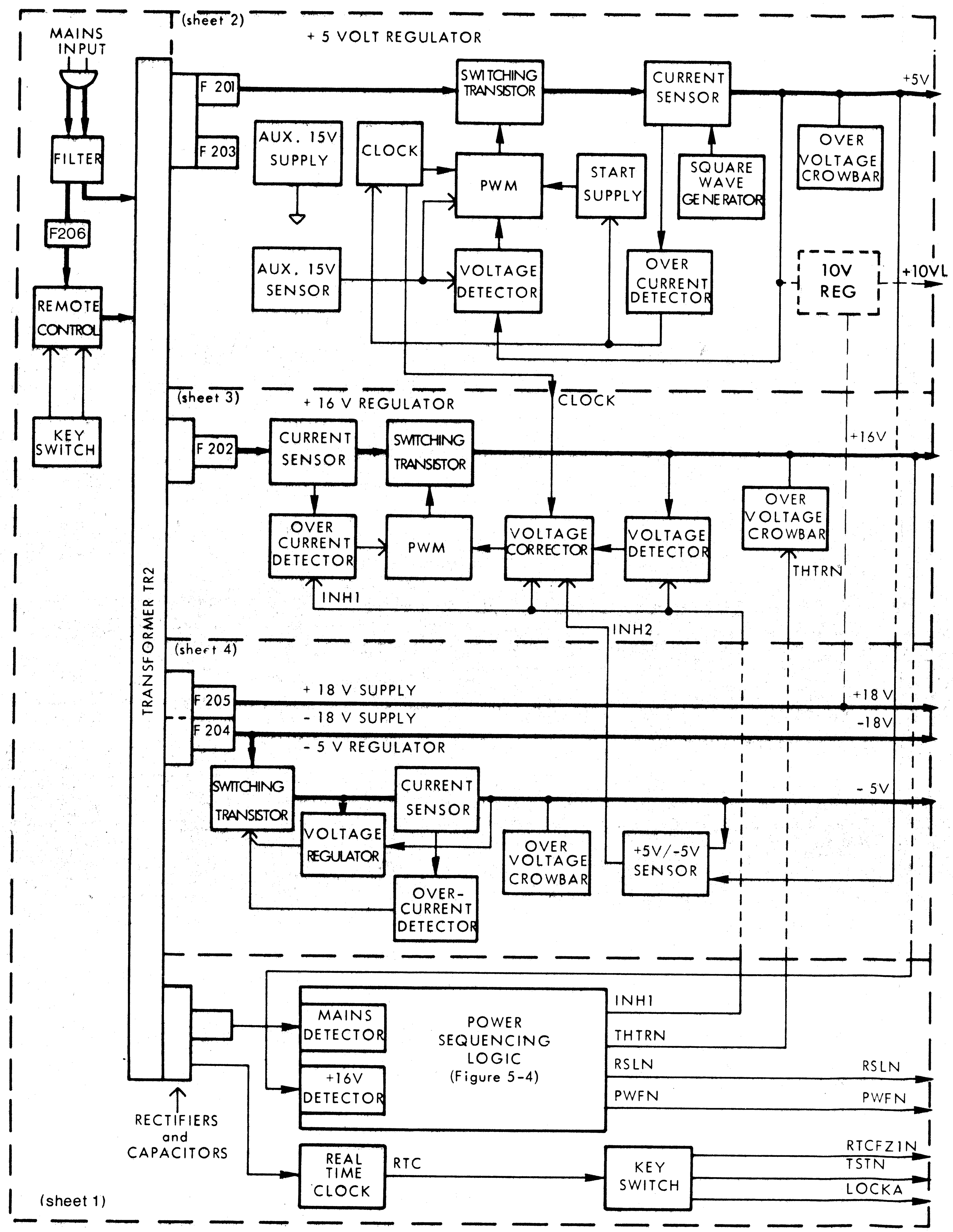
5.1 GENERAL

Each basic chassis contains its own power supply (Table 4-1). The power supply with the M4 chassis provides regulated +5V at 43 amps, +16V at 9 amps, and -5V at 2 amps, as well as unregulated +18 and -18 volts at 2 amps each. The M5 chassis contains two power supplies : one identical to the M4 supply, and a second supply that differs only in some component positions. The main power and logic signals of the basic supply are shown in the block diagram, Figure 5-1. The mechanical configuration and parts locations are shown in Figures 5-6 through 5-9.

5.2 INPUTS

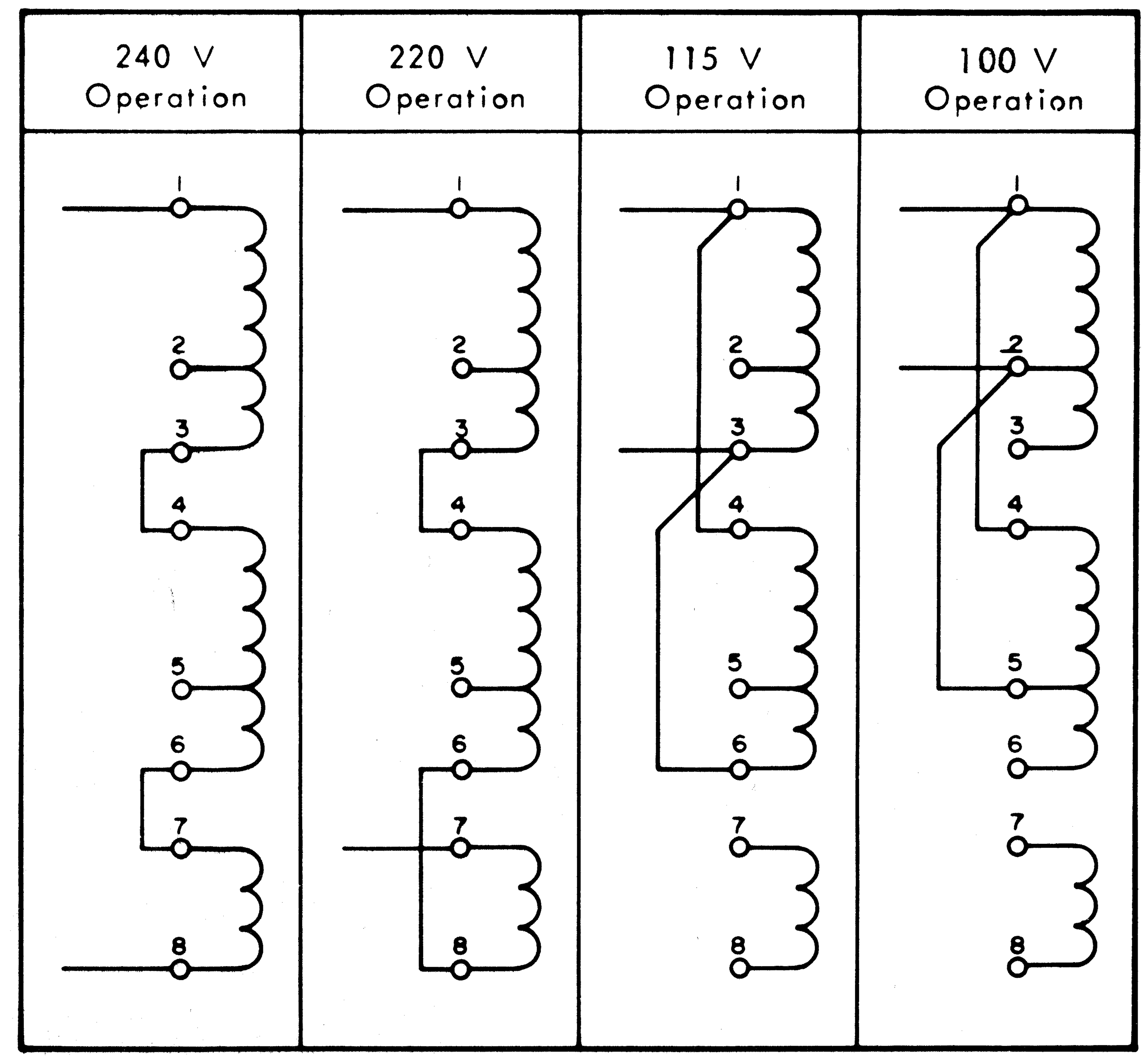
The power supply operates on voltages of 100V, 115V, 220V, or 240V, ±10 %, single phase, at either 50 Hz (± 2 Hz) or 60 Hz (± 3 Hz). The power supply is adapted to the selected input voltage by wiring the input-side of the mains transformer, T201, as shown in Figure 5-2. The transformer input connections and jumpers are made by individual plugs at the rear of the chassis (Figure 5-7). The AC input is filtered, fused, and then switched by means of a remote-control circuit. The remote-control circuit (Figure 5-3, sheet 1) is located on printed-circuit card P1.

5.3 A battery connection is provided on the power supply chassis. The purpose of an external battery supply is to maintain the +16V and part of the +5V supplies during a mains power failure. The battery connector wiring is shown on Figure 5-3, sheet 1.



Notes - The dashed-line sections show the locations on schematic diagram Figure 6-18
The arrows ➞ show normal flow.
The arrows → show the inhibit/enable signals.

Figure 5-1 P857 Power Supply Block Diagram



Point 9 of the transformer is a shield between primary and secondary windings. The shield must be connected to the ground.

The fan is connected on the first 115V winding (1-3) so that it is always supplied with 115V.

Figure 5-2 Mains Input Wiring

5.4 OUTPUTS

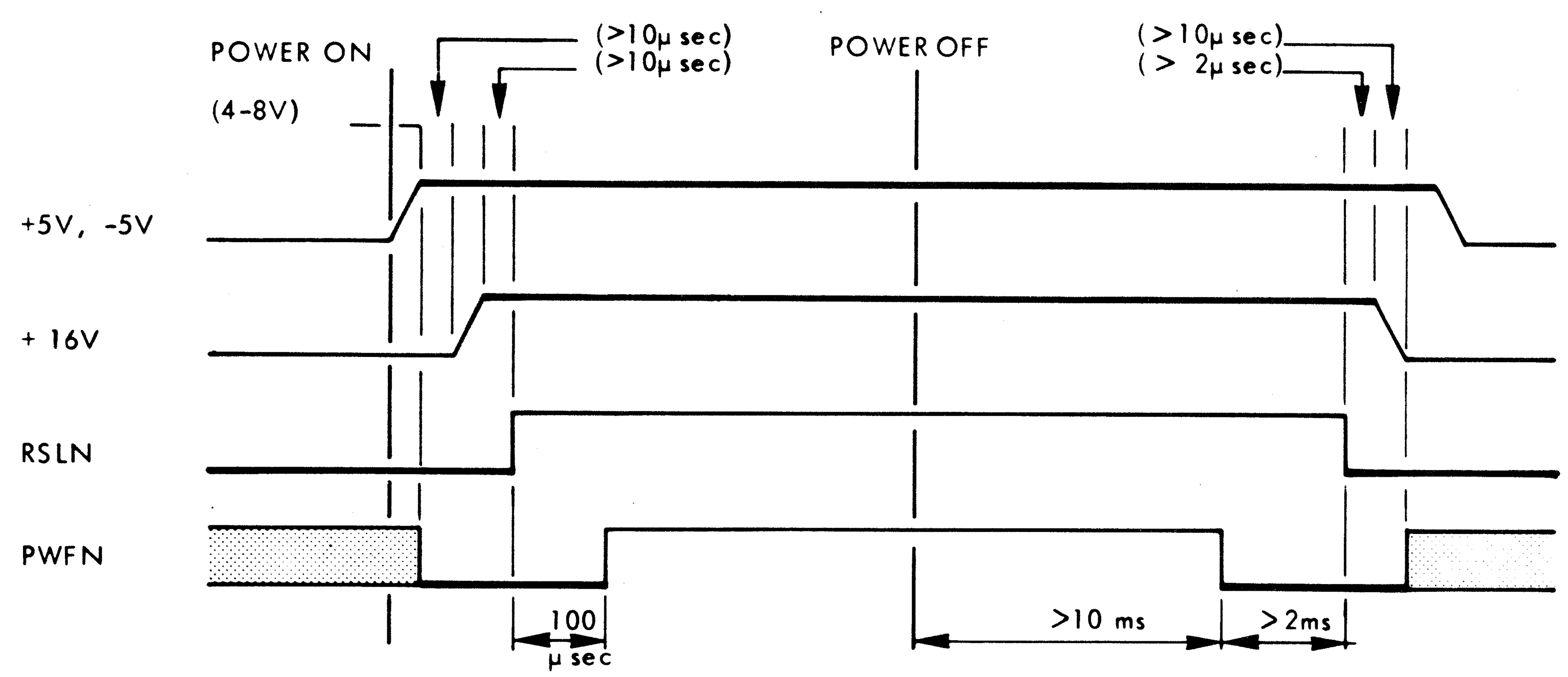
The output voltages, and maximum currents from this supply are:

- +16V, 9A, ±3% (due to ±10% mains and load variation)
- + 5V, 43A, ±2% (due to ±10% mains and load variation) ±2% for
- \- 5V, 2A, ripple and noise, p-p from 0 to 20 MHz.
- +18V, 2A,
- -18V, 2A, -10%, +70% (non regulated)
- +10V, 1A, ±3% (due to ±10% mains and load variations)

The +16V supply is used by the memory (inhibit amplifiers). About 9 amps are required for 32k words of memory; the 16V, 9-amps supplies 64k words of memory with two modules working at the same time, with interleaving. The -5V supply is used by memory (about 1.5 amps) and by the interface drivers for the big disc control unit (about 1 amp). The unregulated +18V and -18V supplies are used by data-communication and teletype control units; the 18 volts are converted to 12 or 6 volts by regulators on the control-unit cards. The +5V supply is used by the CPU logic.

5.5 LOGIC SIGNALS

The RSLN (Reset Line) and PWFN (Power Failure) signals are used for power on/off sequences and automatic restart. The full logic description is given in paragraph 2.45. The power supply controls both signals with the following timing :



The power-off sequence is shown for switching off or for a power failure greater than 10 ms. Shorter line failures will not cause the "off" sequence. Once PWFN goes active (low), the sequence must continue to activate RSLN.

5.6 REAL TIME CLOCK

The Real Time Clock (RTC) is a 1us pulse every 20ms which is sent to the RTCF flip-flop (program status bit 12) to be used as an internal interrupt. The RTC pulse is generated by pulse-shaping the mains frequency. The RTC pulse is then sent via the control-panel key switch (positions ONRTC and LOCK) to the CPU logic RTCF flip-flop (Figure 2-8 PP). The operation of the General Flip-Flop (GF) RTCF is described in Section II. The RTCF interrupt is also shown on the Interrupts and Breaks diagram in Section I.

5.7 FUSES

The power supply contains one fuse in the mains input (F206) and additional fuses for the supplied voltages. The fuses for the supplied voltages (at the rectifier/filter outputs) are to protect the rectifiers and transformer in case of regulator failure; the regulators themselves are electronically protected. The power supply fuses are listed in the following table :

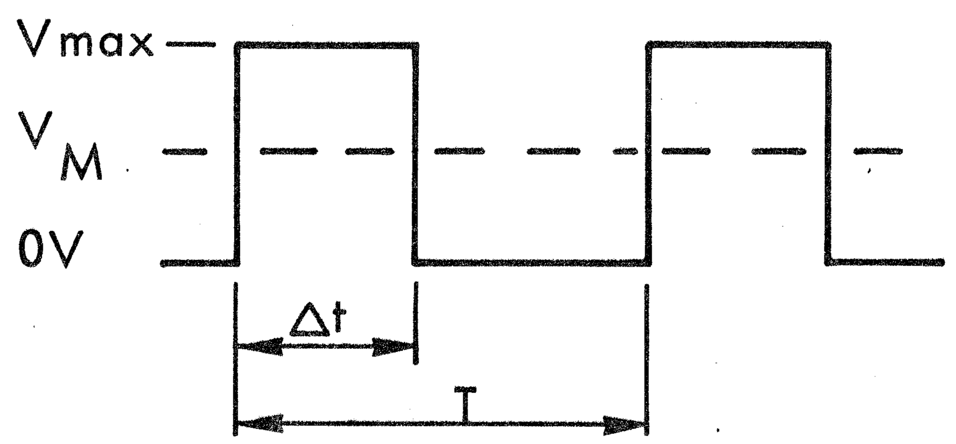
Fuse	Purpose	Schematic	Location
F206	Mains input, slow operating : 4 amp for 220V/240V inputs 8 amp for 100V/115V inputs	sheet 1	
F101	0.1 amp, mains detector	sheet 1	
F201	20 amp, +5V regulator	sheet 2	
F202	10 amp, +16V regulator	sheet 3	
F203	2 amp, auxillary 15V supply for the power-supply circuits.	sheet 2	
F204	3.15 amp, -18V and -5V supplies	sheet 4	
F205	3.15 amp, +18V supply	sheet 4	

5.8 RECTIFIER CIRCUITS

The inputs to all of the DC supplies use center-tapped, full-wave rectifiers. The AC voltage is supplied by five center-tapped secondary windings of transformer TR2. The rectifier circuits are shown on the schematic diagram (Figure 5-3) with their appropriate supplies. The +5V and +16V regulators take the positive voltage from the transformer center taps, which allows the anodes of the high-current diodes to be mounted in heat sinks at zero volts (ground). The +18V and -18V supplies use a full-wave bridge package, but each supply is using only two diodes of the bridge together with the center-tapped transformer winding. The -5V regulator takes its input voltage from the unregulated -18V supply.

5.9 +5V REGULATOR

5.10 Voltage Regulation. The +5V, 43-amp power is supplied by a switching type regulator (Figure 5-3, Sheet 2). High efficiency is obtained because of the regulator transistors (Q201, 202, and 203) being always on or off. The regulator transistors are controlled by a voltage detector and a voltage corrector. The switching frequency is controlled by an independent clock circuit common to the +5 and +16 volt supplies, and is thereby not load-dependent.
The +5V regulator controls the mean voltage V_M (following diagram) by varying the switch-on time in direct relation to the required correction.
The Δt is reduced, while the switching frequency is held constant, in order to reduce the mean output voltage.



$$\frac{V_M \searrow}{V_{max}} = \frac{\Delta t \searrow}{T}$$

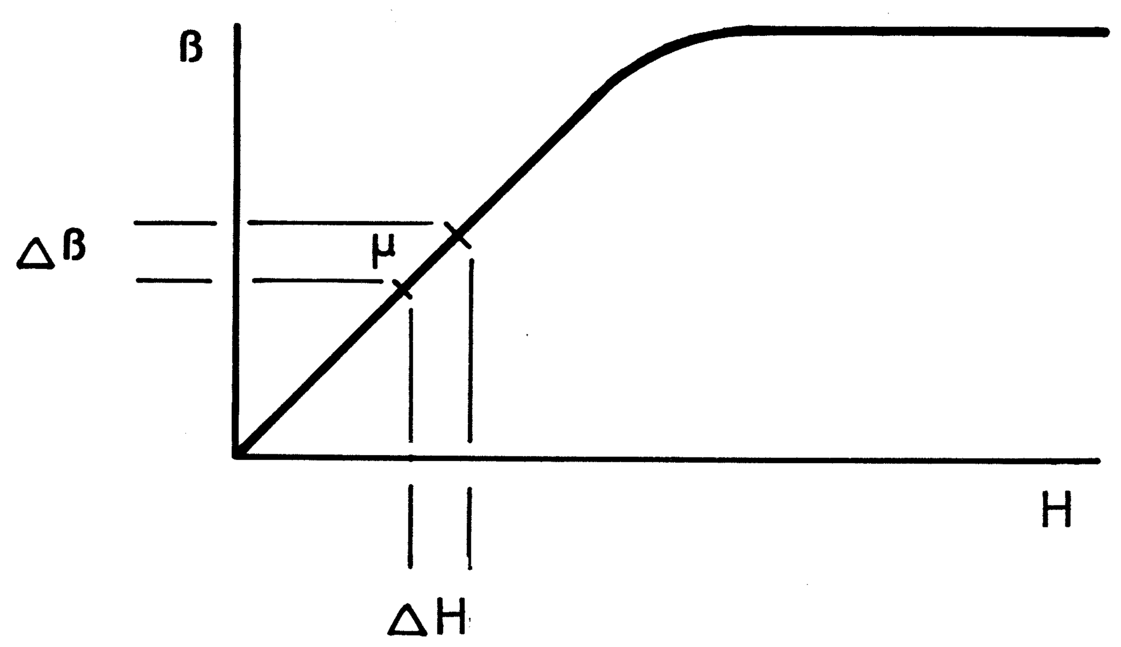
5.11 The switching regulation operates at a frequency (20 KHz, ±5%) determined by the type 555 clock circuit, IC1. The clock provides a timing input to voltage corrector IC2. IC2 is a type 555 pulse-width modulator (PWM) which generates a pulse to switch regulator Q201/2/3 on and off. Regulation is provided by the variable pulse width from IC2. When Q201/2/3 switches on, current flows through inductor L203 to the load, increasing linearly, and charges the output capacitors. When Q201/2/3 switches off, L203 current starts to decrease, and diodes CR201/2 are forward biased; the current now flows through CR201/2, decreasing linearly.

5.12 The +5V output sense voltage is compared to a reference voltage by error amplifier IC4. (Since the type uA723 reference voltage is about 7 volts, the +5V supply (<7V) adjusts the reference level.) As the sense voltage increases, the output current at IC4, pin 9 increases and the Q8 collector voltage to PWM IC2 decreases. The reduced threshold voltage at pin 5 of IC2 decreases the pulse width output from the voltage corrector. The narrower pulse to Q201/2/3 (via transformer T1) results in less "on time" for the regulator transistors, and reduces the final output voltage.

5.13 The 1:1 transformer T1 is used to couple the high-current supply with sensitive, low-current control circuits. This allows higher-voltage regulation control with resultant higher efficiency and higher-speed operation for the voltage regulation. The coupling transformer is supplied with 3.5 volts at high current during power-on time by starting-supply IC5, Q12. Once the +5V supply is up, T1 is supplied by the +5 volts via diode CR1; the emitter of Q12 is back-biased and the starting supply is held off. The starting supply is inhibited (along with the clock) by an overcurrent cutoff.

5.14 Overvoltage Crowbar. An overvoltage detection circuit (crowbar) for the +5V supply is made up of a threshold detector (Q101) and thyristor (CR101). If the supply output voltage increases beyond the nominal value, between 6 and 7 volts, Q101 switches on and the thyristor fires to short-circuit the supply.

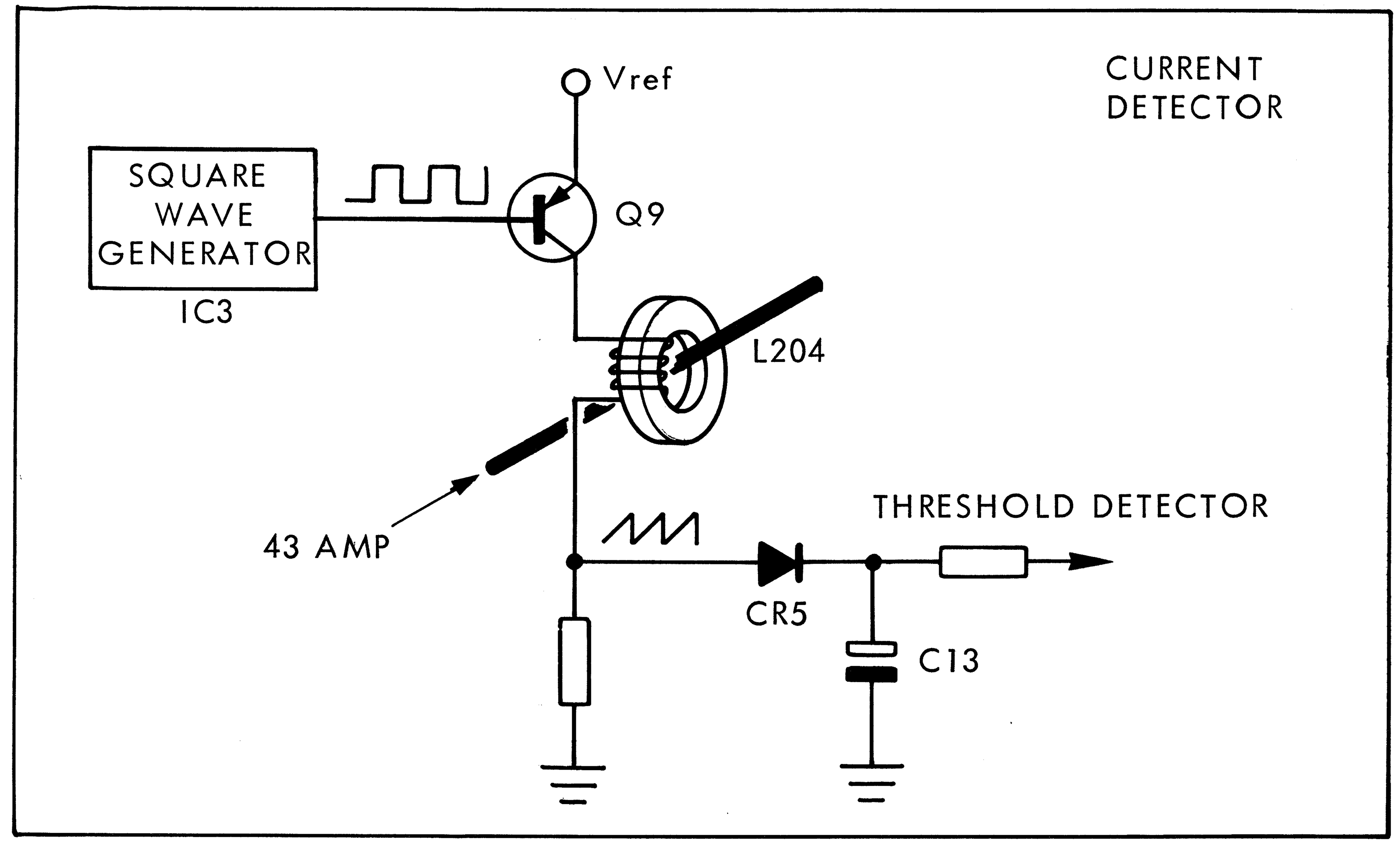
5.15 Overcurrent Detection. Current detection on this 43-amp supply is performed by passing the +5V line through the center of a variable-μ ferrite core (L204) which has a sensing coil wound about it. Direct current through the +5V line produces a magnetic field (H) which causes a magnetic flux in the core. The flux density (B) and the properties of the core material determine the permeability (u) of the core; thus, any change in the current through the core (I) changes the field strength (H) which changes the permeability (u).



$B = \mu H$

$\mu = \frac{B}{H}$

5.16 A type 555 square-wave generator (IC3, Q7, Q9) produces a voltage square wave which passes through the L204 coil (from pin A to B) to the over-current detector (IC6). The inductance (L) of the coil retards the leading edge of the squarewave to produce a current sawtooth output to the detector (following diagram). Since the inductance of the coil (L) is approximately $L = u\,N^2$ (where N is the number of turns, a constant), a change in u causes a change in the sensing current through the coil (ΔI). The height of the sawtooth is thus directly dependant on the coil's L, as follows : $E = L\,\Delta I/\Delta T$, with ΔT and E constants from the squarewave generator. Excess current therefore increases the amplitude of the sawtooth sufficiently to trigger the threshold detector at the IC6 input.



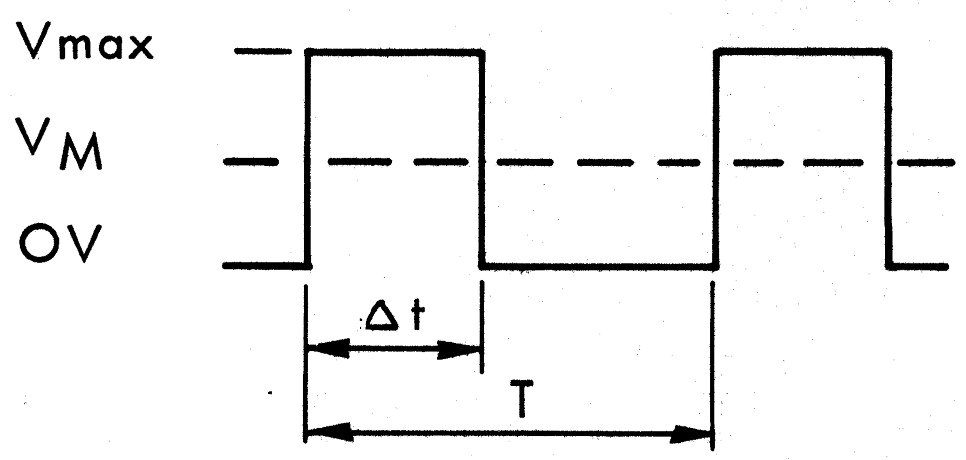
5.17 When overcurrent detector IC6 is triggered, it produces an inhibit signal which blocks the T1 starting supply and the clock. With the clock (IC1) stopped, the +5V and +16V supplies are blocked. The starting supply to T1 must be inhibited separately because it derives its power from the unregulated part of the +5V supply which is not stopped when the IC1 clock stops.

5.18 When the +5 volts is off, the squarewave output is off and transistor Q9 is on. Since the L204 coil is short for DC, the threshold detector detects a constant high and adds to the shut off condition. Negative spikes (caused by L) are removed from the Q9 squarewave output by zener diodes CR14 and CR15. The resistors shown as part of L204 are selected during manufacture to produce the same u for all units. The relationship from current (I) to squarewave voltage is :

$$\underset{\uparrow}{I} \rightarrow \underset{\uparrow}{H} \rightarrow \underset{\downarrow}{u} \rightarrow \underset{\downarrow}{L} \rightarrow \underset{\uparrow}{\Delta I} \rightarrow \underset{\uparrow}{\Delta E}.$$

5.19 +16V REGULATOR

5.20 Voltage Regulation. The +16V, 9-amp power is supplied by a switching type regulator (Figure 5-3, Sheet 3). High efficiency is obtained because regulator transistor Q204 is always on or off. The regulator transistor is controlled by a voltage detector and a voltage corrector. The switching frequency is controlled by an independent clock circuit common to the +5 and +16 volt supplies, and is thereby not load-dependent. The +16V regulator controls the mean voltage V_M (following diagram) by varying the switch-on time in direct relation to the required correction. The Δt is reduced, while the switching frequency is held constant, in order to reduce the mean output voltage.



$$\frac{V_M\downarrow}{V_{max}} = \frac{\Delta t\downarrow}{T}$$

5.21 The switching regulation operates at a frequency (20 KHz) determined by the type 555 clock circuit, IC1. The clock provides a timing input to voltage corrector IC9. IC9 is a type 555 pulse-width modulator (PWM) which generates a pulse to switch regulator Q204 on and off. Regulation is provided by the variable pulse width from IC2. When Q204 switches on, current flows through inductor L202 to the load, increasing linearly, and charges the output capacitors. When Q204 switches off, L202 current starts to decrease, and diode CR203 is forward biased; the current now flows through CR203, decreasing linearly.

5.22 The +16V output sense voltage is compared to a reference voltage by error amplifier IC8. (Since the type uA723 reference voltage is about 7 volts, the +16V supply (>7V) adjusts the sense level.) As the sense voltage increases, the output current at IC8, pin 9 increases and the Q19 collector voltage to PWM IC9 decreases. The decreased threshold voltage at pin 5 of IC9 decreases the pulse width output from the voltage corrector. The narrower pulse to Q204 (via Q20 and Q13) results in less "on time" for the regulator transistor, and reduces the final output voltage.

5.23 Overvoltage Crowbar. An overvoltage detection circuit (crowbar) for the +16V supply is made up of a threshold detector (Q102) and thyristor (CR103). If the supply output voltage increases beyond the nominal value, between 16.8 and 20.3 volts, Q102 switches on and the thyristor fires to short-circuit the supply. The THTR signal is used to shut off the 16-volt supply during the power-off sequence (paragraph 5-41).

5.24 Overcurrent Detection. Current sensing in the +16V supply is performed by resistor R207 in series with the supply. The lower-voltage half of R207 is used as a reference at the pin-4 input of error amplifier IC7. This reference voltage is held constant by zener diode CR8, and an adjustment is provided by trimpot PR4. The higher-voltage half of R207 is used as a sense input at IC7, pin 3. Zener diode CR10 is used to reduce the sense-input voltage. An increase in current through R207 will increase the sense voltage (developed across R45) in relation to the reference voltage. The resultant high output from IC7, pin 2, via Q15, will gate on thyristor CR13 and shut off the +16 volt supply.

5.25 Inhibit Signals. The inhibit signal INH1 delays operation of the +16V regulator during initial power-on time to produce a slow rise of the +16 volts. INH1 is active (high) for approximately 300 ms after power on (Figure 5-5). During this initial delay time, the overcurrent detector is blocked via Q17 and the overvoltage circuit is blocked via Q16 and Q18.

5.26 The inhibit signal INH2 blocks the +16V regulator if either the +5 volts or -5 volts are not present. Loss of either voltage drops out relay K102 (Figure 5-3, Sheet 4) which then grounds the output of the overvoltage corrector circuit (sheet 3).

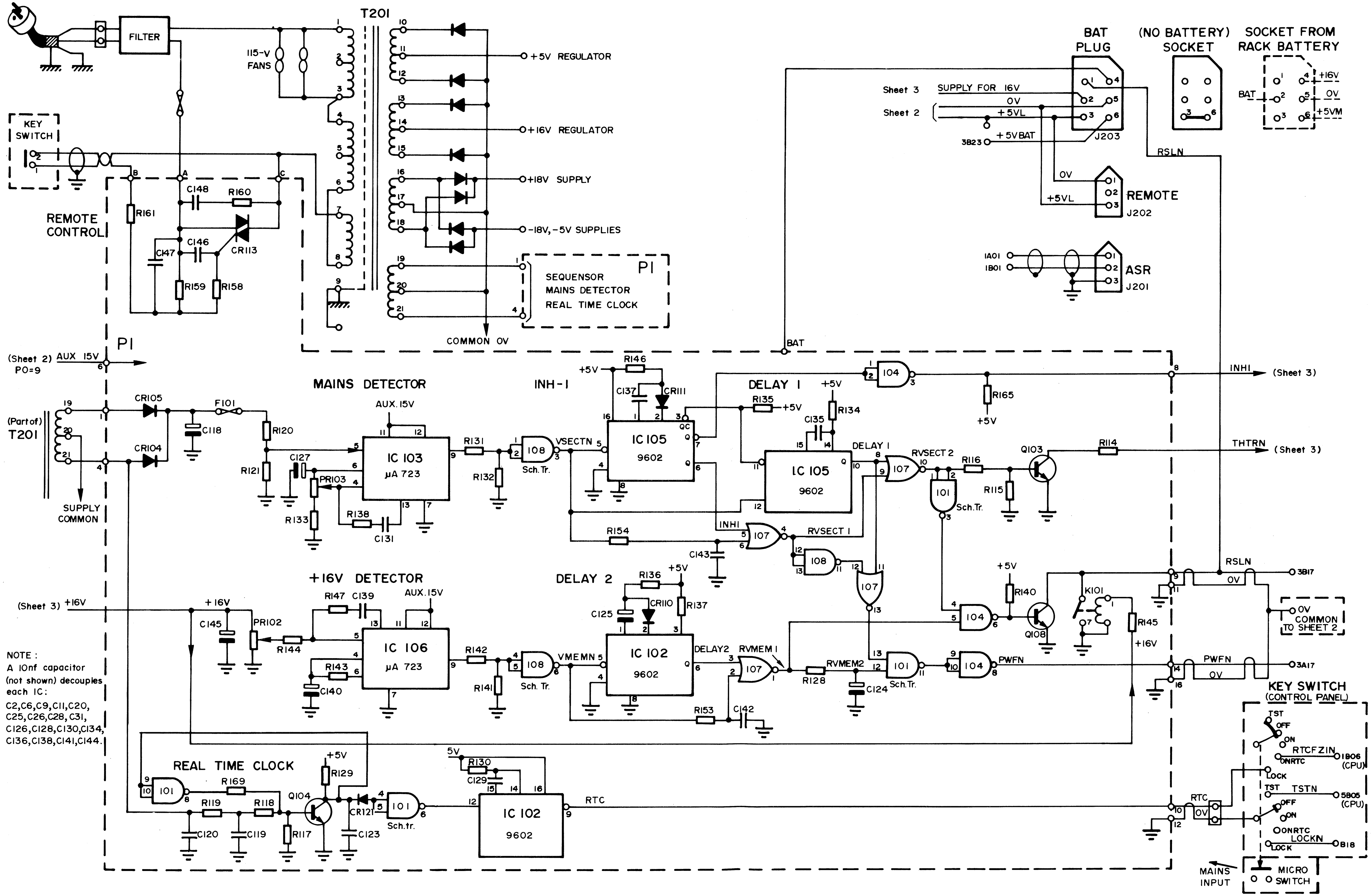


Figure 5-3 (sheet 1) Power Supply Inputs and Sequensor Logic

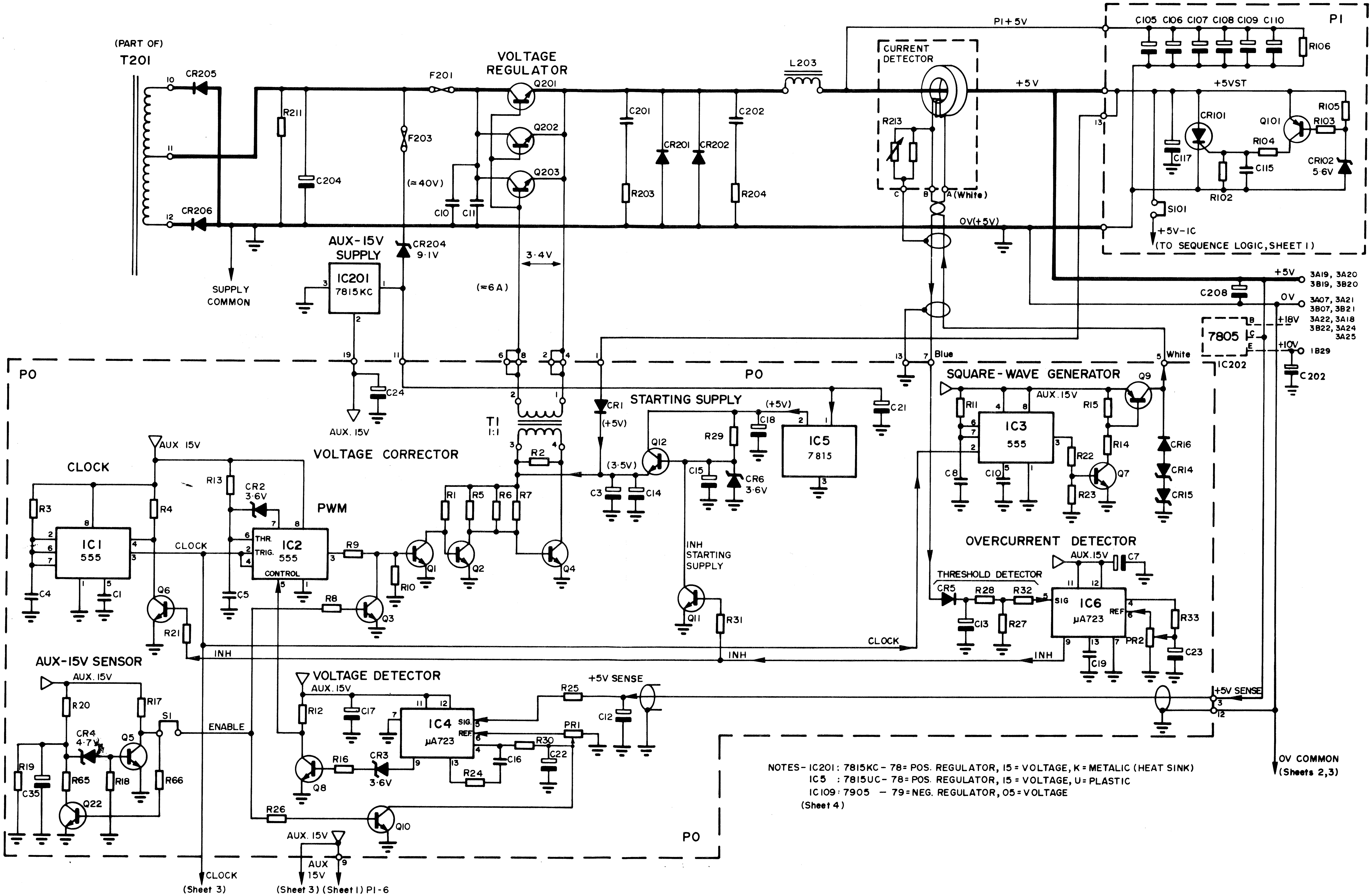


Figure 5-3 (sheet 2) Power Supply +5V Regulator

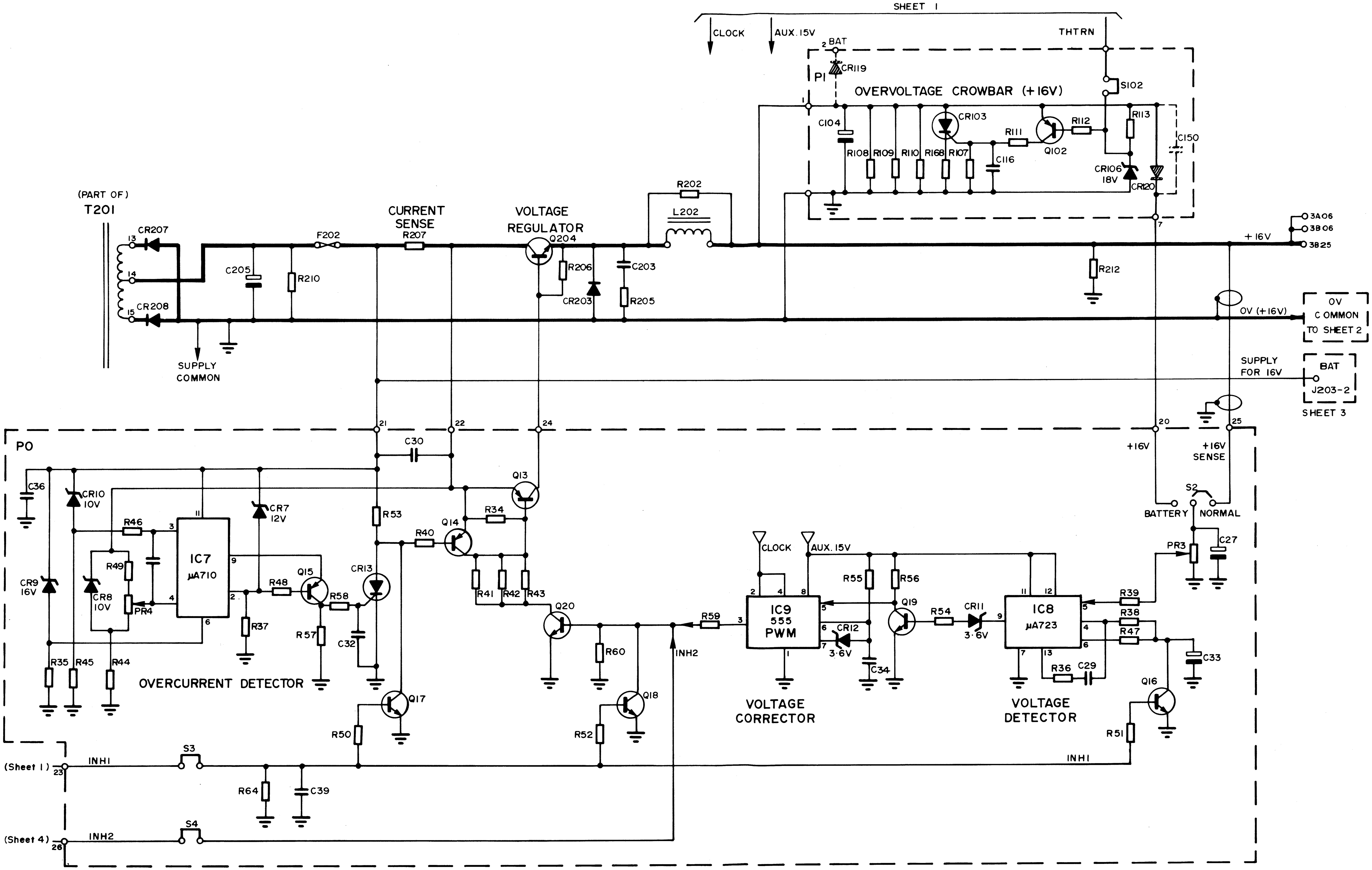


Figure 5-3 (sheet 3) Power Supply +16V Regulator

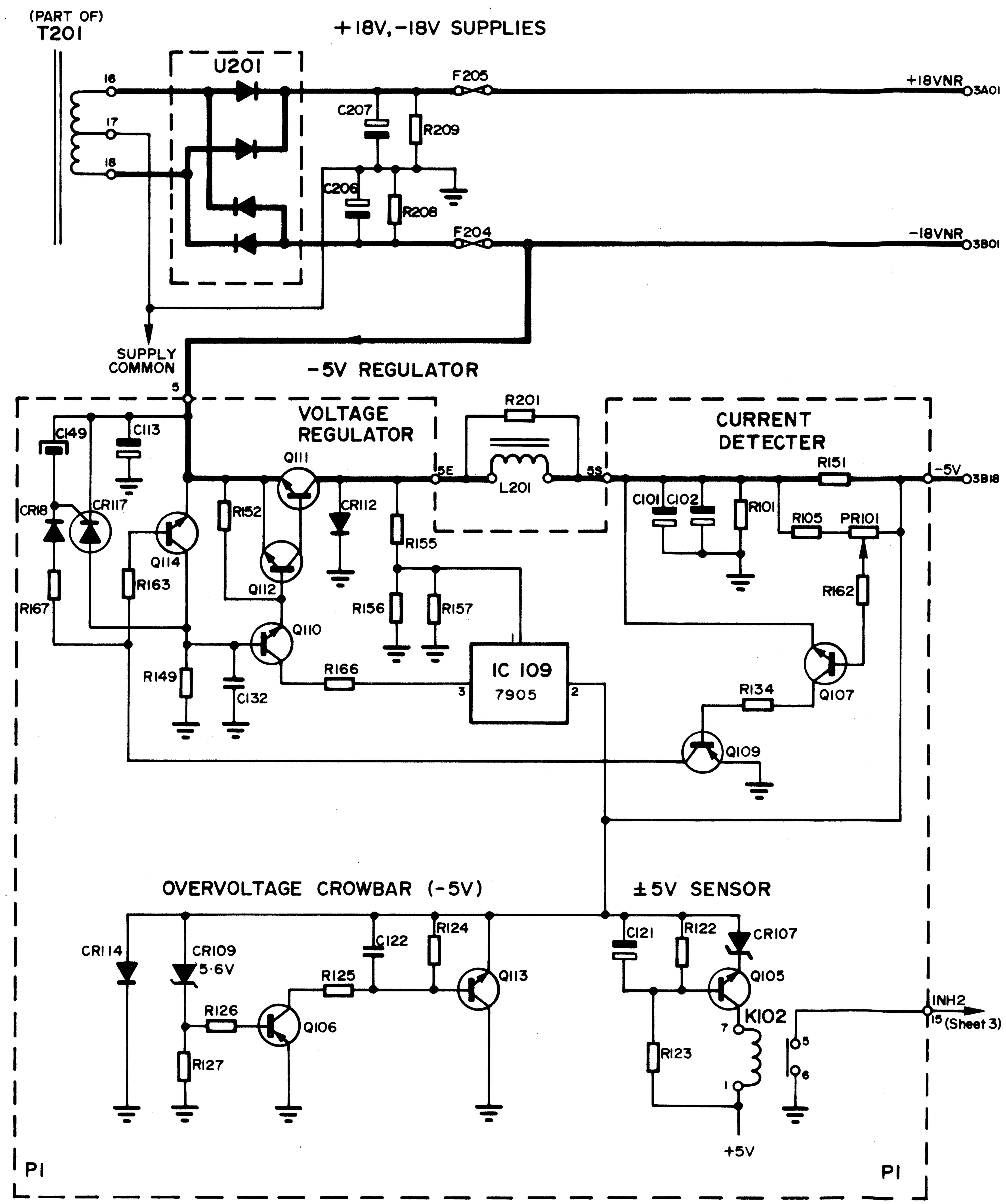


Figure 5-3 (sheet 4) Power Supply +18V, -18V, -5V Supplies

5.27 +10V REGULATOR

The +10V supply for the LOC MOS circuits of the CPU is provided by a series integrated regulator type 7805. It is connected between the +5V and +18V supply lines and provides +10V at a maximum current of 1 ampere which is limited by a circuit within the chip. (See Figure 5-3, sheet 2.)

5.28 Connection for an External Battery Rack

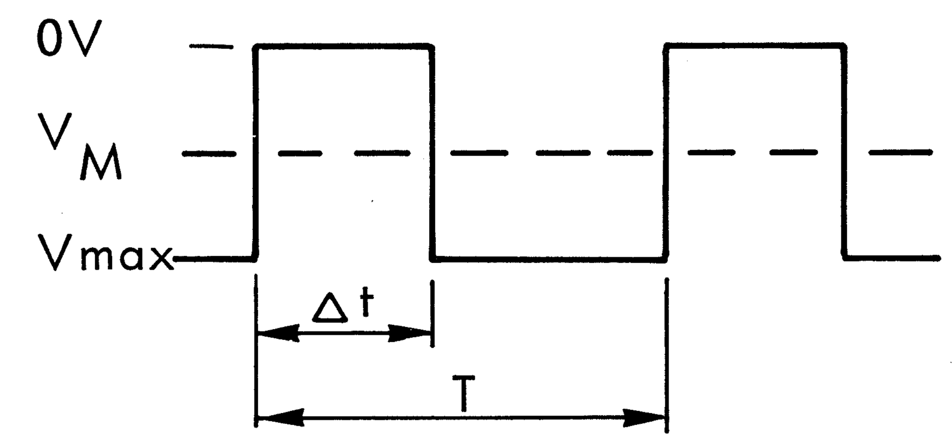
When core memory or both core and MOS memories are used by the CPU an External Battery Rack is not needed and it is dangerous to make such a connection. When only MOS memories are used an External Battery Rack must be connected if the contents of the memories are to be maintained during power failures. Also alterations must be made to the connections of the +16V and S+16V lines as follows:

- Change the position of the U link S2 (Figure 5-3 sheet 3) from NORMAL to the BATTERY position.
- Disconnect the +16V lead on pin 1 of the P1 card and connect it to pin 2 (BAT) on the same card (Figure 5-3 sheet 3).

It is dangerous to use core memories in this configuration because the value of the +16V is increased. When no battery rack is fitted a dummy socket is inserted in the battery plug of the power supply to make the +5VL to +5VM connection (Figure 5-3 sheet 1).

5.29 -5V REGULATOR

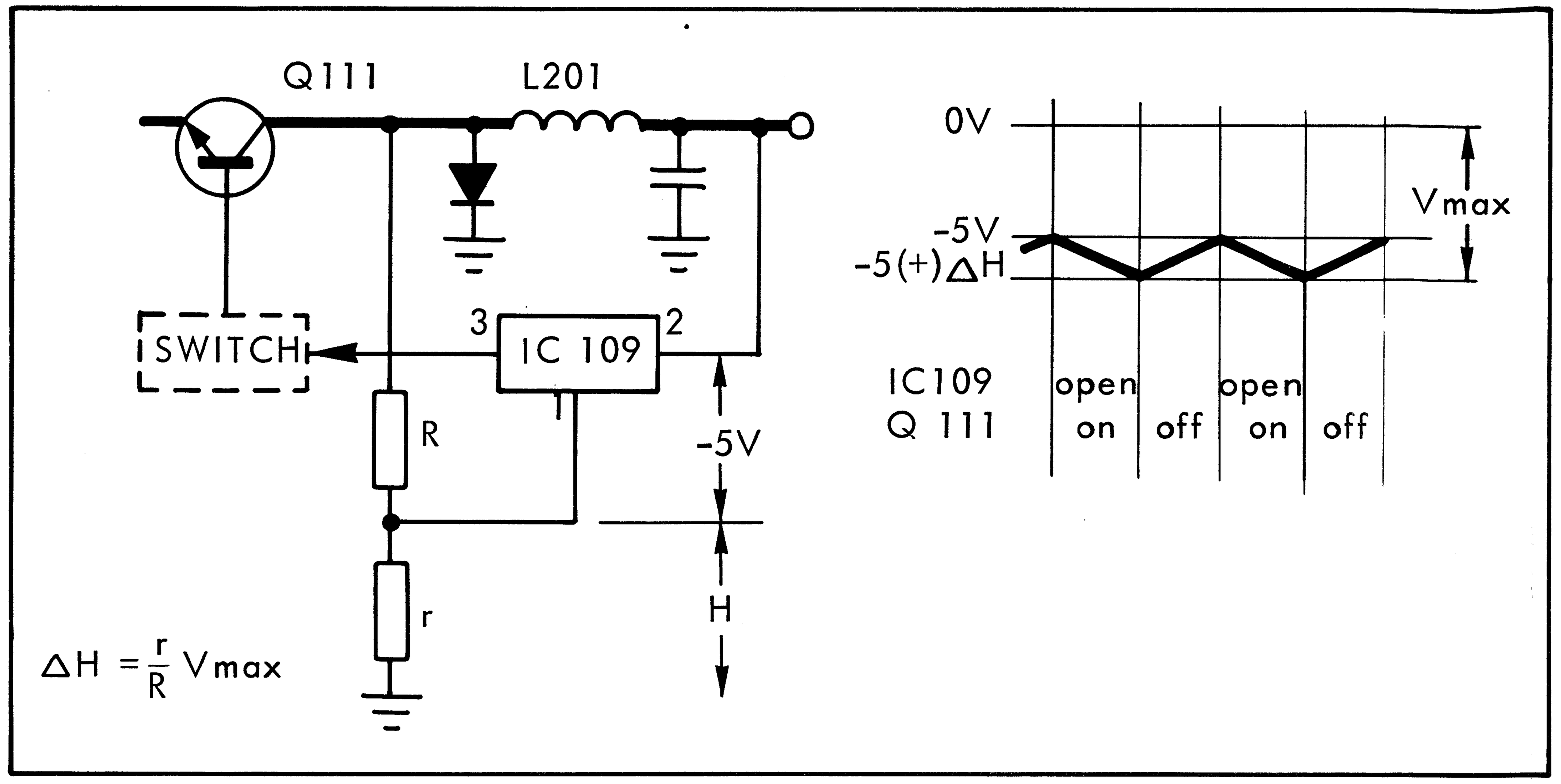
5.30 Voltage Regulation. The -5V, 2-amp power is supplied by a switching type regulator (Figure 5-3, Sheet 4). Transistor Q111 is switched on and off at about 20 kHz by a type 7905 regulator (IC109) used as the voltage detector. The -5V regulator controls the mean voltage V_M (following diagram) by varying the switching frequency T in inverse relation to the required correction. The T is increased, while the on-time Δt is held constant, in order to reduce the mean output voltage.



$$\frac{V_M\searrow}{V_{max}} = \frac{\Delta t}{T\nearrow}$$

5.31 The output voltage is compared to a reference voltage by IC109. The reference voltage at pin 1 is established by the small-value R156/157 relative to the large R155. When IC109 detects less than 5 volts between pins 2 and 1 it switches off, and Q111 is switched on. With Q111 on, diode CR112 is reverse biased and not conducting; current flows through Q111 and L201 to the load, increasing (more negative) linearly by charging the output capacitors C101/102. The capacitors continue charging until the -5V is detected by IC109.

5.32 When IC109 detects more than 5 volts, it switches on and Q111 is switched off. With Q111 off, L201 current starts to decrease and CR112 is forward biased; the current now flows through CR112, and discharging from C101/102, decreases linearly. When the sense voltage falls below the reference, IC109 switches Q111 back on and the cycle repeats.



5.33 Overvoltage Crowbar. An overvoltage detection circuit (crowbar) for the -5V supply is made up of a threshold detector (Q106) and transistor (Q113). If the supply output voltage increases beyond the nominal value, between -6 and -7.5 volts, Q106 switches on and the transistor conducts to short-circuit the supply.

5.34 Overcurrent Detection. The overcurrent circuit comprises sensor R151/R150/PR101, switches Q107 and Q109, thyristor CR115, and switch Q110. An overcurrent through the sensing resistor network produces an increased voltage across the base-emitter of Q107 which switches it on. Transistor Q109 then switches on, and CR115 fires. CR115 switches off Q110 (regardless of the voltage operation through IC109) and holds it off until the current drops back below the maximum of two amps.

5.35 ADJUSTMENTS

The +5V and +16V regulators can each be adjusted for voltage level and overcurrent by trimpots located on circuit card PO. The switching frequency, which is produced by a clock circuit common to these two supplies, is non-adjustable. The -5V regulator can be adjusted, for current protection only, by a trimpot located on circuit card P1.

- +5V output voltage is adjusted by potentiometer PR1.
- +5V overcurrent is adjusted by potentiometer PR2 for a value of 43 amps.
- +16V output voltage is adjusted by potentiometer PR3.
- +16V overcurrent is adjusted by potentiometer PR4 for a value of 9 amps.
- -5V overcurrent is adjusted by potentiometer PR101 for a value of 2 amps.

Power sequence adjustments are made by trimpots located on circuit card P1.

- Power-Off detection time (paragraph 5.39) is adjusted to 10 ms (with mains at 220V) by potentiometer PR103.
- The +16V detector is adjusted by PR102 to switch on when the 16-volt supply reaches 14.7 volts (paragraph 5.37).

5.36 POWER SEQUENCE LOGIC

The power sequencing logic controls the power-on and power-off sequence of the 5-volt and 16-volt supplies and the power logic signals RSLN and PWFN. The logic is included on regulation card P1, and shown on the power supply schematic, Figure 5-3, Sheet 1. A block diagram is shown in Figure 5-4, and Figure 5-5 gives the sequence timing.

5.37 Power-On Sequence. When the power is switched on, the two 5-volt supplies begin to rise; when they reach their nominal value, relay K102 energizes and blocks the INH2 signal (paragraph 5.25). Also at power-on time, the mains detector circuit (amplifier IC103 and schmitt trigger 108-3) generates the VSECTN signal.

5.38 When VSECTN goes low, the INH-1 circuit begins a 300ms delay. During this delay, the INH1 signal blocks turn-on of the +16V regulator (paragraph 5.25). At the end of INH1 time, RVSECT1 is generated while VSECTN and INH1 are both low. Delay 1 is not triggered by the negative-going VSECTN. RVSECT2 drops when RVSECT1 comes on.

5.39 Also at the end of INH1 time, the +16V supply switches on and begins its slow rise. When the 16-volt output reaches 14.7 volts (in about 250 ms), the +16V Detector circuit (amplifier IC106 and schmitt trigger 108-6) generates

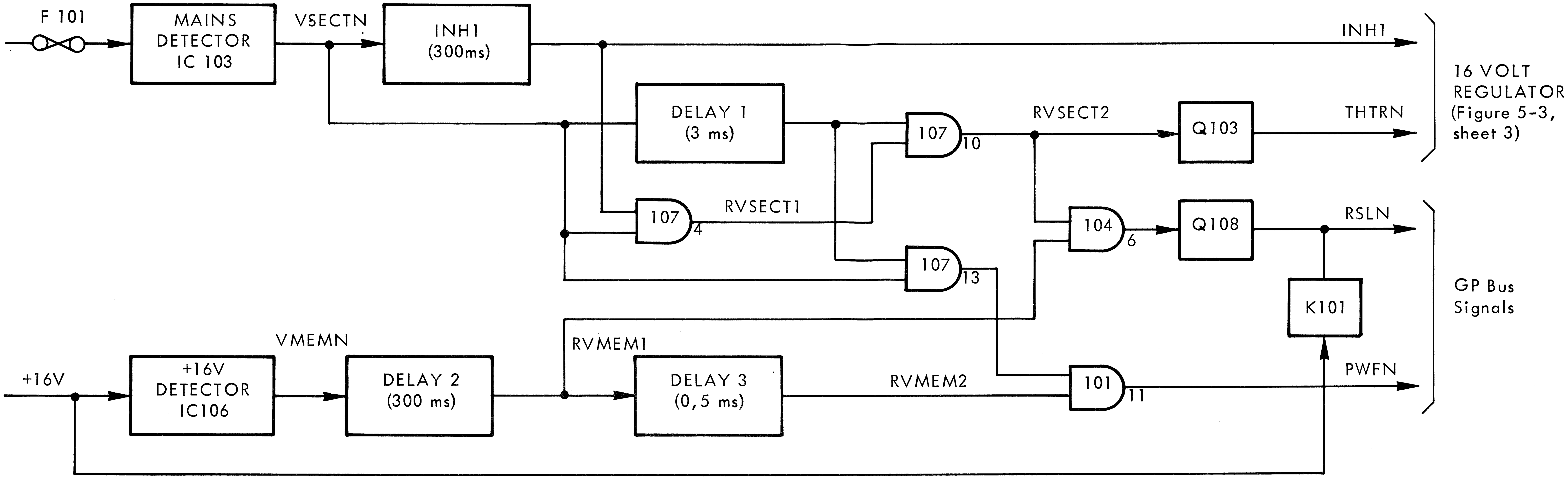


Figure 5-4 Power Sequencing Block Diagram

VMEMN (memory voltage). The leading edge of VMEMN triggers Delay-2. RVMEM1 is generated approximately 300 ms after VMEMN, when Delay-2 drops. RVSECT2 (inverted at gate 101-3) and RVMEM1 together switch off Q108 to terminate the reset-line signal RSLN.

5.40 Delay-3 is an R-C circuit (R128 and C124) which produces the RVMEM2 signal 0.5 ms after the rise of RVMEM1. RVMEM2 terminates the power failure signal PWFN, as long as VSECTN and DELAY1 are both low.

5.41 Power-Off Sequence. The Mains Detector circuit (amplifier IC103 and schmitt trigger 108-3) detects any mains failure longer than 10 ms, and indicates the failure by de-activating the VSECTN signal. The circuit uses the discharge time of C127 through PR103 and R133. The time constant is adjusted by PR103 so that the dropping power reaches the critical level at 10 ms.

5.42 The rising edge of VSECTN drops RVSECT1 without a delay, via the the bypass of the INH-1 circuit. VSECTN immediately generates power-failure signal PWFN via gates 107-12/13, 101-13/11, and 104-9,10/8. The Delay-1 circuit drops 3 ms after RVSECT1 falls, and RVSECT2 goes high. RVSECT2 thus generates reset-line signal RSLN (via gates 101-1,2/3 and 104-4/6) 3 ms after PWFN.

5.43 The RVSECT2 signal switches on transistor Q103 to generate signal THTRN. This signal activates the +16V overvoltage circuit which quickly shuts off the +16V supply. The loss of the +16 volts also drops out relay K101; the closed contacts then ground RSLN, thus holding the signal in its active state while power is off. Since transistor Q108 operates after K101 for power on and before K101 for power off, contact bounce is masked from the reset line.

5.44 The +5V supply begins to drop after the +16V supply is off. The fall time of the +5V supply is about 20 ms with minimum load.

5.45 MECHANICAL

The power-supply chassis is mounted on a base plate in the basic mounting box (Figure 5-6). The power supply output voltages are connected to the system circuit cards via back-panel connector 3. The main power-supply assemblies are shown in Figure 5-7. Most of the electronic circuits are located on two printed-circuit cards, P0 and P1 (Figure 5-8). Power transistors and rectifiers and the fuses are mounted on the heat-sink assembly (Figure 5-9). A few very large components (coils and capacitors) are mounted directly on the chassis base plate. The mains power transformer (T201) and the two fans are mounted on the basic box at the rear. The key switch is mounted on the front of the basic box (Figures 5-6, 5-9). All components not mounted on the two circuit cards have the reference designations 2__.

5.46 Top Cover Removal

Figures 5-6 and 5-7 show the power supply with the top cover removed. The cover is attached by five screws : two on each side and one in the back, at the corner near the fan.

5.47 Power-Supply Chassis Removal

The power -supply chassis (except mains transformer and fans) can be removed on its base plate from the basic mounting box. (Refer to Figure 5-7.)

Disconnect :

- connections at connector 3 (five small plugs, not the two bigger ones; main ground, at connector-3 end).
- RTC connector.
- TR201 outputs 10 through 21.
- Circuit card P0, for access to remote-key connections :
  a. connector plug off.
  b. Remove four screws and spacers onto P1 (one at each corner of P0). A ground wire from the remote key is attached by one of the corner screws.

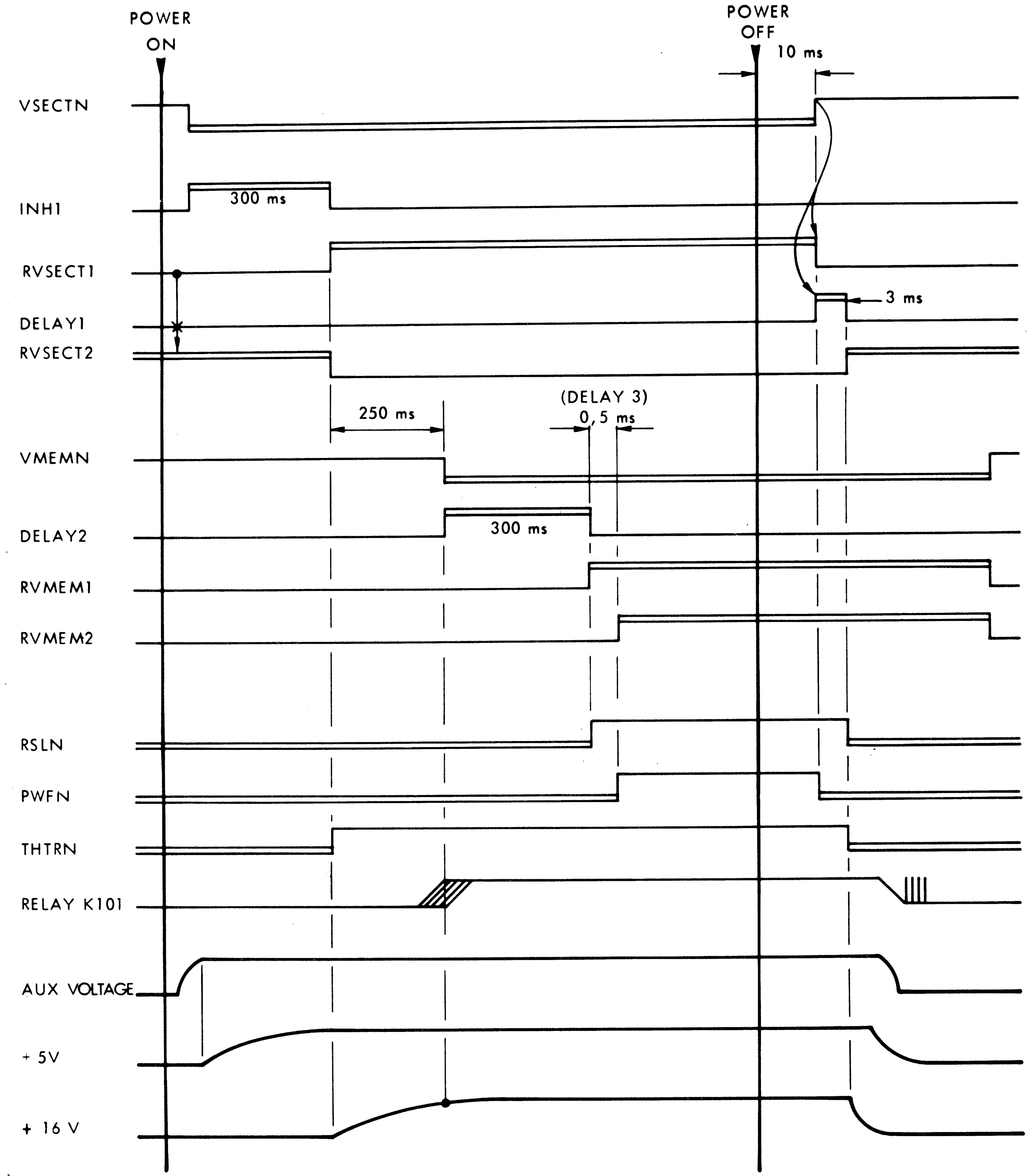
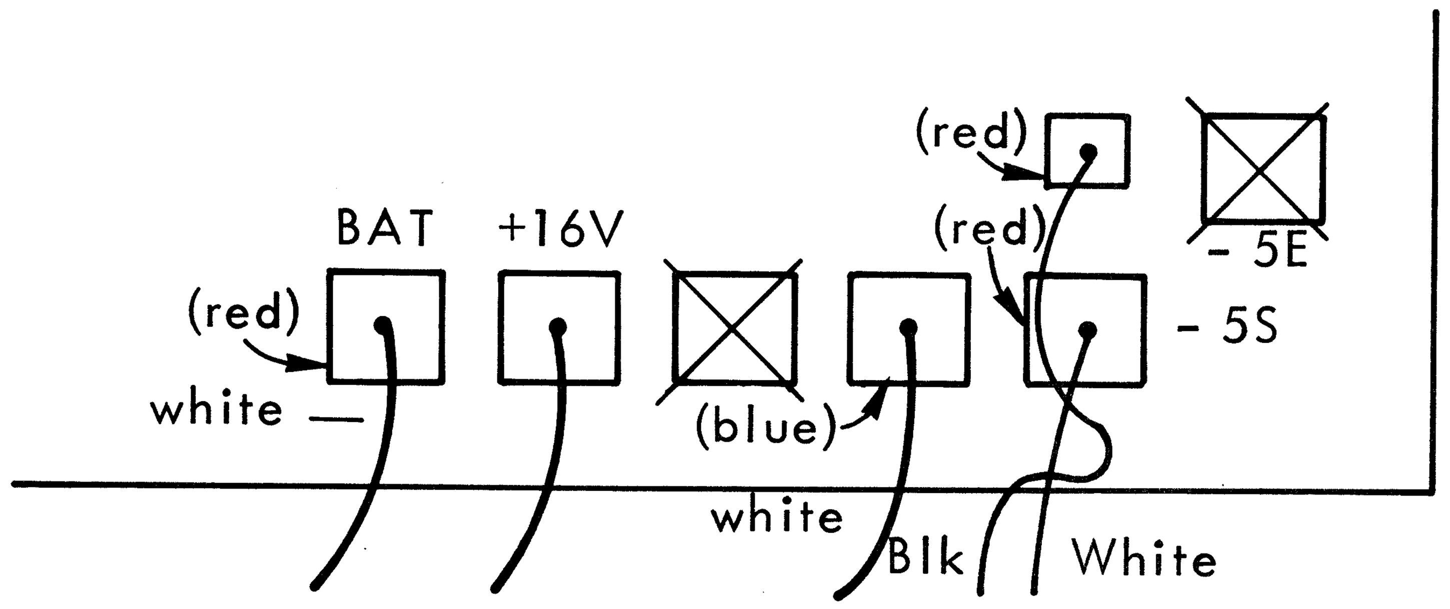


Figure 5-5 Power Sequence Timing

- remote-key connections A, B, C.
- P1 power connections (five plugs) as follows :



- Six screws, three on each side, directly through base plate -- not those through any other brackets mounted on the base plate (however, one does hold a plastic cable clamp).

5.48 Heat-Sink Assembly Removal

The heat-sink assembly can be removed from the basic mounting box. The * indicates the steps already done if the power-supply chassis has been removed from the mounting box. (Refer to Figure 5-7). Disconnect:

* • connections at connector 3 (five small plugs, not the two bigger ones; main ground, at heat-sink end).
  • connector plugs at P0 and P1.
* • RTC connector.
* • TR201 outputs 10 through 21.
  • three cables to C204 and C205 (the two biggest capacitors).
  • Unsolder three wires to C206 and C207 (the two smaller capacitors).
  • +5V cable to L203.
  • +5V power cable (through L204) at both ends.
  • ground connection between heat sink and base plate.
  • four mounting screws, each through a plastic foot bracket, from the base plate.

5.49 Circuit-Card P0 Removal

The P0 circuit card must be removed before the power-supply chassis is removed or the P1 circuit card is removed. P0 can also be removed alone from the basic mounting box. (Refer to Figure 5-7). Disconnect:

- connector plug.
- four screws and spacers onto P1 (one at each corner of P0). A ground wire from the remote key is attached by one of the corner screws.

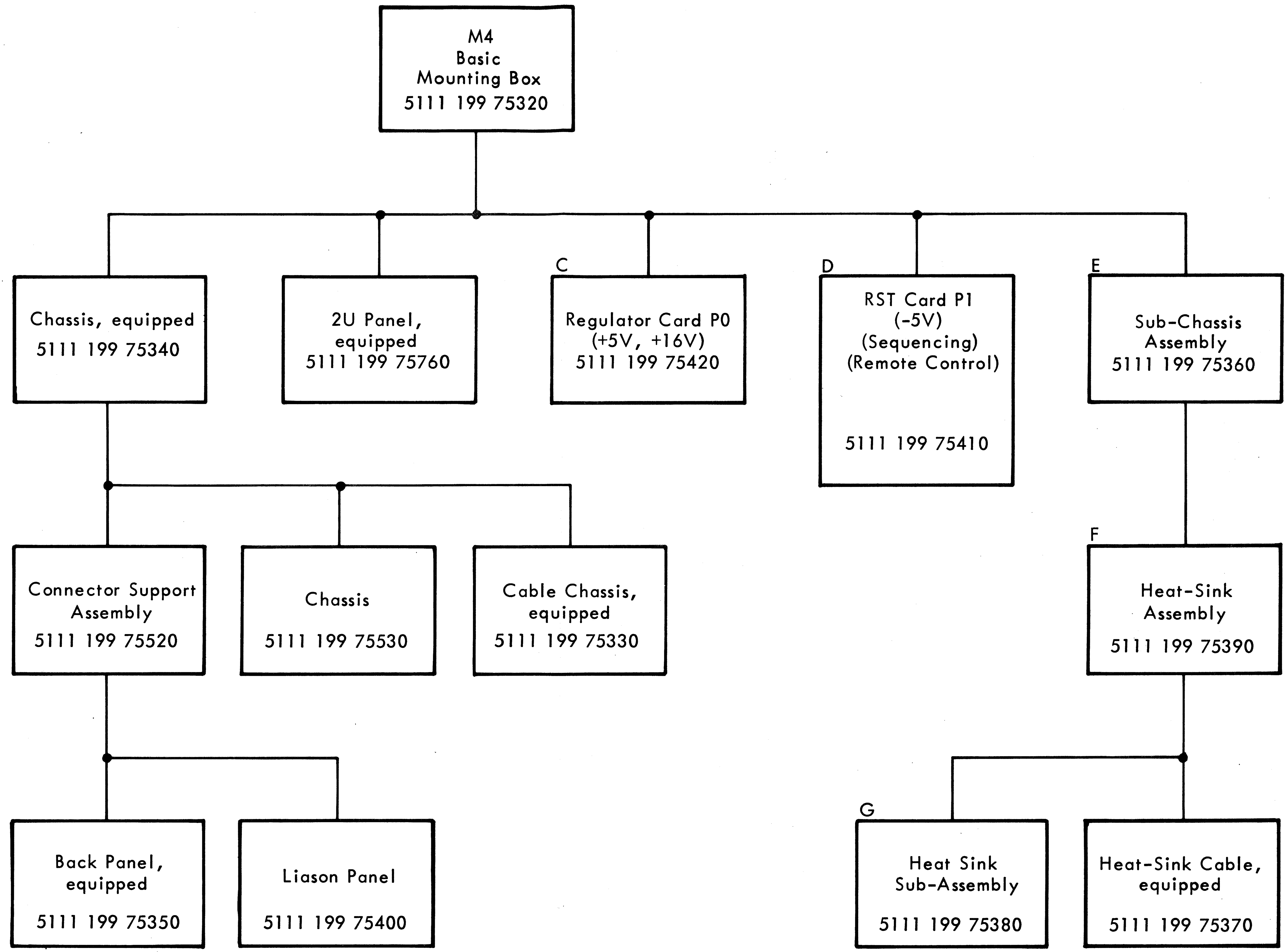
5.50 Circuit-Card P1 Removal

The P1 circuit card can be removed from the power-supply chassis. The * indicates the steps already done if the chassis has been removed from the mounting box. (Refer to Figure 5-7). Disconnect:

* • and remove card P0 (previous paragraph).
* • remote-key connections A, B, C.
  • all eight power connections at connector 3.
  • the +5V cables to L203 and L204 (from the same point on the circuit card).
  • ground cable to the heat sink.

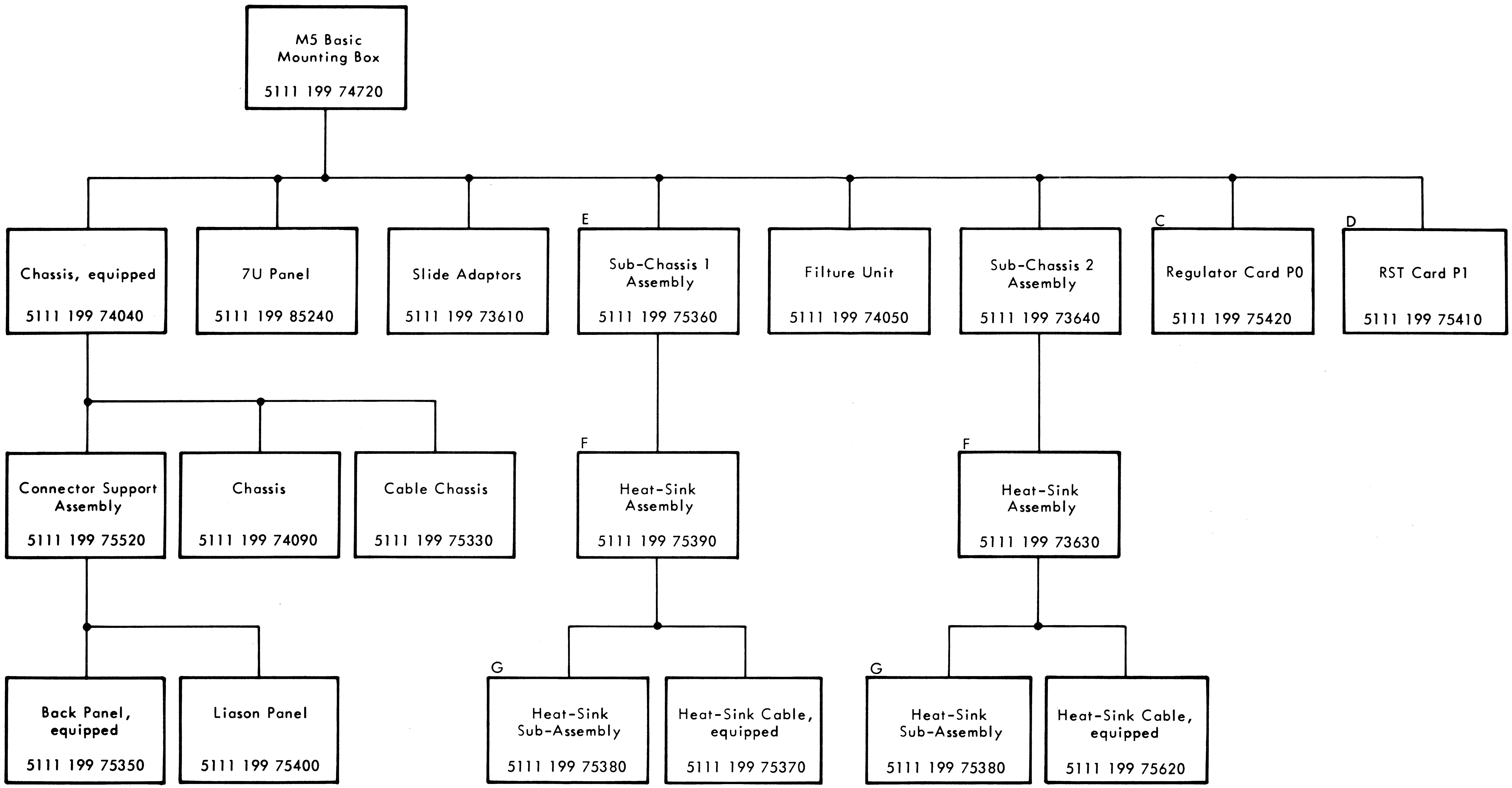
5.51 List of Components

All power-supply components are listed in the following parts-list, Table 5-1.



C to G = Parts Lists Included

Table 5-1A M4 Parts List Guide



C to G = Parts Lists Included

Table 5-1B M5 Parts List Guide

Table 5-1C Regulator Card P0 Parts List

Reference	Description	12NC Code
	Printed Circuit	5111 100 05843
IC1,2,3,9.	Integrated Circuit NE 555 V.	
IC7.	Integrated Circuit µA 710 DC.	
IC4,6,8.	Integrated Circuit UGA 7723 393.	
IC5.	Voltage Regulator 7815 (TO220).	
Q 12.	Transistor 2N 3055 (TO3)	
Q2,4.	Transistor BDX 77 (TO220)	
Q 1,3,7,8,11,17,20.	Transistor 2N2219	
Q 9,14,15.	Transistor 2N2905	
Q 13.	Transistor BDX78 (TO220)	
Q 5,6,10,16,18,19,21,22.	Transistor BSX20	
CR4.	Diode BZX79 C4 V7.	
CR8,10,14,15.	Diode BZX79 C10.	
CR9.	Diode BZX79 C16.	
CR7.	Diode BZX79 C12.	
CR2,3,11,12.	Diode BZX75 C3 V6.	
CR5,16.	Diode BAX 12.	
CR1.	Diode BYX49-300.	
CR6.	Diode BZX79 C5 V6.	
CR13.	Thyristor 2N2323.	
P2,3.	Potentiometer 2600 P102.	
P4.	Potentiometer 2600 P203.	
P1.	Potentiometer 2600 P502.	
R11,38,39,47.	Resistor 1KΩ, 0.25W, 1%.	
R3.	Resistor 5.62KΩ, 0.25W, 1%.	
R13.	Resistor 215Ω, 0.25W, 1%.	
R49.	Resistor 68KΩ, 0.25W, 5%.	
R46.	Resistor 10KΩ, 0.25W, 1%.	
R4,8,12, ,21,24,25,26,29,30,32,33, 36,50,51,52,56.	Resistor 1KΩ, 0.25W, 5%.	
R15.	Resistor 200Ω, 0.25W, 5% .	
R14.	Resistor 1.2KΩ, 0.25W, 5% .	
R16,31,48,54,58.	Resistor 2KΩ, 0.25W, 5%.	
R9.	Resistor 5.1KΩ, 0.25W, 5%.	
R10.	Resistor 510Ω, 0,25W, 5%.	
R60,62.	Resistor 100Ω, 0,25W, 5%.	
R57.	Resistor 4.7KΩ, 0,25W, 5%.	
R40.	Resistor 8.2KΩ, 0,25W, 5%.	
R59,61.	Resistor 390Ω, 0,25W, 5%.	
R20.	Resistor 470Ω, 0,25W, 5%.	

Table 5-1C Contd.

Reference	Description	12NC Code
R22,27,28.	Resistor 10KΩ, 0.25W, 5%.	
R18.	Resistor 150Ω, 0.25W, 5%.	
R17,23.	Resistor 1.5KΩ, 0.25W, 5%.	
R35,37,44,45.	Resistor 3.83KΩ, 0.5W, 1%.	
R1.	Resistor 200Ω, 0.5W, 5%.	
R34.	Resistor 20Ω, 0.5W, 5%.	
R53.	Resistor 3KΩ, 0.5W, 5%.	
R2.	Resistor 47Ω, 10%, WR0617E.	
R41,42,43.	Resistor 1.5KΩ, 5%, WR0617E.	
R5,6,7.	Resistor 33Ω, 5%, WR0825E.	
R63.	Resistor 620Ω, 0.25W, 5%.	
R64.	Resistor 3KΩ, 0.25W, ± 5%.	
R55.	Resistor 1.47K, 0.25W, 1%.	
R65.	Resistor 1.21K 0.125W ± 1%.	
R66.	Resistor 10KΩ, 0.25W, ± 1%.	
R19	Resistor 680Ω 0.25W ± 5%	
C4,8.	Capacitor 0,01µF, 125V, 1%.	
C3.	Capacitor 1500µF, 10V.	
C2,6,9,11,20,25,26,28,31,39.	Capacitor 10nF, ceramic	
C19.	Capacitor 100pF, ceramic	
C5,13,16,30.	Capacitor 0,1µF, 100V, 10%, MPR	
C1,10.	Capacitor 0,01µF, 250V, 10%, MPR	
C7,17,18,24,27.	Capacitor 15µF, 20V, CTS13.	
C12,15,C22.	Capacitor 33µF, 10V, CTS13.	
C23.	Capacitor 3.3µF, 16V, CTS13.	
C14.	Capacitor 220µF, 25V, FITCO.	
C33.	Capacitor 100µF, 10V, FITCO.	
C32,34.	Capacitor 0.022µF, 250V, MPR.	
C37.	Capacitor 470pF, 10%, ceramic	
C29.	Capacitor 0,47µF, 100V, 10%, MPR.	
C38.	Capacitor 10,000pF, 100V, ±10%, ceramic	
C35.	Capacitor 10µF, 25V, FITCO.	
	Heat Sink PB1-2U.	
	Mica insulator 56325.	
T 1.	Transformer TRI M4.	

Table 5-1D .RST Card P1 Parts List

Reference	Description	12NC Code
	Printed Circuit	5111 100 05854
IC103,106.	Integrated Circuit UGA 7723 393.	
IC101,108.	Integrated Circuit 74132.	
IC102,105.	Integrated Circuit 9602.	
IC107.	Integrated Circuit 7402.	
IC104.	Integrated Circuit 1801.	
IC109.	Regulator 7905 (TO220).	
Q 111,113.	Transistor BDX77 (TO220).	
Q 103,112.	Transistor 2N2219.	
Q 109.	Transistor 2N2905.	
Q 104,107.	Transistor BSX20.	
Q 105,108,110,114.	Transistor BSX60.	
Q 101,102,106.	Transistor 2N2906.	
CR 107,116.	Diode BZX79 C4 V7.	
CR 102,109.	Diode BZX79 C5 V6.	
CR 106.	Diode BZX79 C18 or C 20	(later version)
CR 114.	Diode IN4005.	
CR 104,105,110,111,118, 121.	Diode BAX12.	
CR 112.	Diode MR820.	
CR 101,103.	Thyristor BTW92/600RM.	
CR 117.	Thyristor 2N2323.	
CR 113.	Triac TXAL 615 M.	
CR 119	Diode BYZ 14-50	(later version)
CR 120	Diode BZX 75 C2V1	(later version)
P 101,103.	Potentiometer 2600 P 102.	
P 102.	Potentiometer 2600 P 502.	
R 105,127.	Resistor 46.4Ω, 0.25W, 1%.	
R 113.	Resistor 121Ω, 0.25W, 1%.	
R 134.	Resistor 19.6KΩ, 0.25W, 1%.	
R 117.	Resistor 5.11KΩ, 0.25W, 1%.	
R 118,119.	Resistor 4.22KΩ, 0.25W, 1%.	
R 130.	Resistor 31.6KΩ, 0.25W, 1%.	
R 115,116,135,137,143,144,167.	Resistor 1 KΩ, 0.25W, 5%.	
R 123.	Resistor 270Ω, 0.25W, 5%.	
R 133,150	Resistor 510Ω, 0.25W, 5%.	
R 103,111,112,126,128,131,142,153, 154,160,161,166.	Resistor 100Ω, 0.25W, 5%.	
R 124.	Resistor 300Ω, 0.25W, 5%.	
R 158.	Resistor 51Ω, 0.25W, 5%.	
R 132,141.	Resistor 330Ω, 0.25W, 5%.	
R 136,146.	Resistor 30KΩ, 0.25W, 5%.	

Table 5-1D Contd.

Reference	Description	12NC Code
R 159.	Resistor 470Ω, 0.25W, 5%.	
R 156,157.	Resistor 1Ω, 0.25W, 5%.	
R 122.	Resistor 680Ω, 0.25W, 5%.	
R 125.	Resistor 33Ω, 0.25W, 5%.	
R 145.	Resistor 120Ω, 0.25W, 5%.	
R 138,147.	Resistor 2.4KΩ, 0.25W, 5%.	
R 101,106.	Resistor 100Ω, 0.5W, 5%.	
R 104,152,192.	Resistor 10Ω, 0.5W, 5%.	
R 149.	Resistor 1KΩ, 0.5W, 5%.	
R 114.	Resistor 200Ω, 0.5W, 5%.	
R 102,107.	Resistor 300Ω, 0.5W, 5%.	
R 120.	Resistor 510Ω, 0.5W, 5%.	
R 121.	Resistor 240Ω, 0.5W, 5%.	
R 108,109,110.	Resistor WRO617E 560	
R 151,168.	Resistor 0.3Ω, 5%, 224E.	
R 139.	Resistor 10KΩ, 0.25W, 5%.	
R 164.	Resistor 2KΩ, 0.25W, ± 5%.	
R 165.	Resistor 390Ω, 0.125W, 5%.	
R 162.	Resistor 6.8KΩ, 0.25W, 5%.	
R 140.	Resistor 200Ω, 0.25W, 5%.	
R 163.	Resistor 20KΩ, 0.25W, 5%.	
R 155.	Adjustable Resistor 510Ω to 2 KΩ, 0.5W, 5%, E 24.	
R 169.	Resistor 2.7KΩ, 0.25W, 5%	(later version)
R 129	Resistor 5.1KΩ	
C 123,126,128,130,134,136,138,141, 144,190.	Capacitor 10nF, ceramic	
C 131,139.	Capacitor 560pF, ceramic	
C 142,143.	Capacitor 1000pF, ceramic	
C 129.	Capacitor 56pF, ceramic	
C 115,116,120,122. 119.	Capacitor 0.1μF, 100V, 10%, MPR	(later version)
C 119,132.	Capacitor 0.01μF, 250V, 10%, MPR	
C 135.	Capacitor 0.47μF, 100V, 10%, MPR	
C 117,125,137.	Capacitor 33μF, 10V, CTS13	
C 124,127,140.	Capacitor 3.3μF, 16V, CTS13	
C 118,121,133,145,149.	Capacitor 22μF, 25V, FITCO	
C 146.	Capacitor 47000pF, 400V, 20%, PMA	
C 147.	Capacitor 22000pF, 400V, 20%, PMA	
C 148.	Capacitor 0.1μF, 630V, 20%, PMA	
C 104.	Capacitor 680μF, 25V.	
C 105-114.	Capacitor 1500μF, 10V.	
C 101,102.	Capacitor 100μF, 10V, FITCO	
C 150	Capacitor 22 μF 10V.	(later version)

Table 5-1D Contd.

Reference	Description	12NC Code
K 101.	Relay MRMD 15006.	
K 102.	Relay MRMD 15005.	
F 101.	Fuse D1/0.1.	
	Mica insulator 56325	
	Triac Heat-Sink	5111 100 22541.

Table 5-1E Sub-Chassis Assembly Parts List

Reference	Description	12NC Code
	Sub-Chassis	5111 100 22312
	Heat-Sink Assembly	5111 199 75390
L201.	Inductance SLF 2091	
L202.	Inductance SLF 1712	
L203.	Inductance SLF 2101	
C204-C205.	Capacitor 33000μF, 40V.	
C206-C207.	Capacitor 5000+5000μF, 40V.	
R 208-209.	Resistor 1KΩ, WR0617E.	
R210-211.	Resistor 3.9KΩ, 0.5W, 5%.	
R202.	Resistor 560Ω, 0.5W, 5°.	
R212.	Resistor 30Ω, 25 W, ± 3%.	
R213.	Adjustable Resistor 22Ω to 3KΩ, 0.5W, ± 5%. or 300Ω - 910Ω	(later version)

Table 5-1F Heat Sink Assembly Parts List

Reference	Description	12NC Code
	Heat-Sink Sub-Assembly	5111 199 75380
	Harness (M4, M5) or	5111 199 75370
	Harness (M5).	5111 199 73630
R203, 204, 205.	Resistor 10Ω, 0.5W, 5%.	
R206.	Resistor 27Ω, 0.5W, 5%.	
C201, 202, 203.	Capacitor 0.022μF, 250V, 10%, PMA.	
C210, 211.	Capacitor 10 μF, 100V, 20%, PMA.	
C212	Capacitor 3,3 μF 25V	Late Version
F201.	Fuse A13/20.	
F202.	Fuse D8/10.	
F203.	Fuse D1/2.	
F204, 205.	Fuse D1/3.15.	
REG 1	Regulator 7805 (TO22)	(later version)

Table 5-1G Heat-Sink Sub-Assembly Parts List

Reference	Description	12NC Code
	Heat Sink	5111 100 22523
L204.	Inductance SLF 2541.	
IC 201.	Regulator 7815 (T03).	
Q 201, 202, 203, 204.	Transistor 2N5685.	
CR201, 202.	Diode RPR 1040 R.	
CR203.	Diode IN3910 R.	
CR204.	Diode BZY93 C9V1.	
CR205, 206, 207, 208.	Diode BYX52/300R.	
U 201.	Bridge MDA 952-2.	
R 207.	Resistor 0.1 Ω, 3%, (RH10).	
	Relay REVIC ID.	

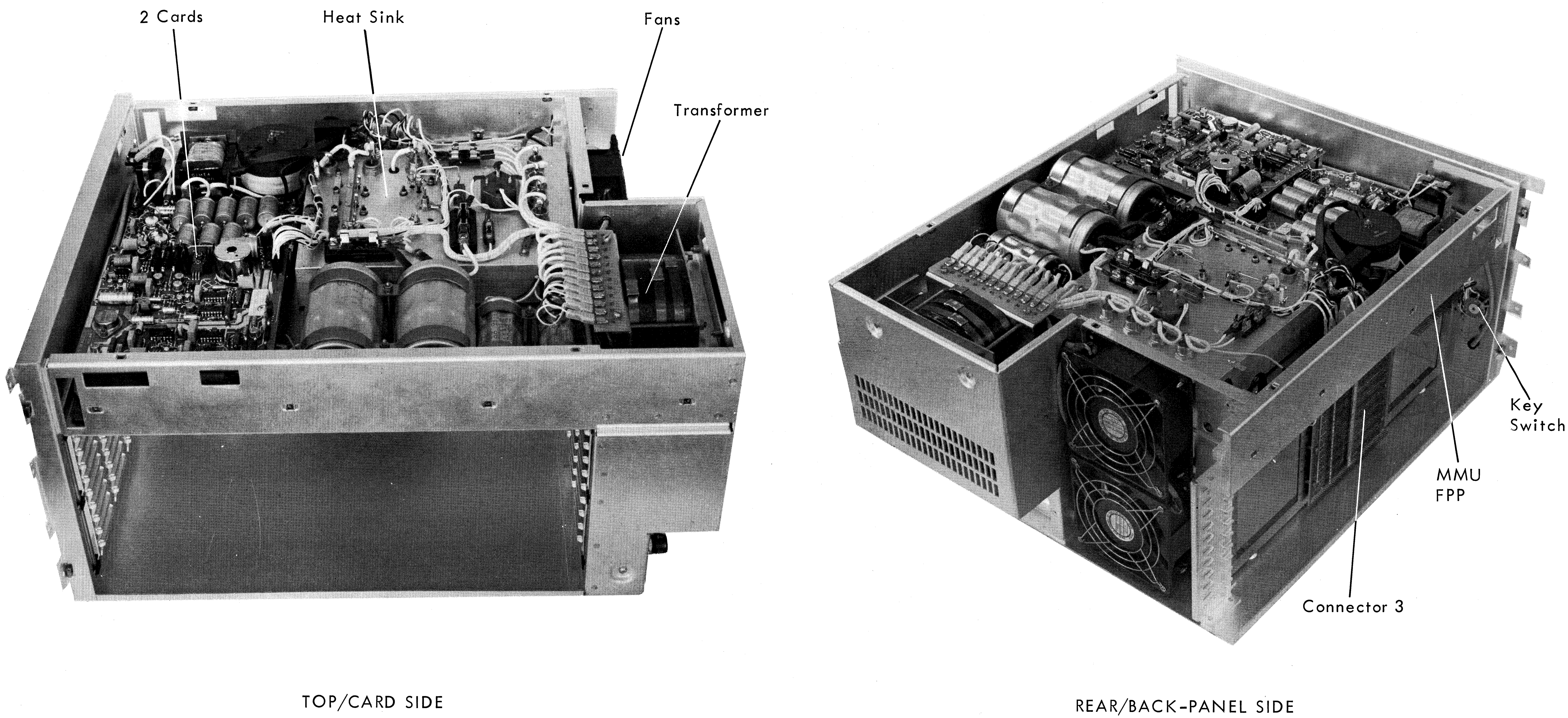
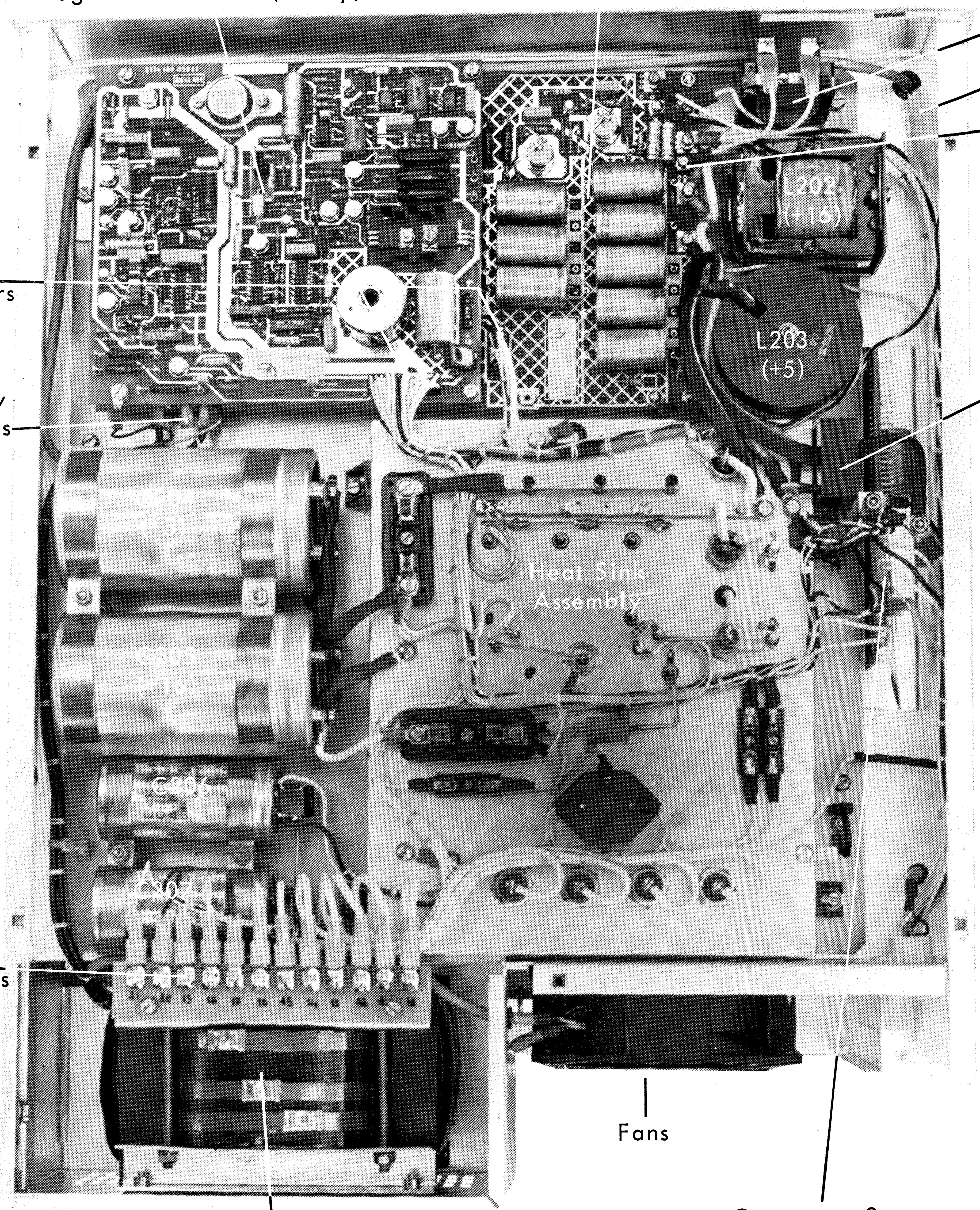


Figure 5-6 Basic Mounting Box



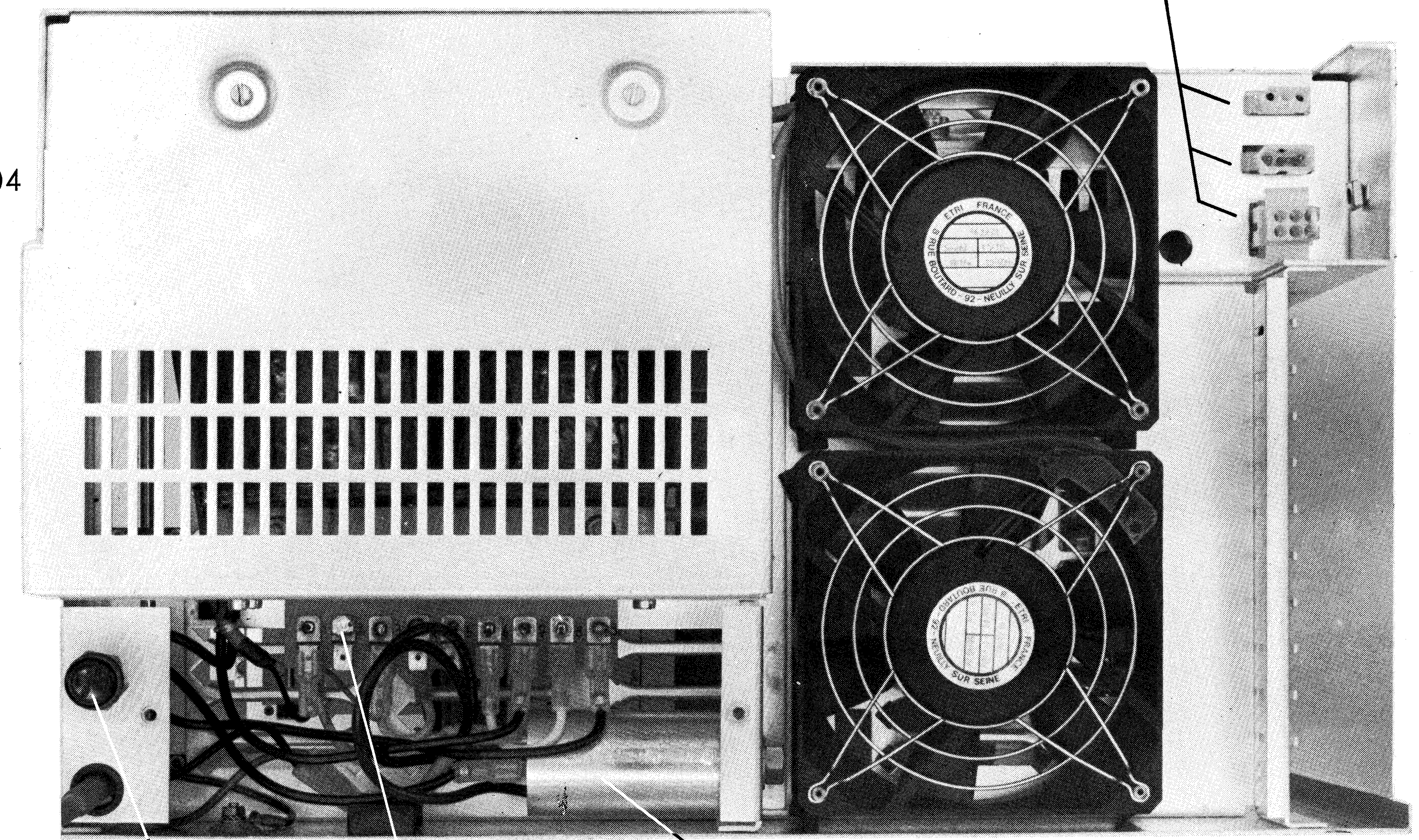
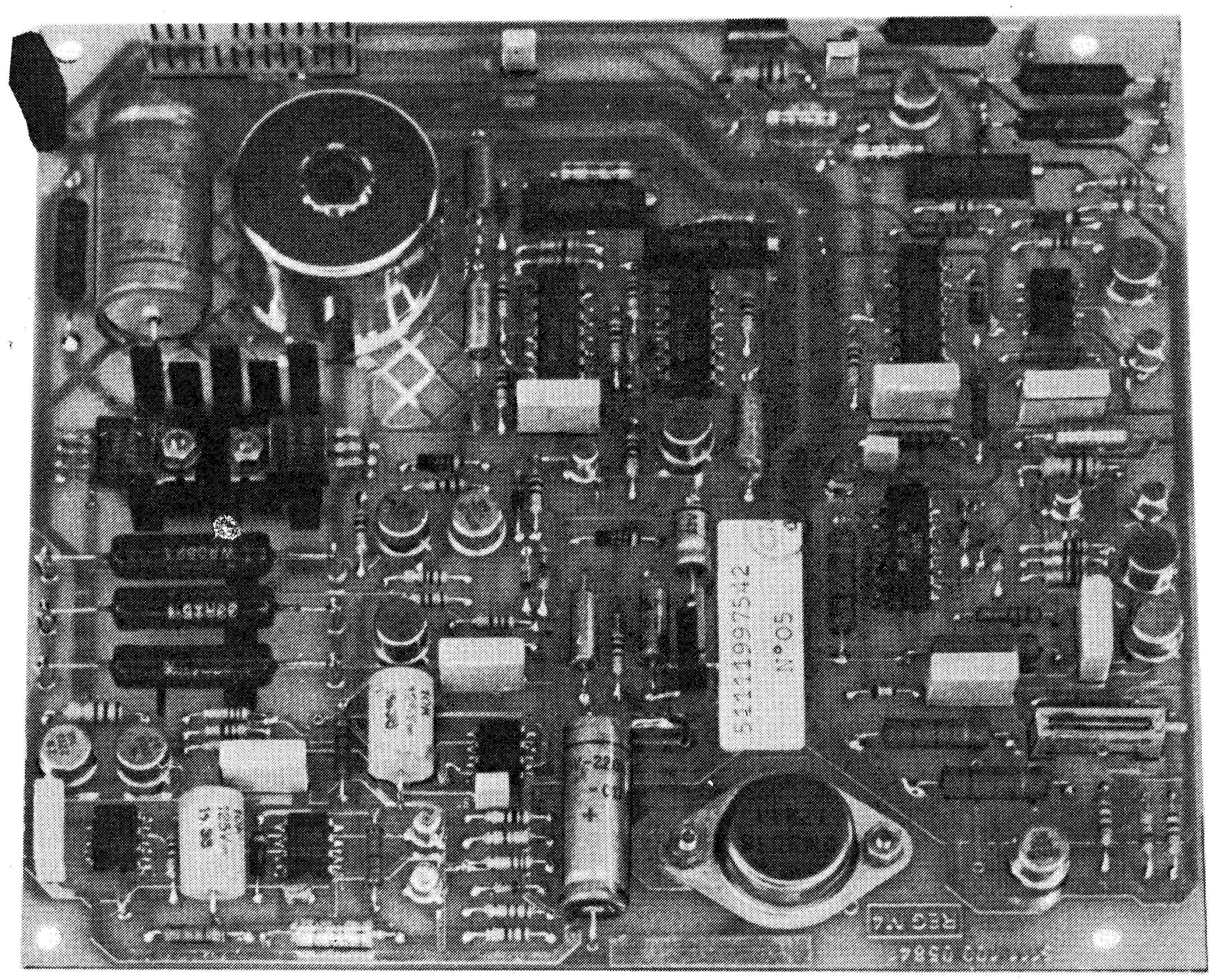
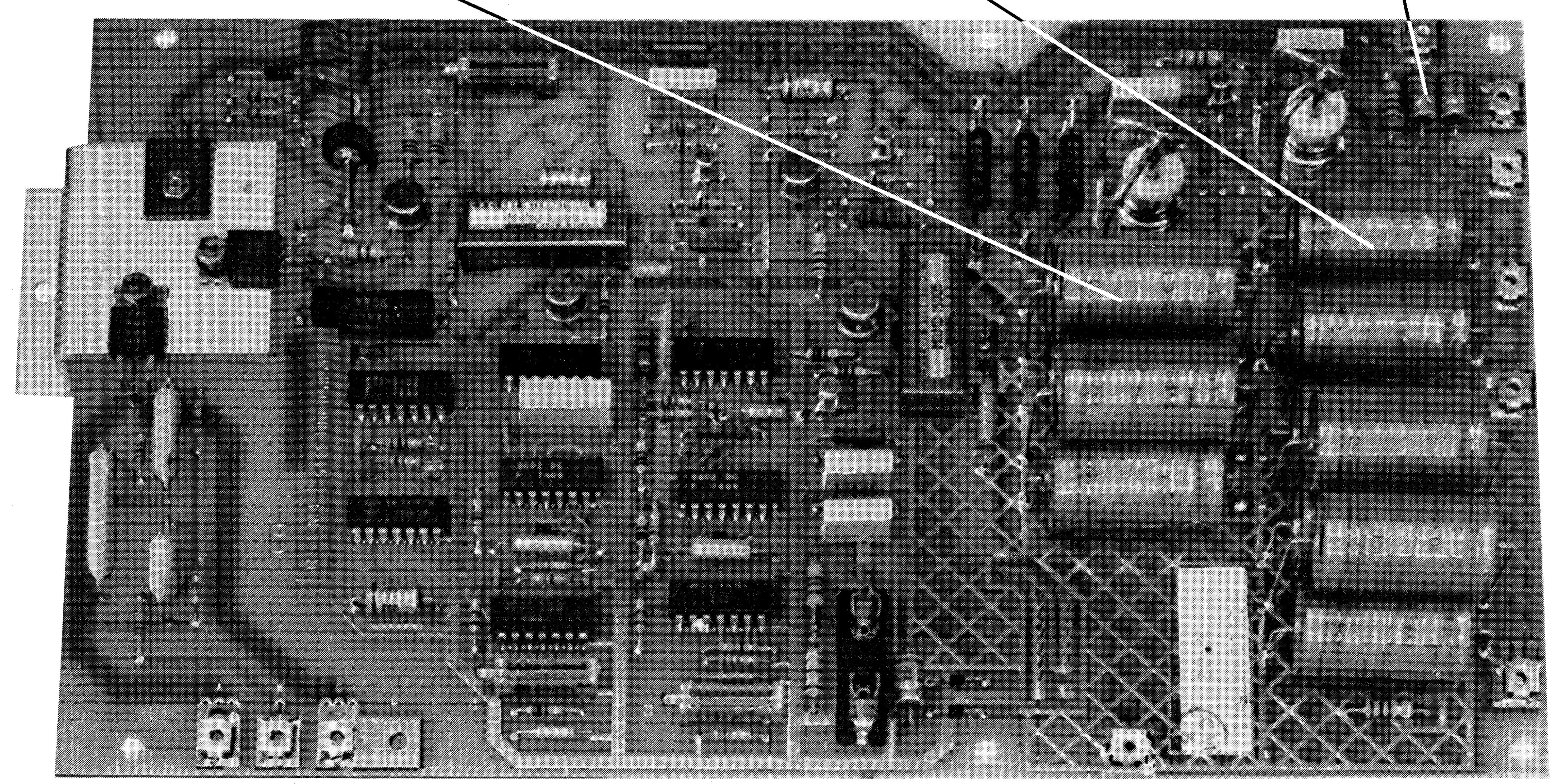


Figure 5-7 Power Supply Assembly Locations



REG CARD



RST CARD

Figure 5-8 Power Supply Circuit Cards

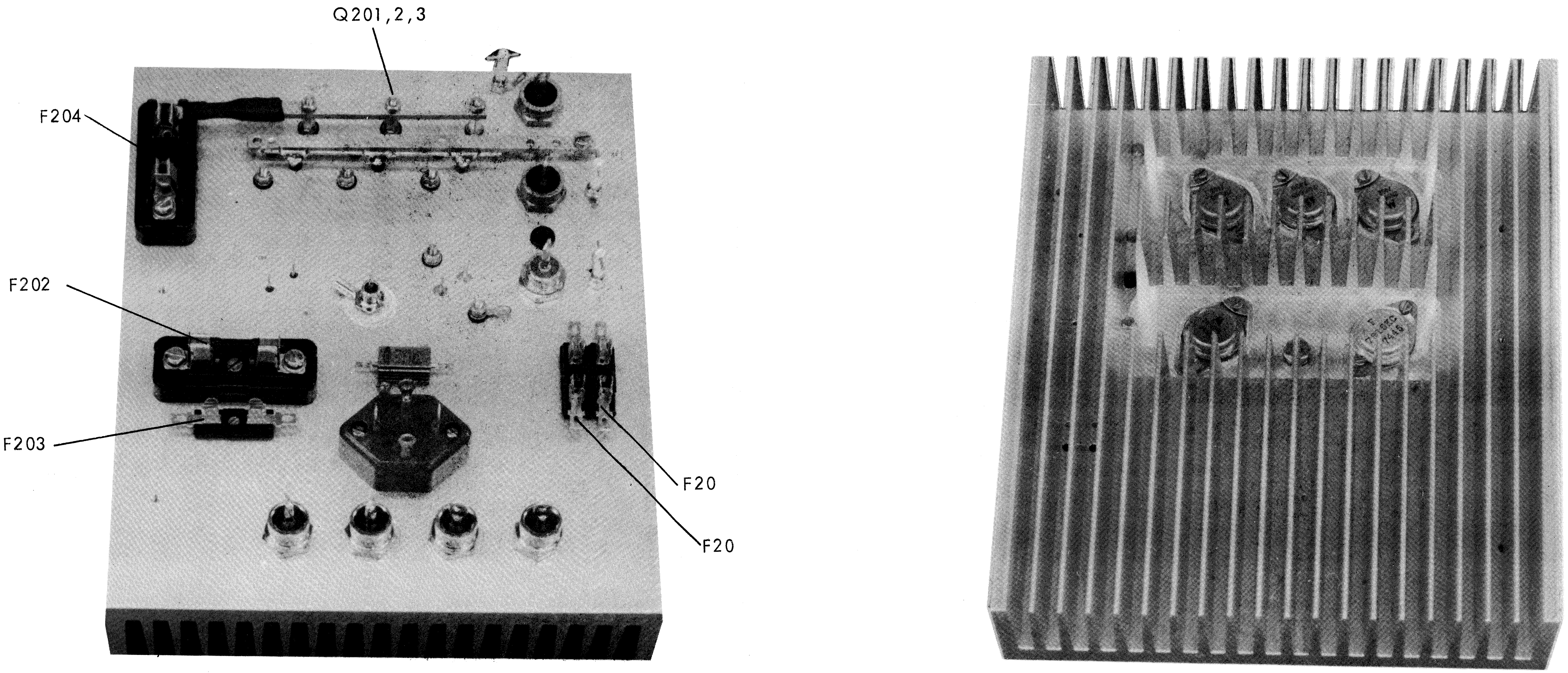


Figure 5-9 Heat-Sink Assembly

5.52 POWER SUPPLY FOR EXTENSION RACK E2

The power supply provides a +5V 2.4% regulated d.c. voltage, +18V and -18V unregulated d.c. voltages, and the Power Failure and Reset Logic signals for the control units in the rack. It also provides the +19V used by the mains detector logic. The power supply is made up of five sub-assemblies. These are:

- Mains Filter and Local/Remote switching
- Mains Transformer
- Power Block
- Sequence Card
- Fan unit to cool the whole cabinet

The mechanical layout of these sub-assemblies is shown in Figure 5-10.

5.53 ELECTRICAL DESCRIPTION

Figure 5-11 shows a block diagram of the Power Supply and Figure 5-15 is the schematic diagram. The following paragraphs describe the function of each block and relate the blocks to the components on the schematic diagram.

5.54 a.c. Input

The a.c. mains is connected to the mains transformer via a mains filter and a Local/Remote switch S201. The value of the input fuse F303 is for 110/115 volts operation type D8TD/6.3A slow blow and for 220/240 volts operation is type D8TD/3.5A slow blow. With the Local/Remote switch in the Local position the live line of the mains is connected to pin 1 of the transformer via the contact of switch S201. With the Local/Remote switch in the Remote position the live line of the mains is connected via the contact of relay K201, which has to be energised by an external +5V supply.

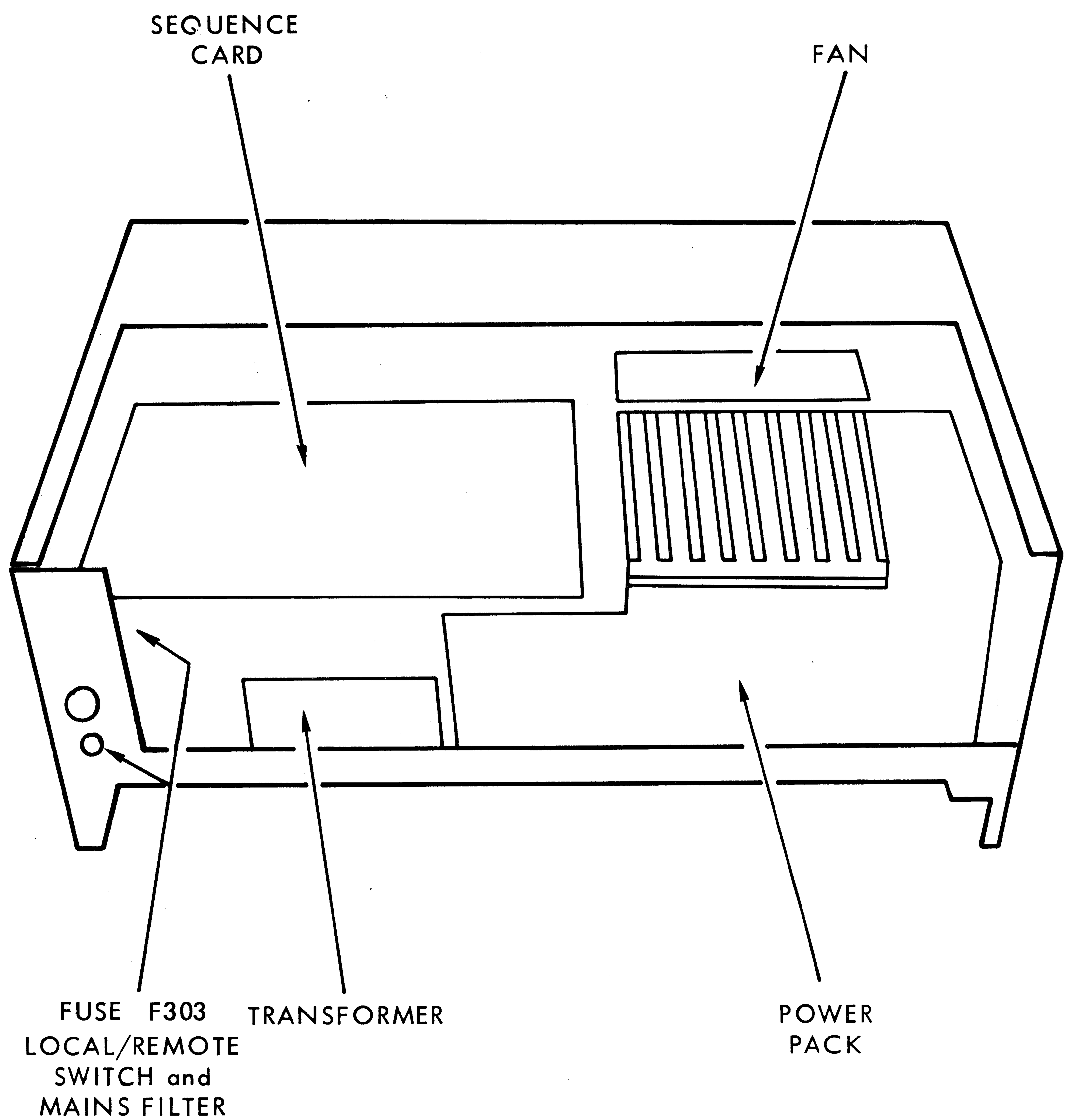


Figure 5-10 Power Supply Sub Assemblies

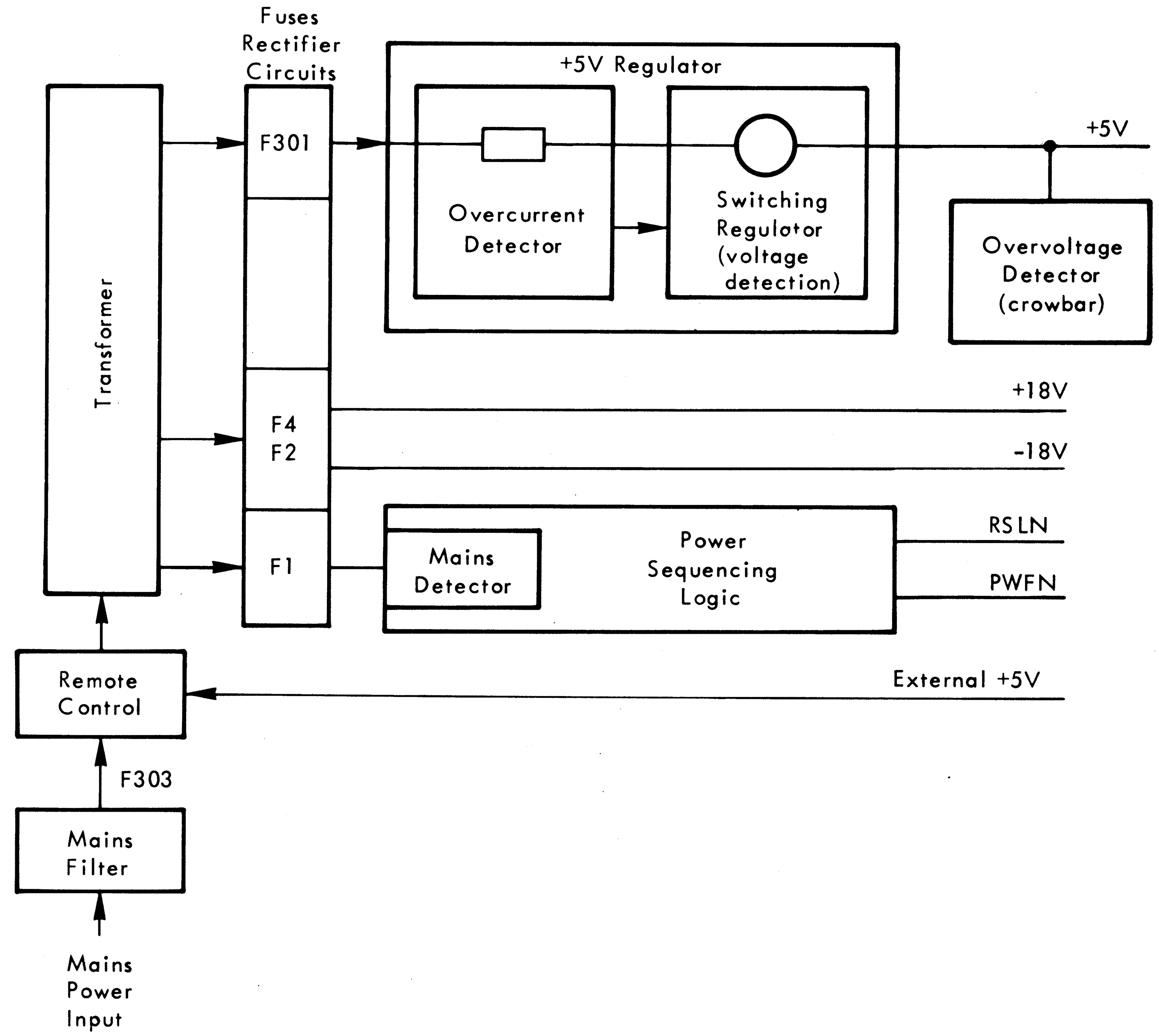


Figure 5-11 Block Diagram

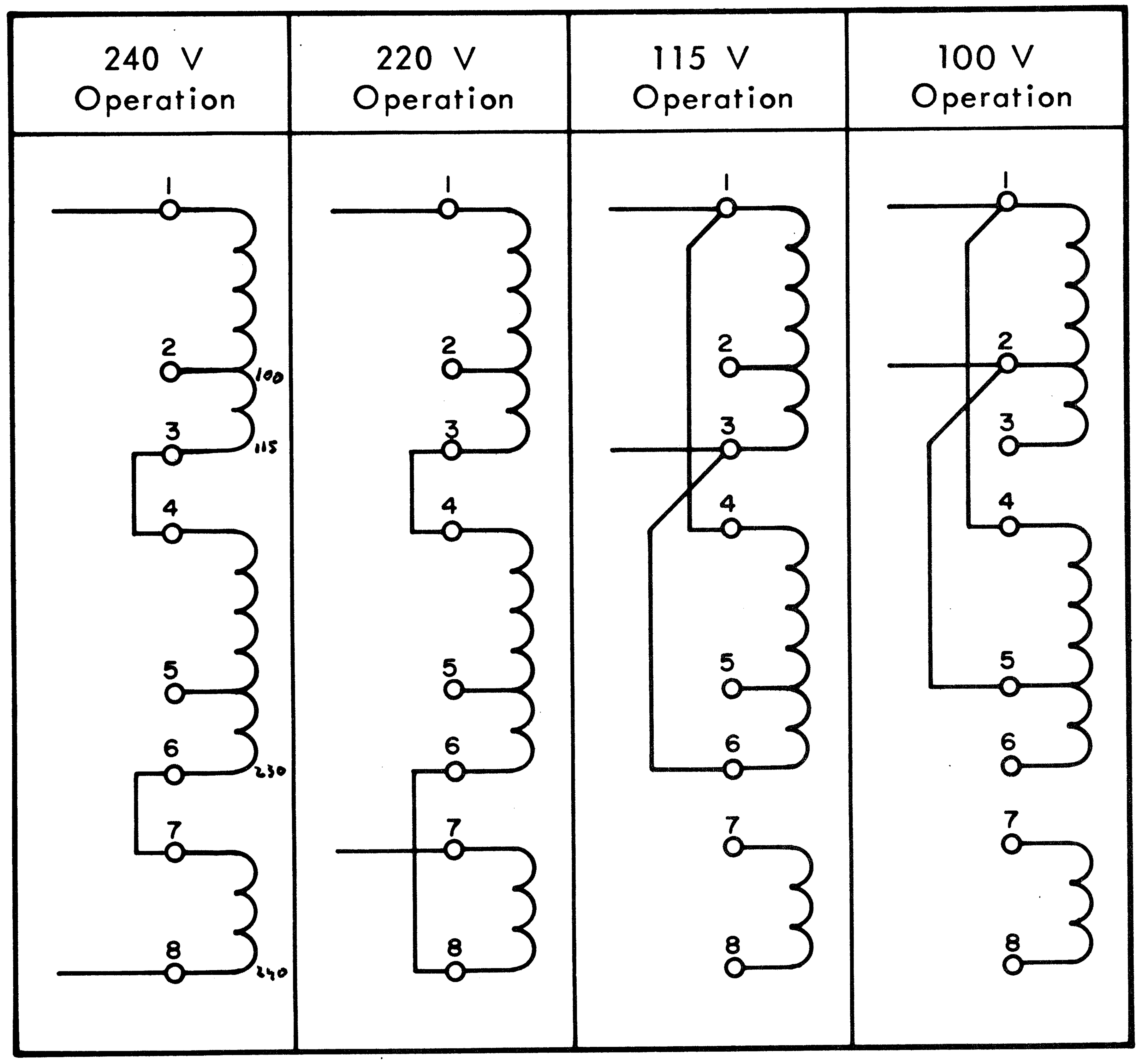


Figure 5-12 Mains Transformer Connections

5.55 Mains Transformer Connections

The mains transformer can be connected to either 240V, 220V, 115V, or 100V mains supply; Figure 5-12 shows the different configurations. Pin 9 of the transformer is the shield and is always connected to ground, and the fan is always connected between pins 1 and 3 (115V nominal).

5.56 Rectifiers and Filters

The output voltages from the three secondary windings of the transformer are rectified and filtered to provide four raw d.c. voltages. Fuses F301, F1, F2 and F4 protect the rectifiers and transformer if overload conditions occur.

5.57 <u>+5V Regulator.</u> The raw d.c. voltage for this circuit is provided by a full wave bridge using CR304-CR307 and is filtered by C302; the output at fuse F301 is approximately 30V d.c.

5.58 <u>+18V and -18V Unregulated Voltages.</u> These two voltages are provided by two center tap full wave rectifier circuits CR10-CR13 and are filtered C304 and C305.

5.59 <u>Mains Detector Voltage.</u> This is provided by a center tap full wave rectifier circuit CR8 and CR9 and is filtered by C5; the output voltage is approximately +19V d.c.

5.60 +5V Regulator Circuit

This is a switching regulator using Q301 and Q302 for the switching controlled by IC7. The switching frequency is 20KHz at full load but the frequency will decrease when the load decreases. The frequency is adjusted by changing the value of R49, and the output voltage is adjusted by potentiometer PR1.

5.61 +5V Overcurrent Protection

The overcurrent detector circuit uses transistors Q11 and Q12 and thyristor CR15. Q11 monitors the current flowing through R301 and when the signal is about 700mV Q11 is switched ON and the thyristor is triggered. Then Q12 is saturated so Q301 and Q302 are cut off and the signal from R65 inhibits IC7. Overcurrent adjustment is by potentiometer PR2 and it is normally set for a value of 20.7A at 25°C ambient temperature. Thermistor R68 is included in the adjustment network to give temperature correction.

5.62 +5V Overvoltage Protection

The overvoltage protection is provided by transistor Q101, zener diode CR105, and thyristor CR101. The transistor and zener are the threshold detector and Q101 is normally OFF. When the voltage increases above the operating value of the zener (in this case between 6V and 7V), Q101 is turned ON and the thyristor is triggered short circuiting the output from the +5V supply.

5.63 Sequence Logic

This logic uses the output from the mains detector circuit to control the Switch ON / Switch OFF sequence logic and to provide the Powerfail (PWF) and Reset (RSL) logic signals.

5.64 Mains Detector. The output from the two diodes CR8 and CR9 drives the transistor Q2 circuit that triggers the two monostable chips IC1 and IC4 if a mains failure longer than 10mS occurs. The 10mS delay time is adjusted by changing the value of R4.

5.65 Switching ON. When the mains present signal (from the mains detector circuit) is high on pin 4 of IC4 the monostable is triggered and after a delay of 300mS (to allow the output voltages to reach their normal operating values) the output from pin 7 triggers the other two monostables that enable the PWF and RSL signals. During this delay the output from pin 6 has triggered IC1 and after a delay of 150mS the output from pin 10 activates the transistors of the relay driver (Q3-Q5), and the relay operates opening the contact and removing the ground connection to the collector of Q6; the state of the RSL signal will still stay at the low level (0) until enabled high by the output from IC4 pin 9. When pin 7 of IC4 goes high the signal from pin 3 of IC3 triggers IC1 pin 4 and it is also used to bypass IC4 (which is only used during Switch OFF) and enable RSL to go high.

After a delay of 400µS the output from pin 7 of IC1 goes high enabling the PWF signal to go high and the supplies are considered operational.

5.66 Switching OFF. When either a mains failure longer than 10mS occurs, or the +5V d.c. disappears, or the power supply is switched OFF the following sequence occurs. The mains present signal goes low and bypasses IC4 pin 4 and IC1 pin 4 to send the PFW signal low. At the same time it is used to activate pin 11 of IC4 which triggers and after a delay of 2mS the output from pin 9 sends the RSL signal low. A feedback signal from the RSL logic (via IC6 pin 12) is sent to the relay driver circuit and the relay is de-energised, the contact closes and the RSL line is grounded.

5.67 Timing

The timing diagram of the sequence logic is shown in Figure 5-13 (the circled numbers refer to points on the schematic diagram), and the timing diagram for the d.c. voltages and the logic signals is shown in Figure 5-14.

5.68 Timing Adjustments

The timing of the sequence logic can be adjusted by changing the values of R and C as follows:

- 300mS — change R24 and C11
- 150mS — change R21 and C13
- 400µS — change R19 and C8
- 2mS — change R28 and C10

5.69 MECHANICAL DETAILS

The position of the extention cabinet (E2) in the 19in rack in relation to the other units will decide if the cabinet has to be withdrawn completely before trying to troubleshoot the power supply. The following description assumes that the cabinet has to be removed from the rack, so if this does not apply to your system the instructions for removal and replacement can be ignored.

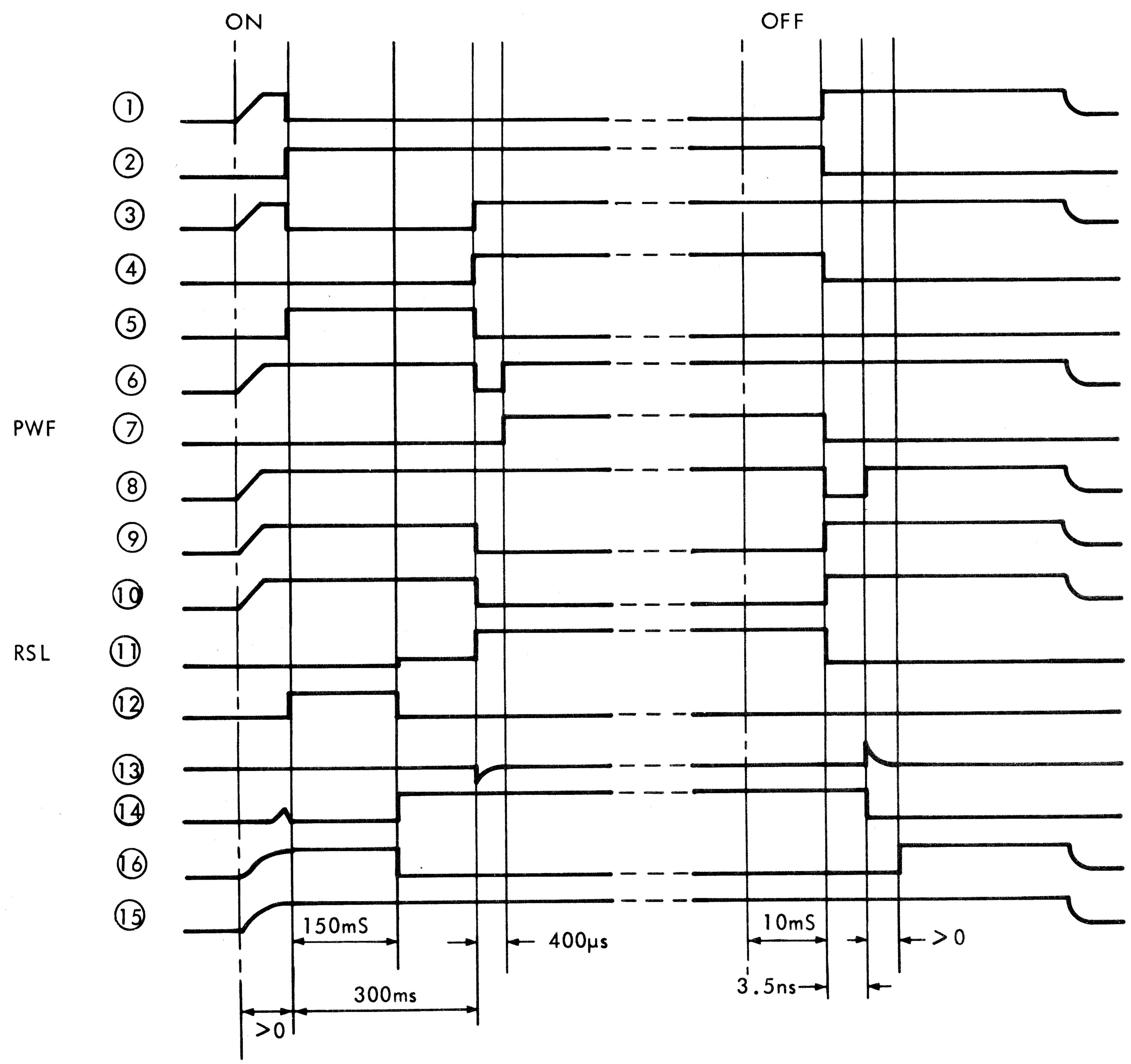


Figure 5-13 Timing Diagram of Sequence Logic

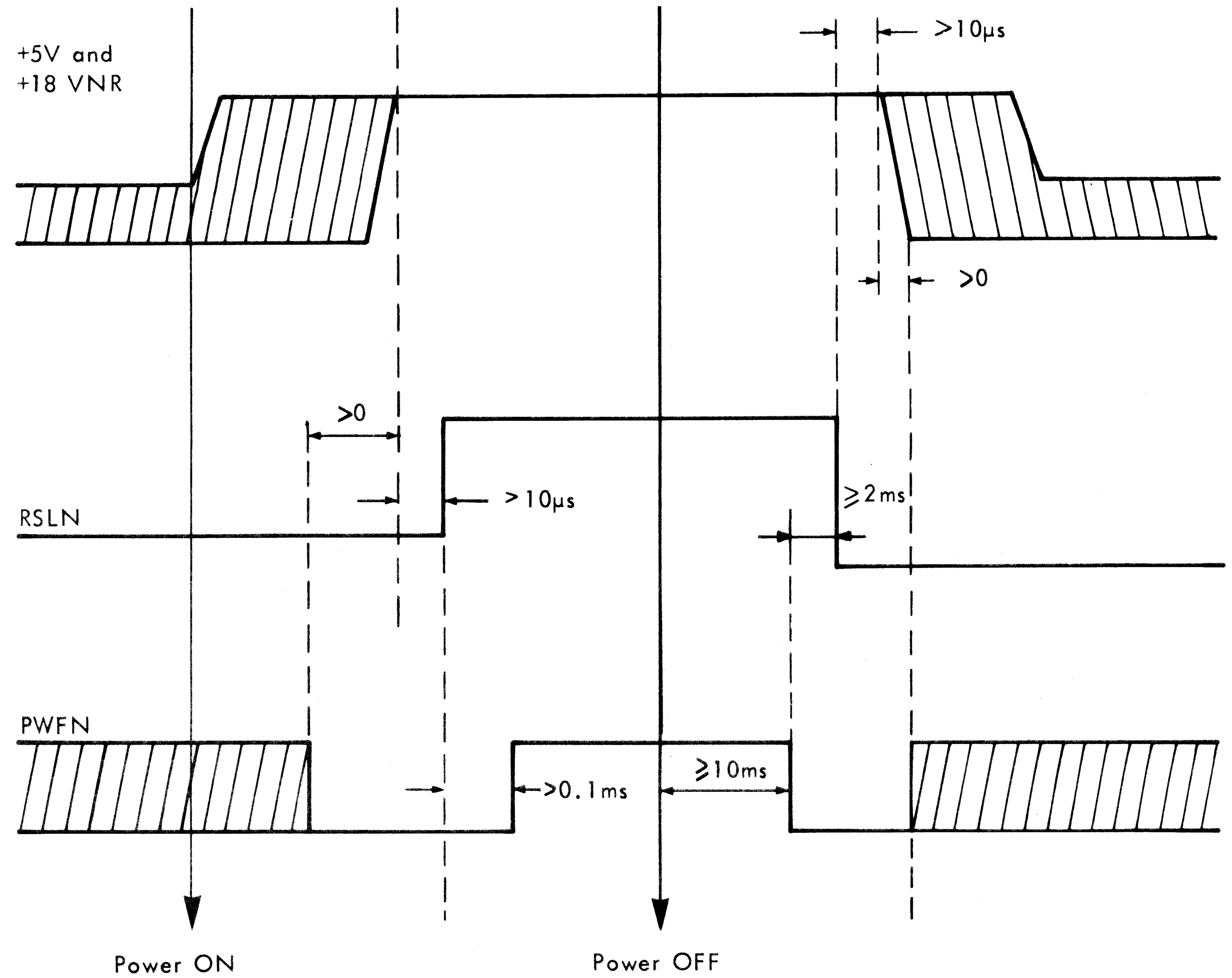


Figure 5-14 Timing Diagram of d.c. Voltages and Logic Signals

5.70 Cabinet Removal and Replacement

Make sure that the cabinet has been disconnected from the mains supply then:

- Remove the protective cover from the power supply by removing the four retaining screws and lifting the cover clear of the cabinet.
- Remove the blank front panel by unscrewing the two Allen screws and lifting clear.
- Remove the cabinet's four retaining screws and pull it towards you until the telescopic slides are fully extended.
- Disconnect the I/O cables.
- Turn the two fixed slide retaining screws (one for each telescopic slide), one half turn in either direction until the fixed slide spigot is free.
- Pull the cabinet towards you until it is clear of the telescopic slides.

The cabinet can be replaced using the above instructions in the reverse order.

5.71 Power Supply Sub-Assemblies

The following paragraphs describe the removal of the five power supply sub-assemblies.

5.72 Sequence Card (REG E2)

Remove the four retaining screws and unplug the card from the connector.

5.73 Power Block

Remove the Sequence card, then remove the card mounting plate by unscrewing the three retaining screws, then:

- Disconnect the input leads from the transformer making a careful note of their positions.
- Disconnect the output leads from the power block making a careful note of their positions.
- Remove the retaining screws (underneath the cabinet) and lift the Power Block clear of the cabinet.

The Power Block can be replaced using the above instructions in the reverse order.

5.74 Mains Transformer

Remove the Sequence card and the card mounting plate, then:

- Disconnect the input and output leads of the transformer making a careful note of their positions.
- Remove the four retaining screws (underneath the cabinet) and lift the transformer clear of the cabinet.

The transformer can be replaced using the above procedure in the reverse order.

5.75 Mains Filter and Local/Remote Switch

Remove the Sequence card and the card mounting plate, then:

- Remove the fuse and Local/Remote switch by unscrewing their locknuts.
- Disconnect the input leads to the transformer making a careful note of their positions.
- Remove the relay mounting plate retaining screws (located on the side of the cabinet).
- Remove the two filter retaining screws and the mains lead clamp, then lift the assembly clear of the cabinet.

The mains filter and Local/Remote switch can be replaced by carrying out the above procedure in the reverse order.

5.76 Fan Unit

Remove the Sequence card and card mounting plate and disconnect the fan leads from the transformer, then:

- Remove the top protective cover (over the control unit cards) and remove the control unit cards.
- Remove the four fan retaining screws and lift the fan clear of the cabinet.

The fan unit is replaced using the above procedure in the reverse order.

5.77 COMPONENTS

Figure5-16shows the component layout of the Power Block, Overvoltage Card, Mains Filter and Local/Remote switch assemblies and Tables 5-2 to 5-4 list the components. Figure5-17 shows the component layout of the Sequence card and Table 5-5 lists the components.

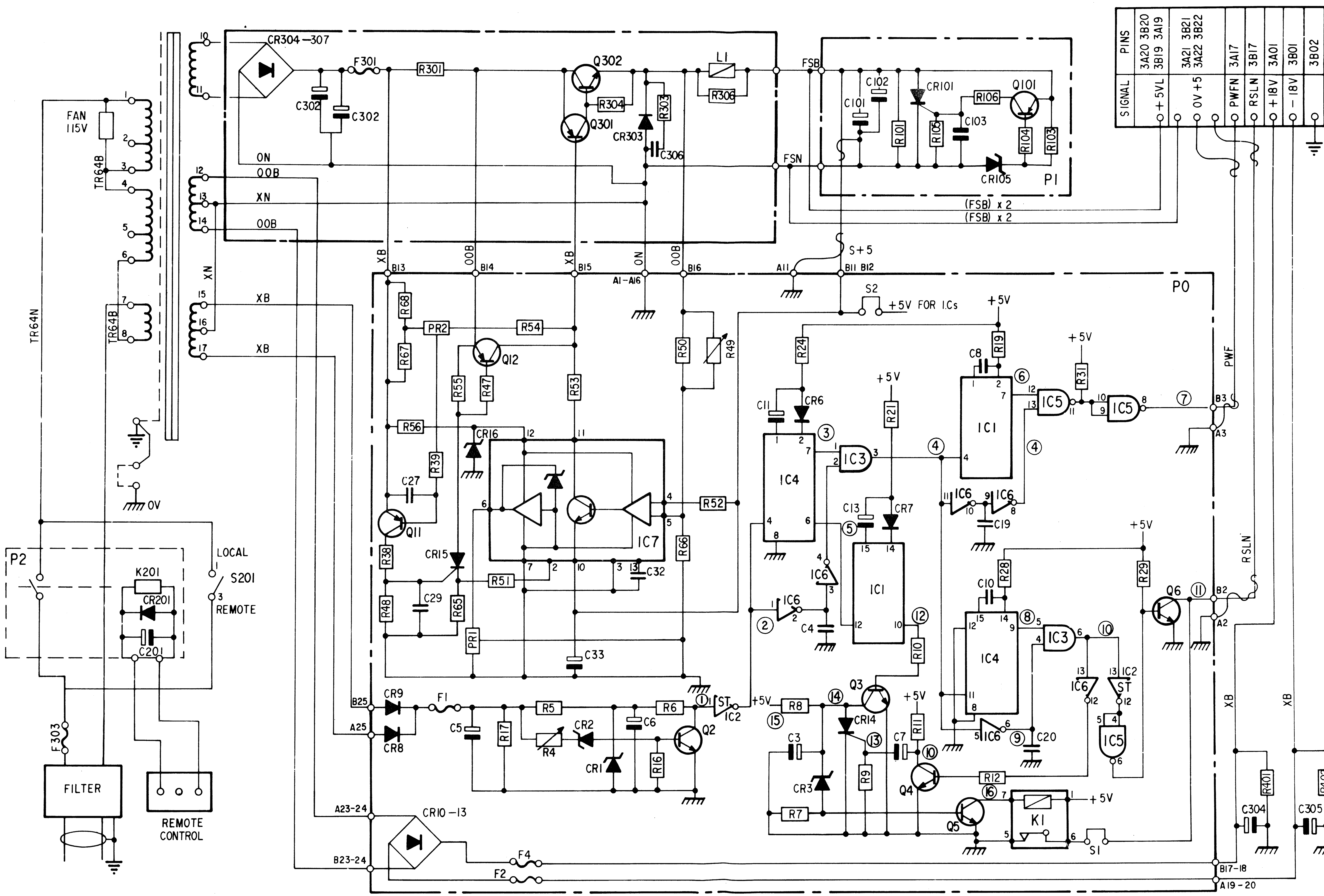
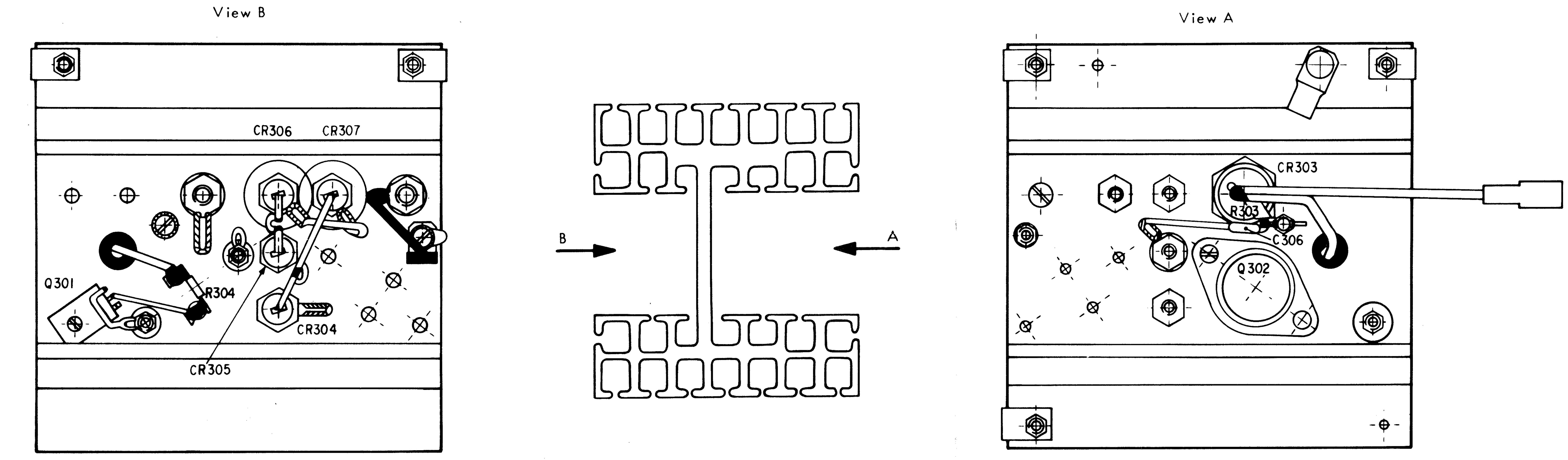
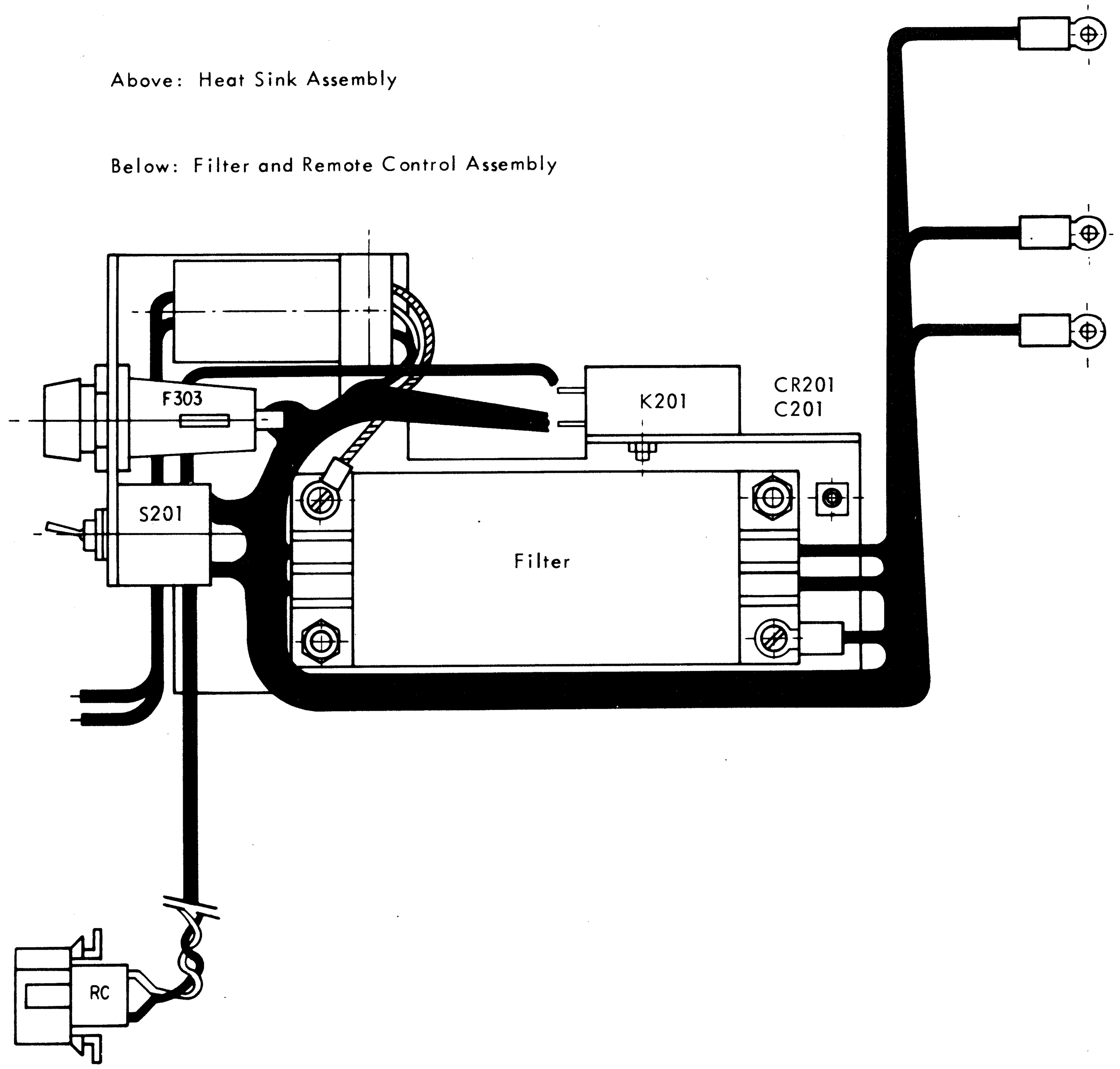


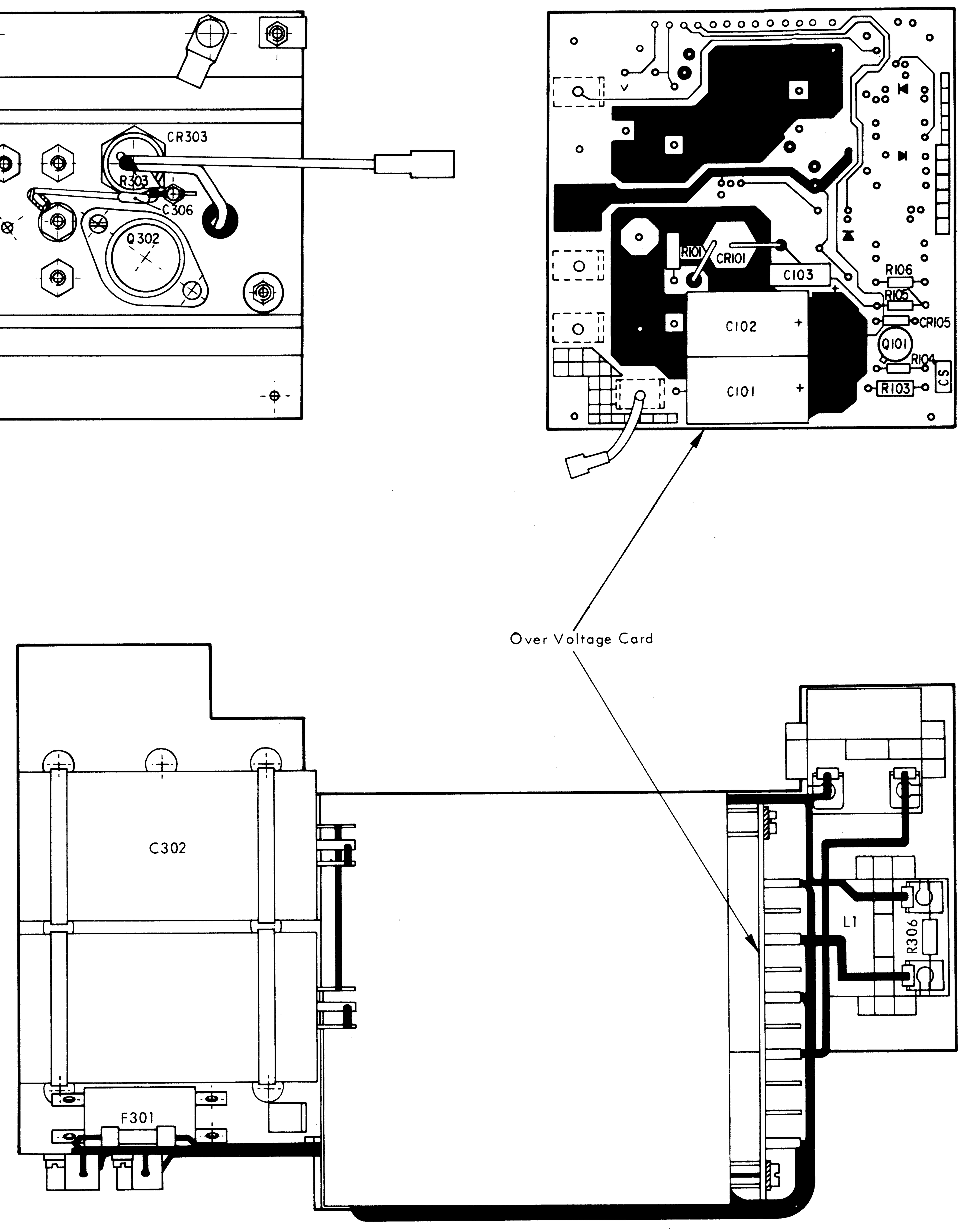
Figure 5-15 Schematic Diagram



Above: Heat Sink Assembly

Below: Filter and Remote Control Assembly





Above: Power Block Assembly

Figure 5-16 Power Supply Component Location

Table 5-2 Power Block Parts List

Reference	Description	12NC Code
	Heat Sink	5111 199 78070
	Harness	5111 199 78060
R304.	Resistor 22Ω, 0.5W, 5%.	
R303.	Resistor 4.7Ω, 0.5W, 5%.	
C306.	Capacitor 0.1μF, 100V, PMA.	

Table 5-3 Overvoltage Card Parts List

Reference	Description	12NC Code
	Printed circuit	5111 100 05404
R103.	Resistor 46.4Ω, 0.25W, 1%.	
R104.	Resistor 100Ω, 0.25W, 5%.	
R101.	Resistor 560Ω, 0.25W, 5%.	
R105.	Resistor 270Ω, 0.25W, 5%.	
R106.	Resistor 10Ω, 0.25W, 5%.	
C101,102.	Capacitor 1500μF, 10V, elect.	
C103.	Capacitor 0.1μF, 100V, 10%, MPR.	
CR105.	Zener Diode BZX79 C5V6.	
CR101.	Thyristor BTW47 — 600RM.	
Q101.	Transistor 2N2906.	

Table 5-4 Filter and Local/Remote Switch Parts List

Reference	Description	12NC Code
	Mains Filter B84 102 C150.	
	Capacitor B81711 A B25.	
K201.	Relay KS-N V23016-A0002-A101.	
S201.	Switch 6AT2.	
CR201.	Diode BAX12.	
C201.	Capacitor 47μF, 10V, Fitco.	

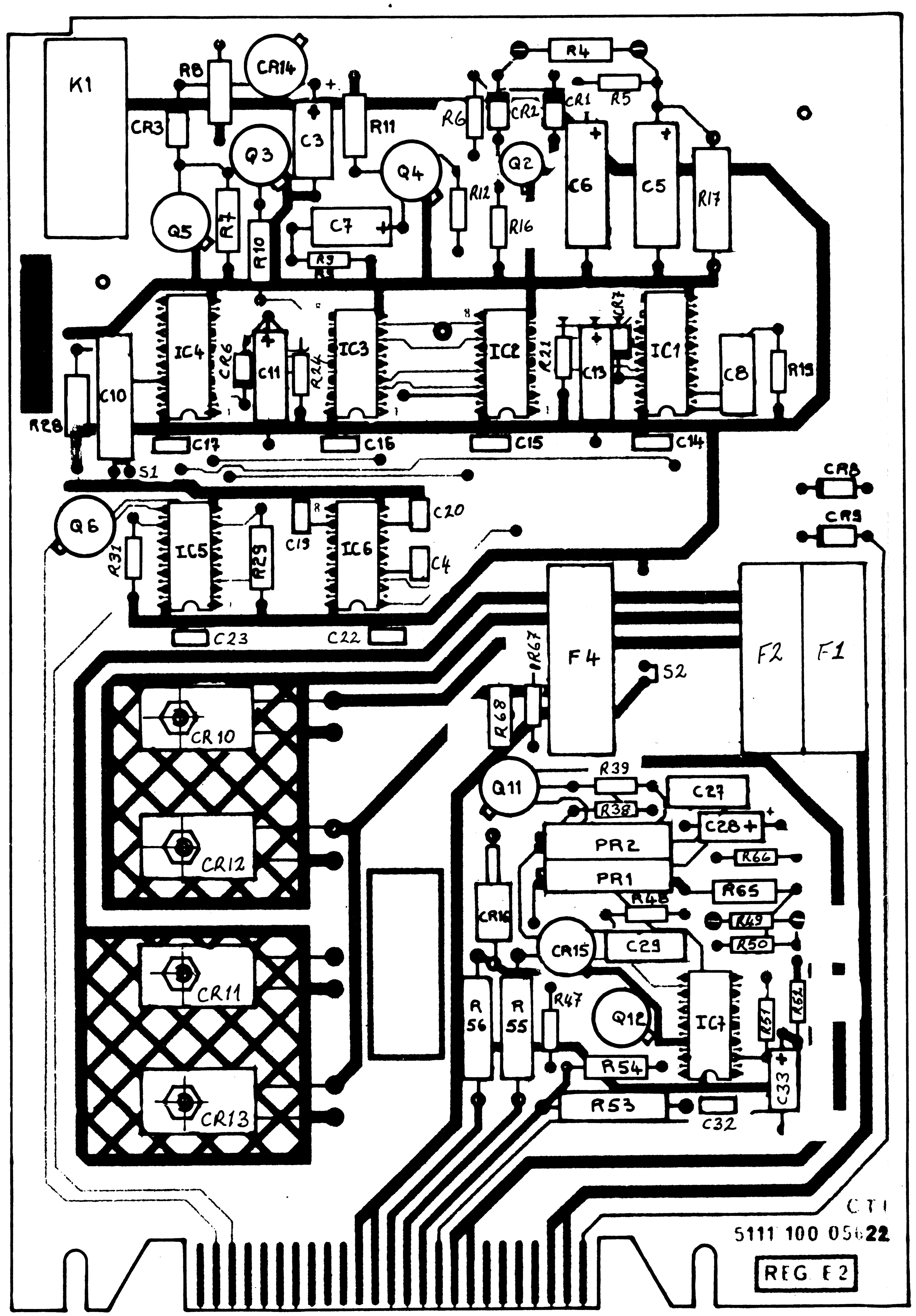


Figure 5-17 Sequence Card Component Layout

Table 5-5 Sequence Card Parts List

Reference	Description	12NC Code
	Printed Circuit	5111 100 05622
IC5.	Integrated circuit 1801.	
IC6.	Integrated circuit 7404.	
IC3.	Integrated circuit 7408.	
IC2.	Integrated circuit 7414.	
IC4, IC1.	Integrated circuit 9602.	
IC7.	Integrated circuit U6A 772 3393.	
Q2.	Transistor BSX20.	
Q6.	Transistor BSX60.	
Q11, Q12.	Transistor 2N2905.	
Q3, Q4, Q5.	Transistor 2N2219.	
CR14.	Transistor 2N1595.	
CR16.	Zener Diode BZV16C15.	
CR1.	Zener Diode BZX79C4V7.	
CR2.	Zener Diode BZX79C7V5.	
CR6, CR7.	Diode BAX13.	
CR8, CR9.	Diode BAX12.	
CR3.	Diode 1N746A.	
CR10, CR11.	Diode BYX49 - 300.	
CR12, CR13.	Diode BYX49 - 300R.	
K1.	Relay MRMD 15005.	
PR1.	Potentiometer 2600 P102 1000.	
PR2.	Potentiometer 2600 P101 100.	
CR15.	Thyristor 2N2323.	
F2, 4.	Fuse D1TD/3.15.	
F1.	Fuse D1TD/0.31.	
R65.	Resistor 100Ω, 0.25W, 1%.	
R67.	Resistor 22Ω, 0.25W, 5%.	
R9, 12, 31, 48, 51, 52, 66.	Resistor 1KΩ, 0.25W, 5%.	
R47.	Resistor 8.2KΩ, 0.25W, 5%.	
R16.	Resistor 820Ω, 0.25W, 5%.	
R24.	Resistor 30KΩ, 0.25W, 5%.	
R6.	Resistor 2.7KΩ, 0.25W, 5%.	
R39.	Resistor 100Ω, 0.25W, 5%.	
R28.	Resistor 19.6KΩ, 0.25W, 1%.	
R5.	Resistor 2KΩ, 0.25W, 5%.	
R21.	Resistor 24KΩ, 0.25W, 5%.	
R19.	Resistor 12KΩ, 0.25W, 5%.	
R54.	Resistor 150Ω, 0.5W, 5%.	
R7, 10, 11.	Resistor 300Ω, 0.5W, 5%.	
R8, 29.	Resistor 100Ω, 0.5W, 5%.	
R53.	Resistor 390Ω, 1W, 5%.	
R55.	Resistor 2.2KΩ, 1W, 5%.	

Table 5-5 contd.

Reference	Description	12NC Code
R17.	Resistor 1KΩ, 1W, 5%.	
R50.	Resistor 4.7MΩ, 0.25W, 10%.	
R56.	Resistor 820Ω, 1W, 5%.	
R38.	Resistor 1KΩ, 0.25W, 5%.	
R49.	Adjustable Resistor 470KΩ-3.3MΩ, 0.25W.	
R4.	Adjustable Resistor 1KΩ-10KΩ, 0.25W, 1%.	
R68	Thermistor CTN B 832001/P50E.	
C8,29.	Capacitor 0.1μF, 100V, 10%, MPR.	
C27.	Capacitor 0.01μF, 250V, 10%, MPR.	
C10.	Capacitor 0.47μF, 100V, 10%, MPR.	
C32.	Capacitor 100pF, 63V, 2%, ceramic.	
C4,19,20.	Capacitor 470pF, 100V, 10%, ceramic.	
C7.	Capacitor 1μF, 63V, Fitco.	
C3.	Capacitor 47μF, 10V, Fitco.	
C5.	Capacitor 57μF, 25V, Fitco.	
C6.	Capacitor 100μF, 10V, Fitco.	
C28,33.	Capacitor 4.7μF, 10V, CTS13.	
C13.	Capacitor 22μF, 16V, CTS13.	
C14-17,22,23.	Capacitor 3900pF, 100V, 10%.	
C11.	Capacitor 33μF, 10V, CTS13.	

SECTION VI

V24 SERIAL CONTROL UNIT

6.1 GENERAL

The V24 Serial Control Unit (V24-CU), located on the CPU card, interfaces an Operator's device with the CPU. The V24-CU converts between eight-bit parallel CPU characters (at +5/0V) for the CPU and bit-serial data (at +12/-12V) for the device. The V24-CU can provide parity bit insertion and checking; this feature is plug-programmable by means of U-links on the CPU/CU card. Operating speed is also plug-programmable, with the possibilities :

110 baud, for ASRV24
600 baud, for PER 3100
1200 baud
2400 baud
4800 baud
9600 baud } for display

The V24-CU uses direct interface connections with the CPU for address, command, and data to minify GP-Bus utilization. The address and commands are received directly from the CPU K-register. CU/CPU data is received directly from the least-significant half of the L-register (L8-15) and sent directly to the CPU via the C-selector.

6.2 Channel and Interrupts

The CU operates via the system programmed channel, under direct control of the CPU. The CU sends an interrupt (INTSERN) to the CPU to request each character transfer and to indicate when it is in Wait Status mode.

6.3 Serial Data Format

The bit-serial data format for a single character is shown in Figure 6-1. The eighth "Data"bit may be either an actual CPU data bit or it may be a parity bit supplied and checked by the V24-CU. The parity bit, if used, can be either even or odd. Either one or two stop bits may be selected (by a plug on the card) to conform to the device requirements.

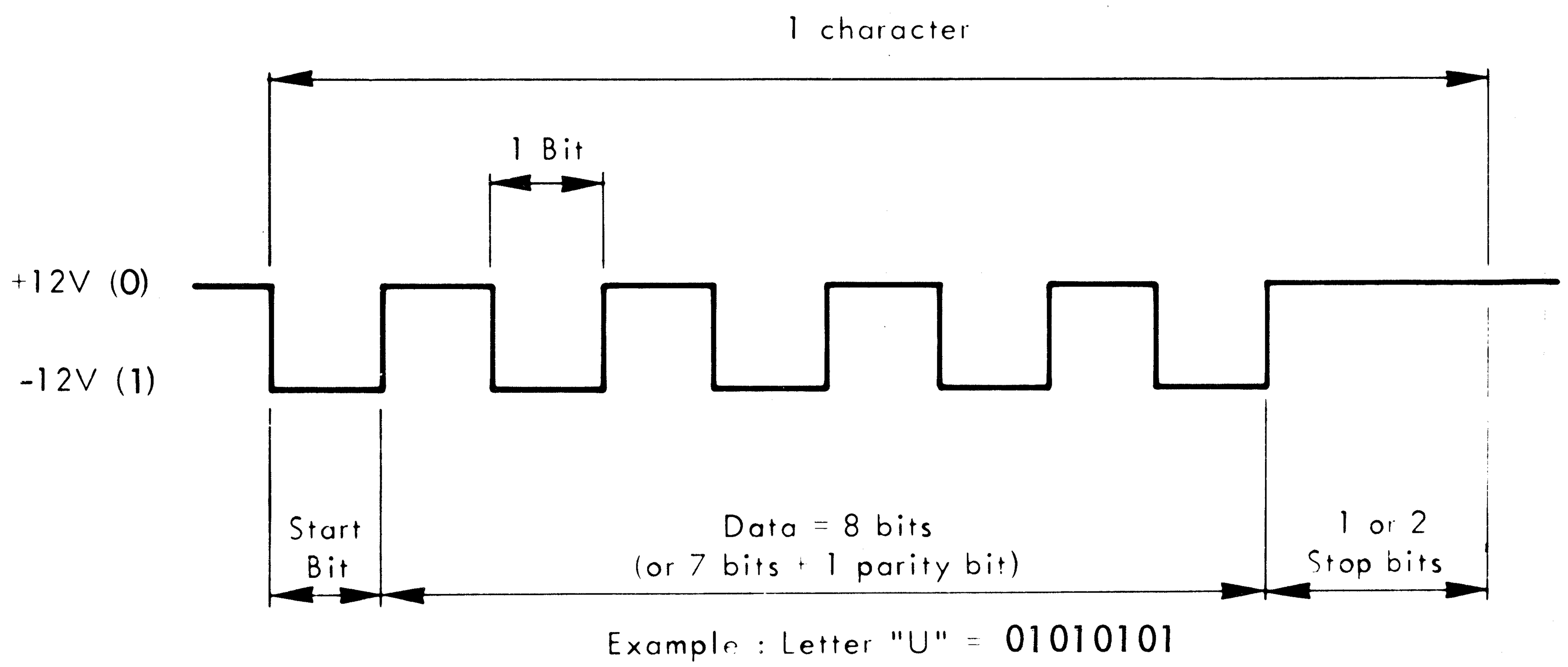


Figure 6-1 Serial Data Format

A logic One is represented by a low level (-12V) and a logic Zero is represented by a high level (+12V). When no character is being exchanged, the line is High. Characters may be transferred end-to-end or may be separated by any amount of gap.

6.4 CPU INTERFACE AND CONTROL

The V24-CU operations are controlled by CPU I/O instructions (CIO Start, CIO Halt, OTR, INR, SST). These instructions are format-0, type T8. The instruction word contains address and control information which is sent directly to the CU on the K lines 4, 8-15, as follows :

K-Register

Bit :	0	1 – 4	5 – 7	8 9	10 – 15
	0	OPC	R3		DA
CIO		1000		11 = start I/O	
				10 = stop I/O	
OTR		1000		0F	F may be used by device to specify a function, such as binary or ASCII
INR		1001		0F	
SST		1001		11	

To control logic   To Address-Recognition Circuit

Additional control is provided directly from the CPU D-register. Bit D15 indicates input (D15 high) or output (D15 low) operations. This bit controls the setting of the CU flip-flop IN/OUT during a CIO Start operation. Bit D10 is set during a CIO Start operation to indicate Echo mode (routing the device input data directly back to the device during input data transfers).

6.5 The CU detects its address from K10-15 when it receives CPU timing signal TMPR from the CPU. The CU recognizes its address, generates internal Address-Recognized signal TYARE/N, and sends AREDELA back to the CPU. The address code is determined by the addressing U-links. If the command is accepted, the CU sends signal TYAC back to the CPU; if the command is not accepted, TYAC is not sent and no other action is performed by the CU.

6.6 OPERATION

The V24-CU operates in six different states, as follows :

Operational State	F0/F1	Function
Inactive	0 0	The CU is inactive.
Execute-In	0 1	The CU is waiting for or receiving a character bit-serial from the device.
Exchange-In	1 1	The CU is waiting for or performing an INR command (Character to CPU).
Execute-Out	0 1	The CU is transmitting one character bit-serial to the device, or is waiting for a free line.
Exchange-Out	1 1	The CU is waiting for or performing an OTR command (Character from CPU).
Wait Status	1 0	The CU is waiting for or executing an SST command.

The operating flow is shown in Figure 6-2. The CU is set to the Inactive state by master clear (MCL/RSL), including at system power-on time. When the CU receives a CIO Start command (input or output), it goes to the Execute State if the device is operable; if the device is inoperable, the CU switches to Wait Status state.

6.7 In input mode, the CU goes to Execute-Input state and receives or waits for a character from the device. When the character has been received, the CU switches to Exchange-Input mode and transfers the character to the CPU, with an INR command.

6.8 In output mode, the CU switches from Execute to Exchange state as soon as the device is ready. The CU waits for, or receives, an OTR command with one character from the CPU. The CU switches to Execute-Output state to transfer the character bit-serial to the device, and switches back to Exchange state.

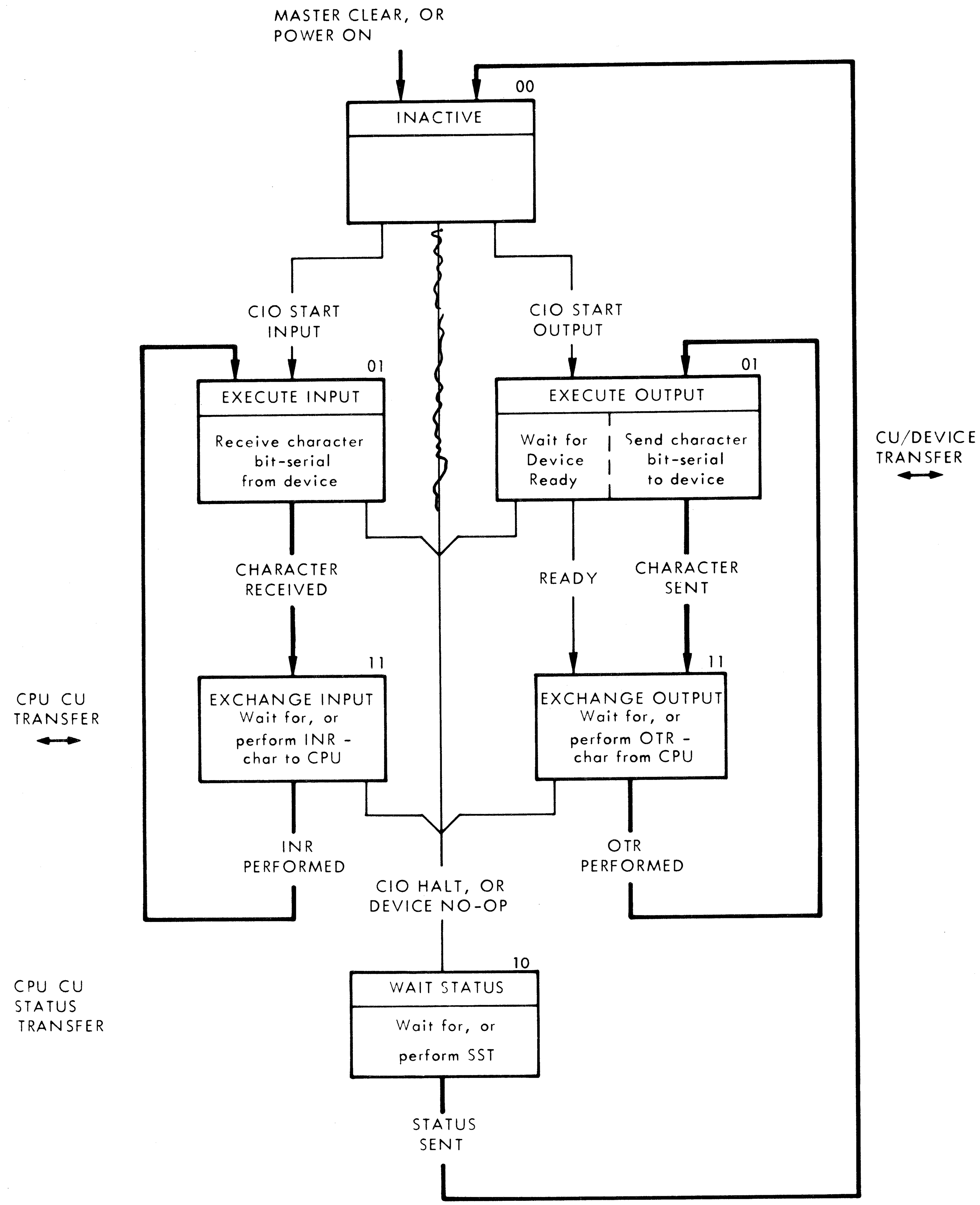


Figure 6-2 V24-CU Operational States

6.9 When the CU receives a CIO Halt command from the CPU, it goes to Wait Status state. The switch to Wait Status is not delayed if CIO Halt is received while the CU is exchanging a character with the device, because the exchange is performed independently by the UART chip once the CU has initiated the operation. If a new exchange is requested, however, it must wait for the end of the current transmission, when FTEOC is reset by the TEOC signal from the UART chip.

6.10 LOGIC DESCRIPTION

A block diagram of the V24-CU is shown in Figure 6-3. Detailed logic is provided in Figure 6-4.

6.11 Data Path

Output data is 8-bit parallel from the CPU L-register direct to the integrated receiver/transmitter circuit UART in the CU. The UART circuit outputs the character bit-serial (with the appropriate start, stop, and parity bits added) to the device on the CT104 line. Input data is received bit-serial from the device via the CT103 line. The input character is assembled by the UART circuit (with start, stop, and parity bits removed) and sent 8-bit parallel to the CPU C-selector.

6.12 The input data path to the CPU is via a type 74157 multiplexer circuit. This circuit is used to place status bits on the data lines during an SST command. Voltage conversion, from +5/0V logic to the +12/-12V used by the device, is performed by the type 1488 output inverters and the 1489A input inverters.

6.13 Character Conversion

Character conversion between bit-serial and bit-parallel is performed directly by receiver/transmitter circuit UART. During output operations, the UART circuit adds the start bit (logical zero) and stop bits (logical 1) to each character. Either one or two stop bits are added, depending on the U-link selection for the NSB input of the UART. (Two stop bits are normal, but a device may require just

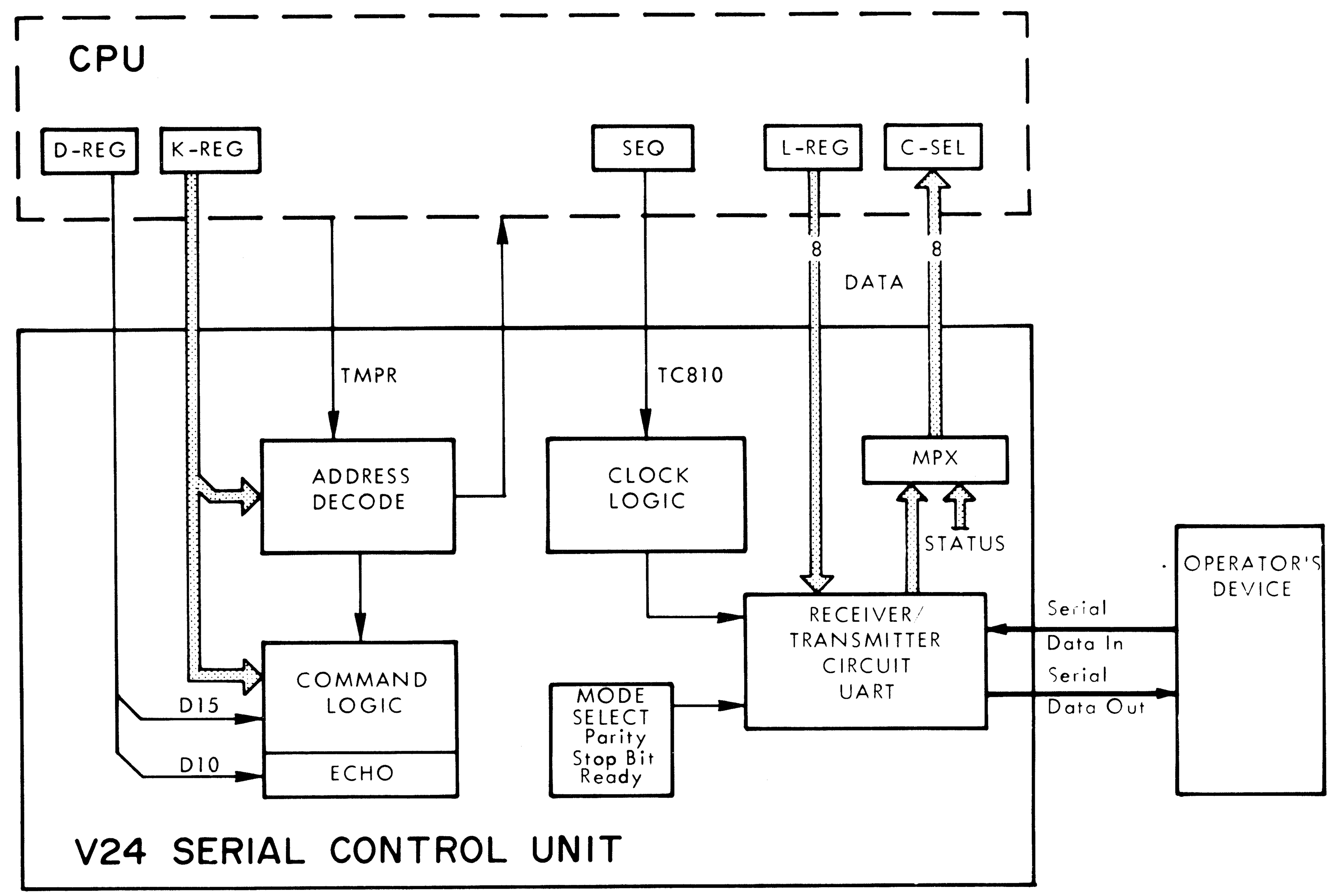


Figure 6-3 V24 Serial CU Block Diagram

one.) During input operations, the UART circuit strips off the same start and stop bits that it is set to add during output operations.

6.14 Transmitting speed is controlled by CU clock logic which is synchronized with the CPU clock. The basic clock signal is TC810 from the CPU. This clock rate is divided by type 74161 counters and flip-flops to provide clock rates for 110, 600, 1200, 2400, 4800, and 9600 baud. The desired rate is selected by positioning a U-link jumper. The selected clock signal drives the UART circuit, via flip-flop CLV24, through the TCP and RCP inputs.

6.15 Parity

The UART circuit can be manually programmed (by means of circuit U-links) to operate without parity, or to insert and check either even or odd parity. With no-parity selected (high level to the UART inputs NPB and ND1), all eight data bits to and from the CPU are sent and received as data bits on the device data lines. If the U-link is set to select parity operation (NPB/ND1 low), a second U-link at the POE input determines whether even or odd parity is to be used.

6.16 Whenever parity is selected, the UART circuit ignores the least-significant bit to and from the CPU (bit 15). For output operations, the UART circuit determines the parity of the seven data bits used and places the resulting parity bit in place of the eighth data bit with the serial character to the device. For input operations, the UART circuit checks the parity of the first seven data bits received, and compares the result with the eighth (parity) bit. The UART circuit indicates an error by activating the PE signal. A parity error sets the CHIPER and FHALT flip-flops and switches the CU to Wait Status state.

6.17 Device Interface

All CU-Device signals are via connector 1, as follows :

Conn. 1	Line	Signal
1A30	CT101	Mechanical Ground, to/from GP Bus connector 3
1B31 to 1B37	CT102	Ground
1A27 1B27	--	Signal Ground
1A28 1B28	--	+5V
1A31	CT103	Data In, serial data from device to CU.
1A32	CT104	Data Out, serial data from CU to device.
1A33 1A34 1A36	CT106 CT107 CT109	High = CPU switched on. High when CT1082 is high. High = CPU switched on.
1A35	CT1082	Operable, high when device is connected and switched on.
1A37	CT133	DREADY, Device ready to transfer next output character. If the device does not provide this signal, a U-link on the CU must be positioned to hold this signal active within the CU.

6.18 Echo Mode

Echo Mode is programmable by bit 10 in the R3 register associated with the CIO Start; this is indicated to the CU during CIO Start by D10 to the FECHO flip-flop. Each accepted INR command is memorized by the FACINR flip-flop. At T7 of the CPU ROM address 159, the active MUQ1 sets the CU flip-flop ECHO, and TDSN is validated up to T5 of the next cycle. The low level to the TDN input of the UART circuit causes the input data to be channelled back to the Data Out line to the device. The FACINR flip-flop remains set until the Wait Status state when it is reset by FNU from the CPU.

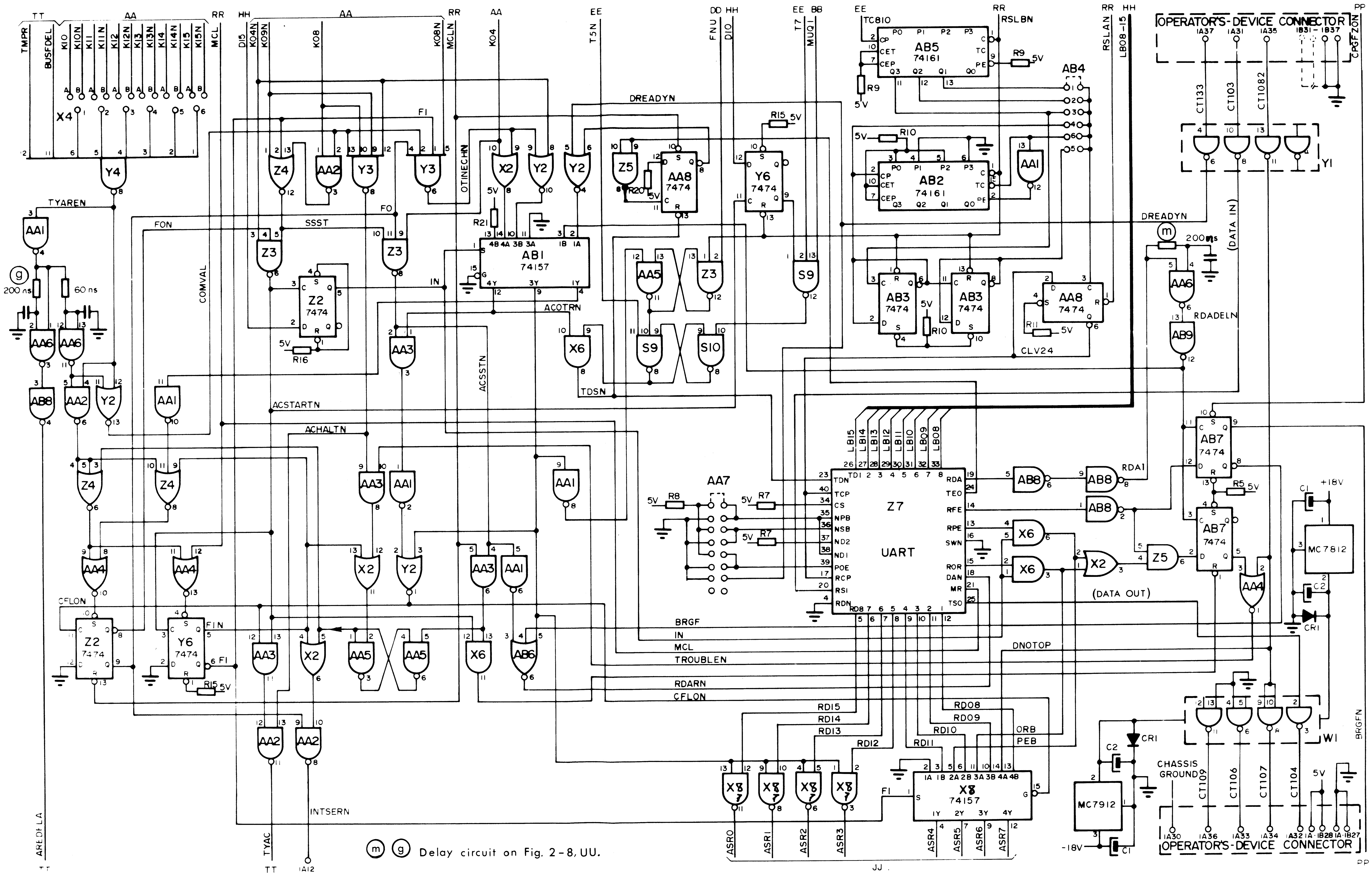
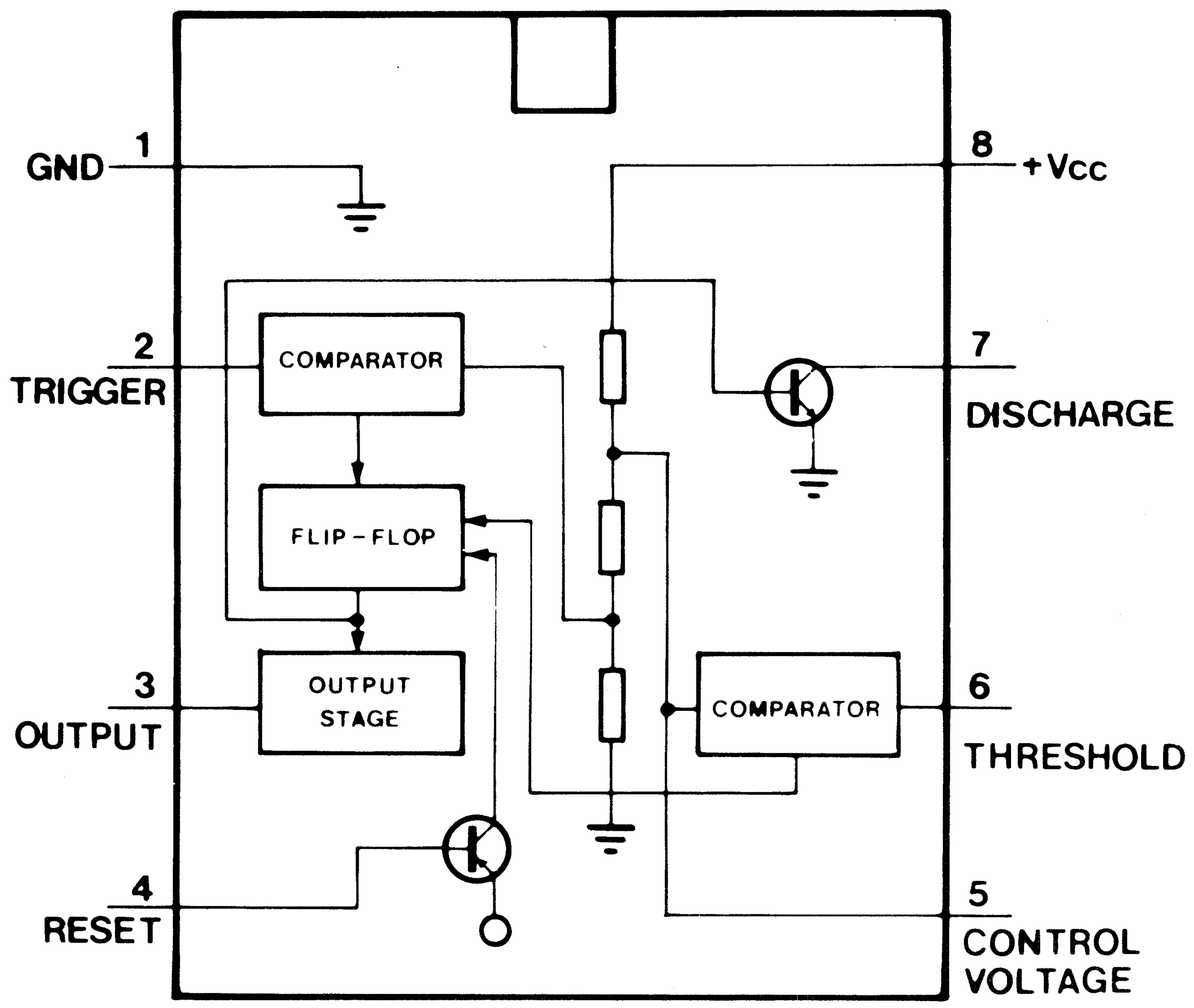


Figure 6-4 V24 Serial Control Unit Detailed Logic Diagram

NE555V

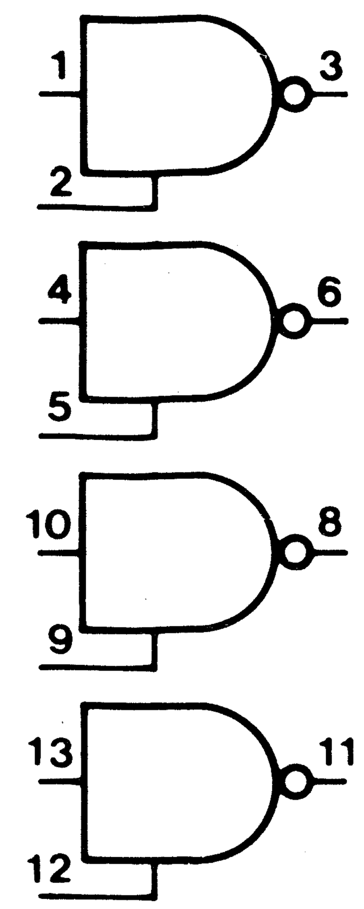
TIMER



1489

1489A

LINE RECEIVER



Response control gives:

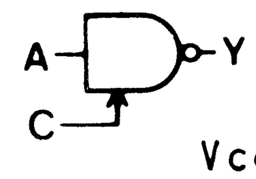
a) Logic threshold shifting

b) Input noise filtering

POSITIVE LOGIC:

Y=$\overline{A}$

C = response comtrol

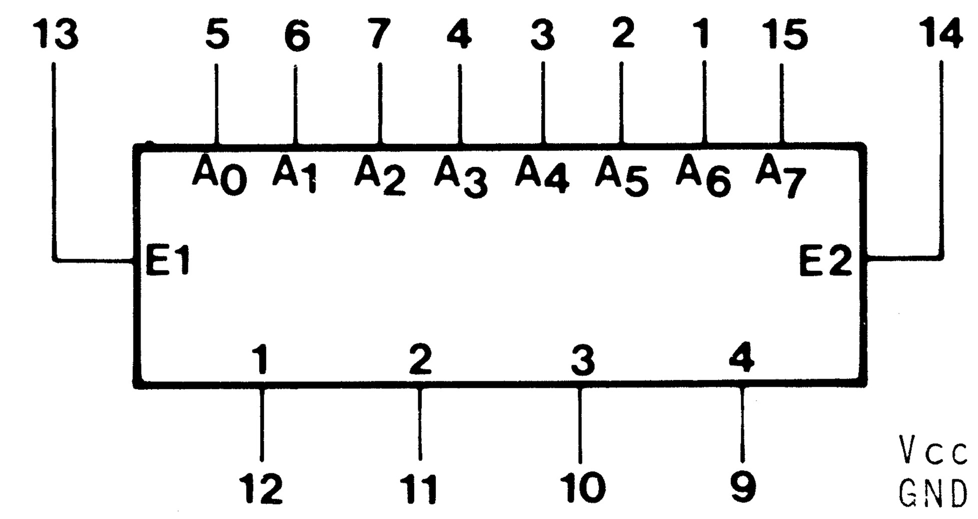


Vcc 14
GND 7

6200

1024-BIT ROM (256x4) O.C.

ENABLE ADDRESS ENABLE



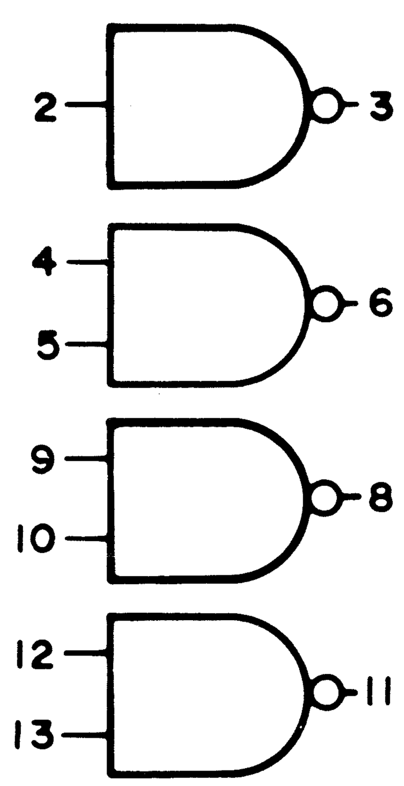
Vcc 16
GND 8

DATA OUTPUT

1488

1488

QUAD MDTL LINE DRIVER



7= 0V
14=V+=+15V max
1=V-=-15V max

1801

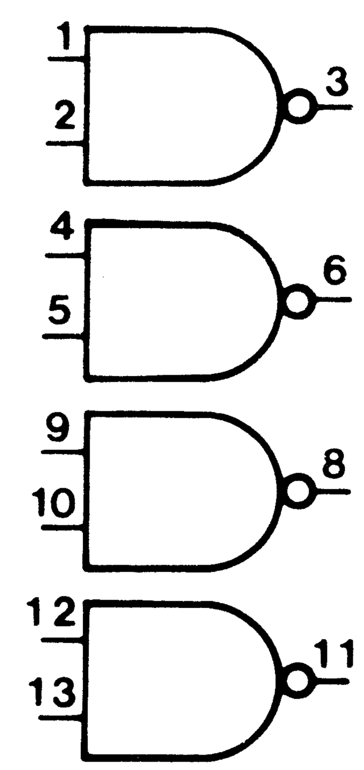
(see 7400)

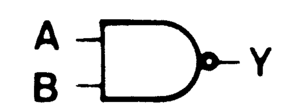
7400

7400
S00
H00
03[1]
37[2]
38
1801[1]

1 open collector

2 buffer



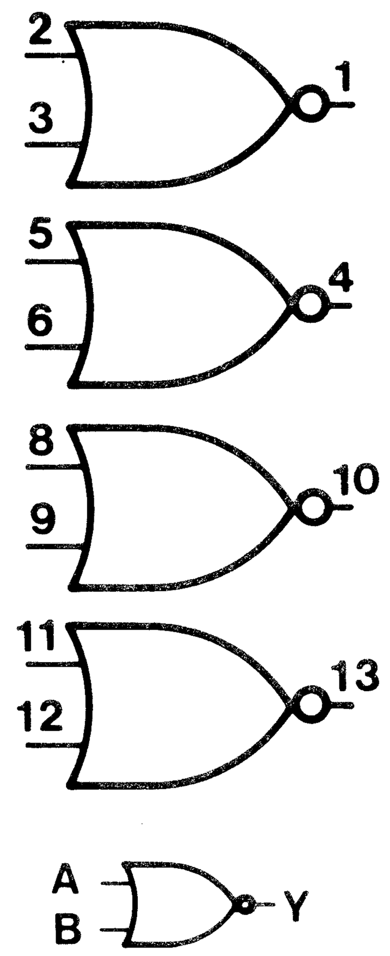


Y=$\overline{AB}$

Vcc 14
GND 7

7402

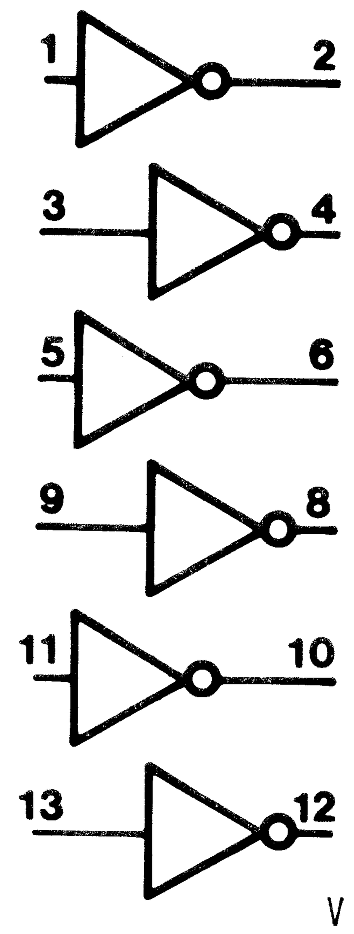
QUADRUPLE 2-INPUT
POSITIVE-NOR GATE



Vcc 14
GND 7

7404

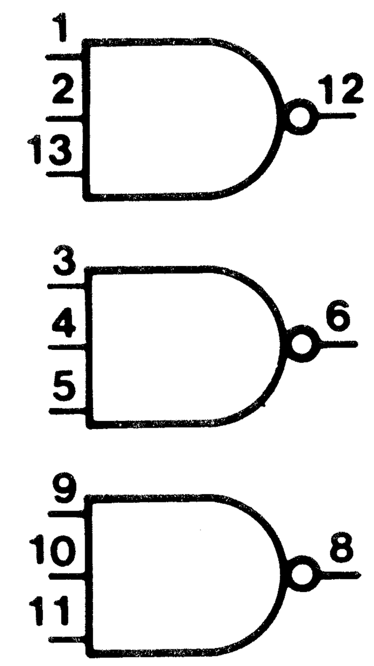
7404, S04, H04
7406 o.c. 30V
7414 sh Tris
7416 o.c. 15V



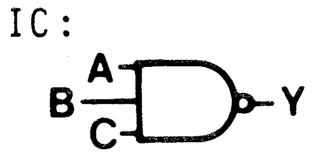
Vcc 14
GND 7

7410

7410



POSITIVE LOGIC:
$Y=\overline{ABC}$



Vcc 14
GND 7

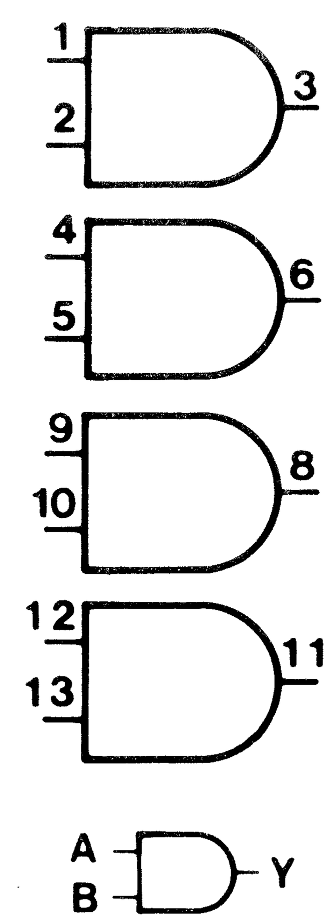
7403

(see 7400)

7408

7408
09[1]

1 open collector

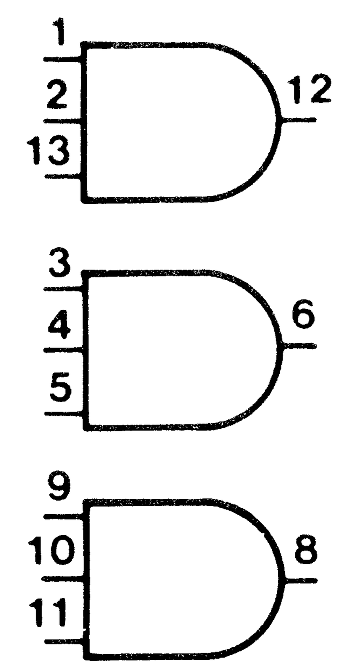


Y=AB

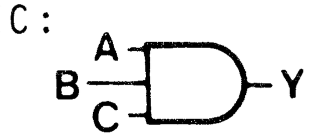
Vcc 14
GND 7

7411

7411



POSITIVE LOGIC:
Y=ABC



Vcc 14
GND 7

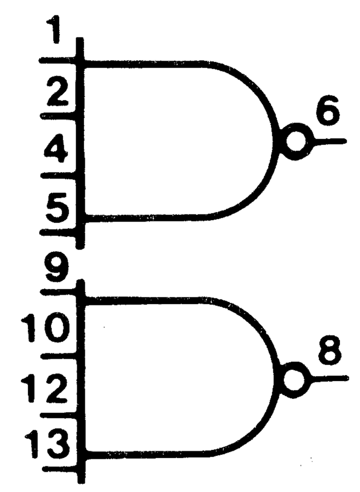
7416

(see 7404)

7420

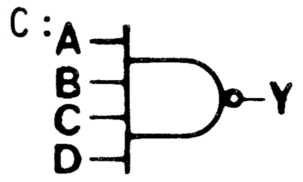
7420 7440

BUFFER



POSITIVE LOGIC:

Y=$\overline{ABCD}$



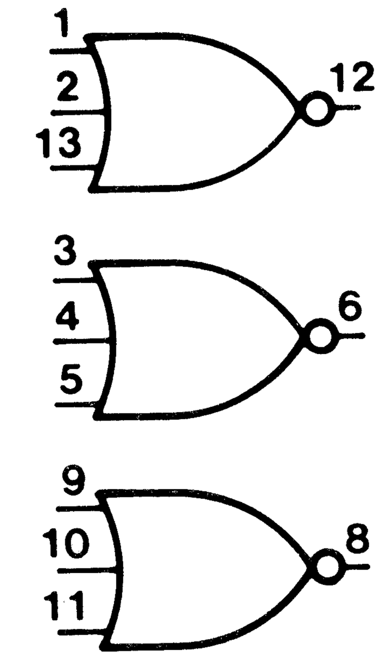
3,11 N.C

Vcc 14

GND 7

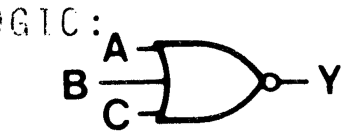
7427

7427



POSITIVE LOGIC:

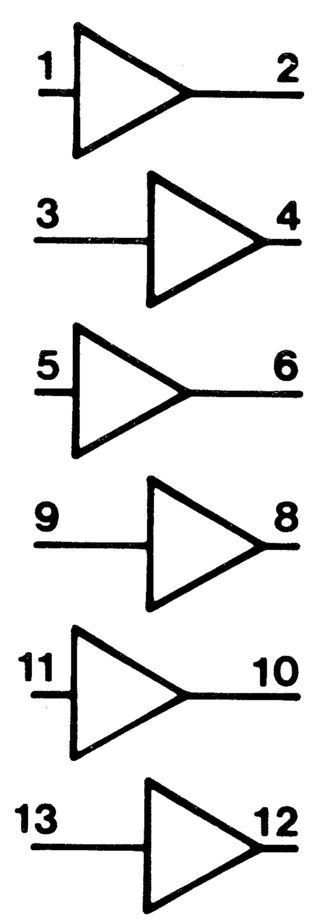
Y=$\overline{A+B+C}$



Vcc 14

GND 7

7417

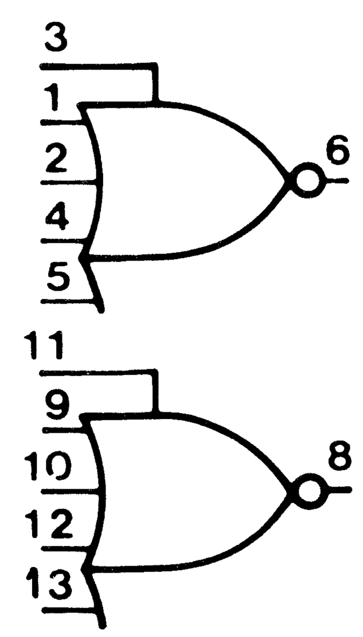


Vcc 14
GND 7

7425

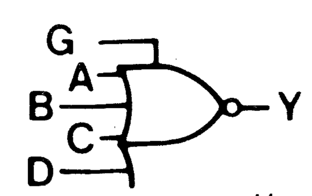
7425

4-INPUT NOR
WITH STROBE



POSITIVE LOGIC:

Y=$\overline{G(A+B+C+D)}$

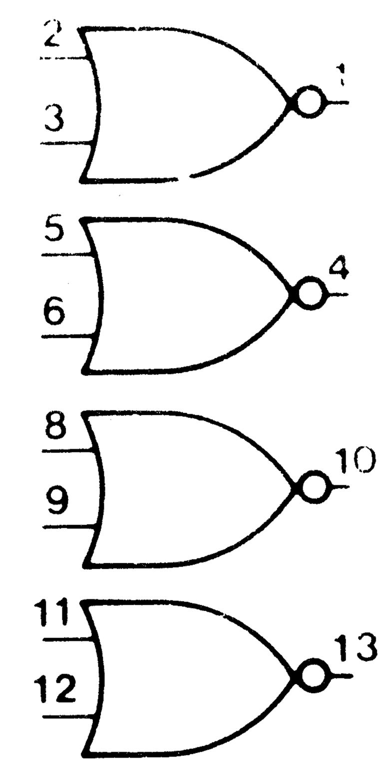


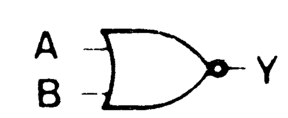
Vcc 14

GND 7

7428

2-Input Positive-NOR-Buffer





Vcc 14

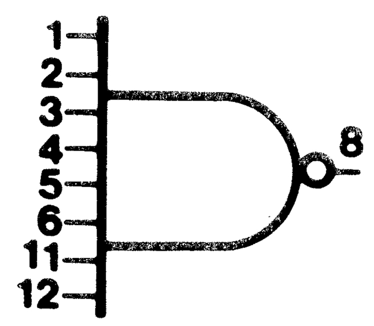
GND

Positive Logic: Y = $\overline{A+B}$

7430

7430

8-INPUT NAND GATE



Vcc 14
GND 7

N.C. 9,10,13

7437

7438

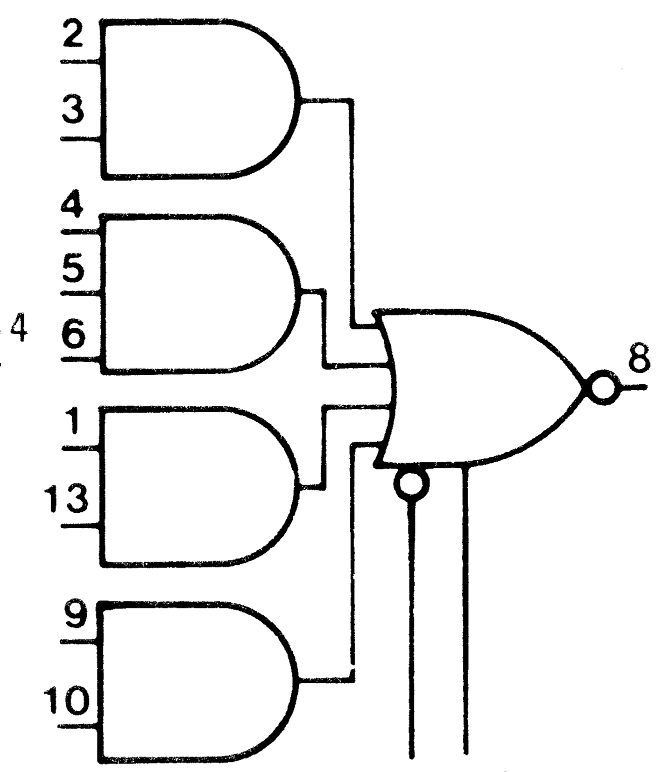
(see 7400)

7453

7453 (EXPANDABLE)

7454

4-WIDE
AND-OR-INVERT



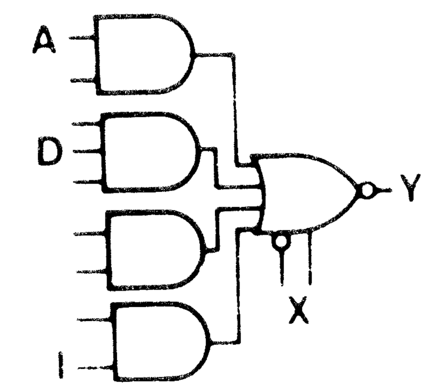
POSITIVE LOGIC:

'53
$Y=\overline{AB+CD+FG+HI+X}$

'H53
$Y=\overline{AB+CDE+FG+HI+X}$

'54
$Y=\overline{AB+CD+FG+HI}$

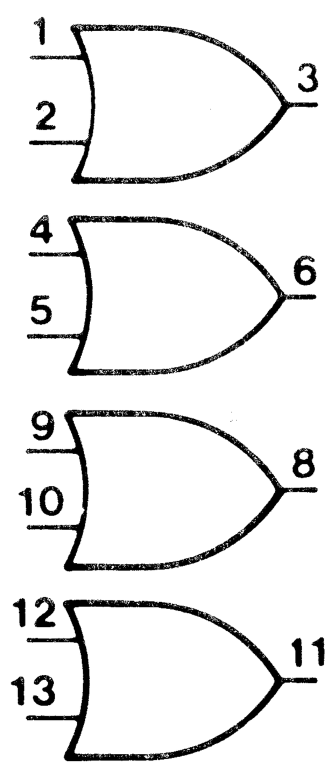
'H54
$Y=\overline{AB+CDE+FG+HI}$



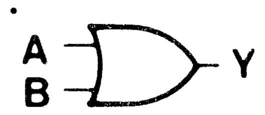
Vcc 14
GND 7

7432

7432



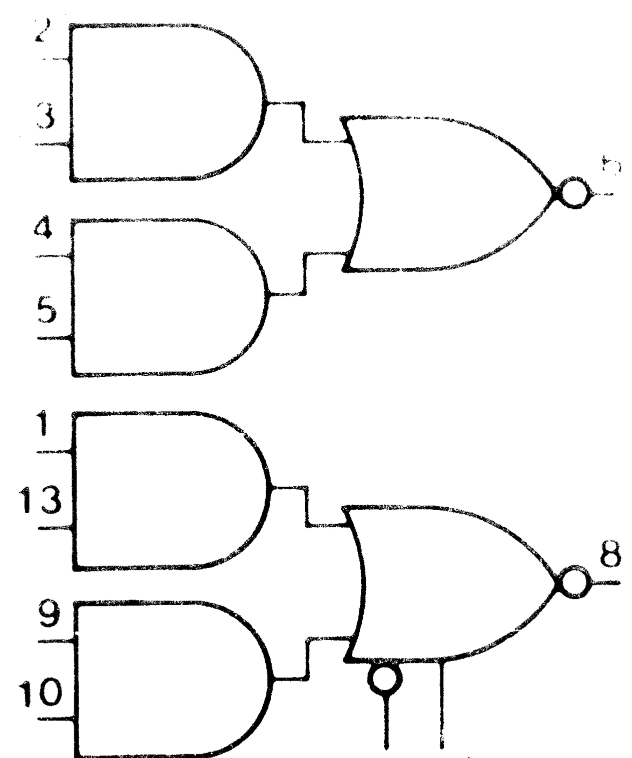
POSITIVE LOGIC:
Y=A+B



Vcc 14
GND 7

7450

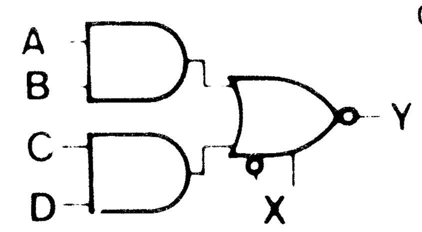
2-WIDE
2-INPUT
AND-OR-INVERT



POSITIVE LOGIC:

$Y=\overline{AB+CD+X}$

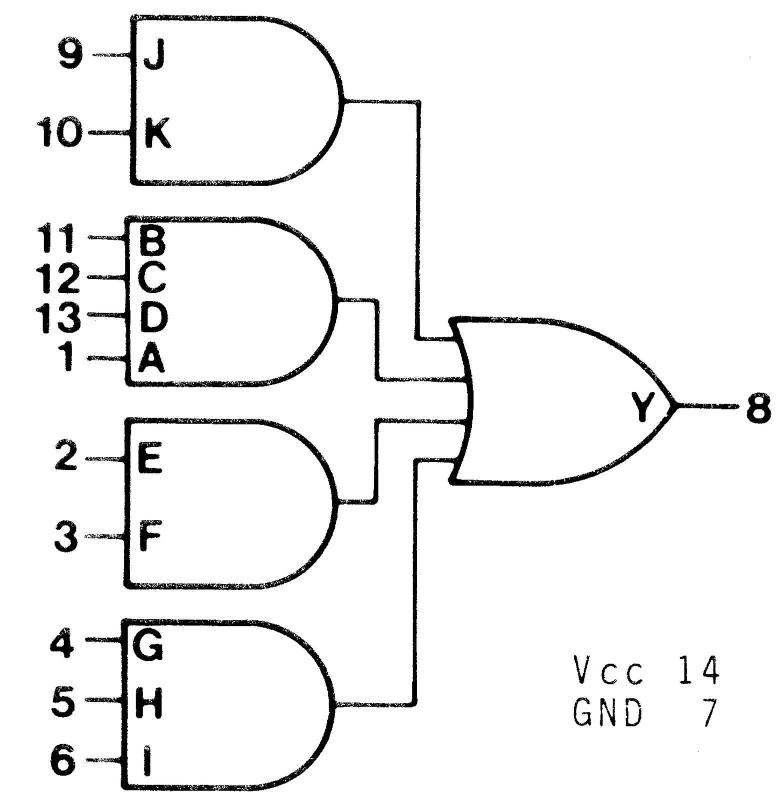
X=OUTPUT OF
7460,7462



Vcc 14
GND 7

7464

4-2-3-2-INPUT AND-OR-INVERT GATES



Vcc 14
GND 7

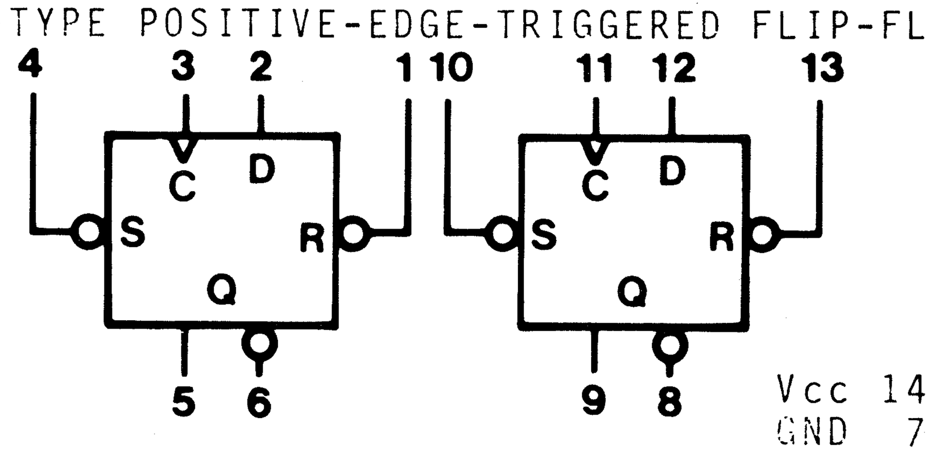
INPUTS					OUTPUTS	
PRESET	CLEAR	CLOCK	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H	H
H	H	↑	L	L	Qo	$\overline{Q}$o
H	H	↑	H	L	H	L
H	H	↑	L	H	L	H
H	H	↑	H	H	TOGGLE	
H	H	L	X	X	Qo	$\overline{Q}$o

Positive Logic: Y=ABCD+EF+GHI+JK

7474

7474, H74, S74

DUAL D-TYPE POSITIVE-EDGE-TRIGGERED FLIP-FL



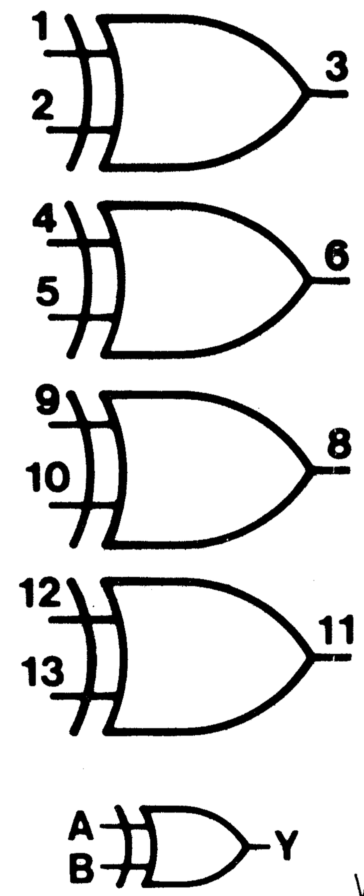
FUNCTION TABLE

INPUTS				OUTPUTS	
S	R	C	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q_0	$\overline{Q}_0$

7486

7486

QUADRUPLE 2-INPUT EXCLUSIVE-OR GATES

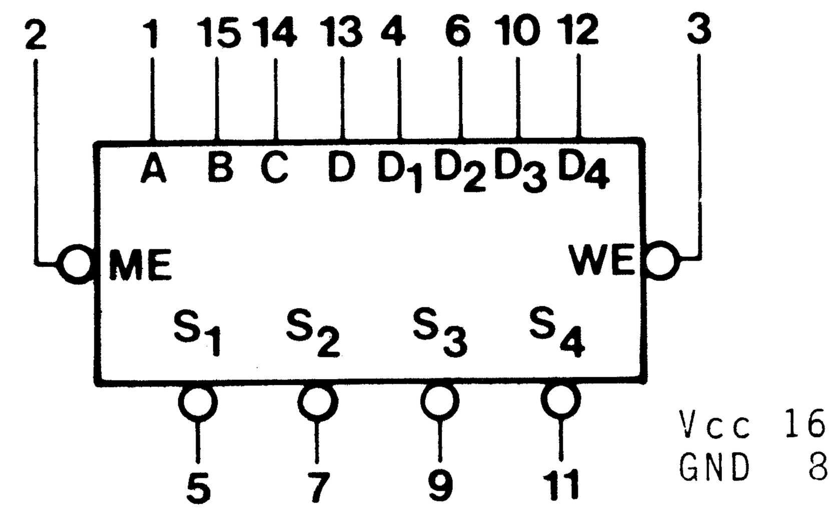


Vcc 14
GND 7

Positive Logic: $Y=A\oplus B=\overline{A}B+A\overline{B}$

7489

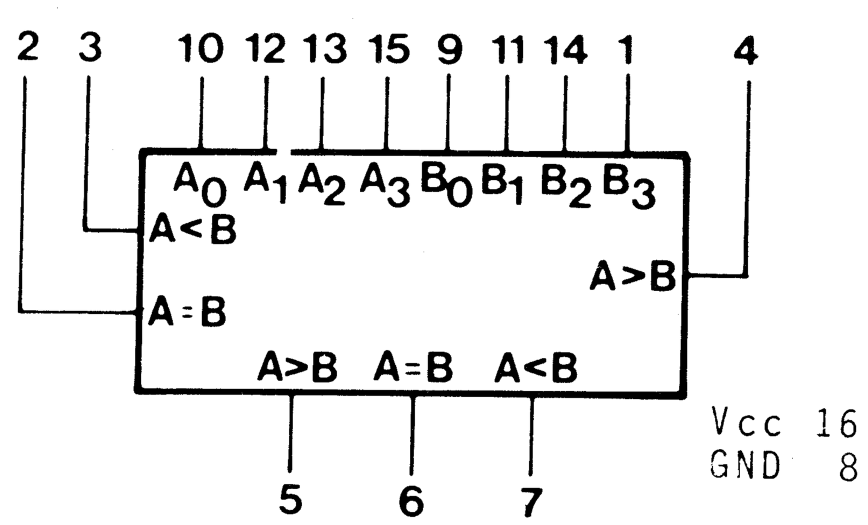
64-BIT READ/WRITE MEMORY



ME	WE	OPERATION	CONDITION OF OUTPUTS
L	L	Write	Complement of Data Inputs
L	H	Read	Complement of Selected Word
H	L	Inhibit Storage	Complement of Data Inputs
H	H	Do Nothing	High

7485

4-BIT MAGNITUDE COMPARATOR



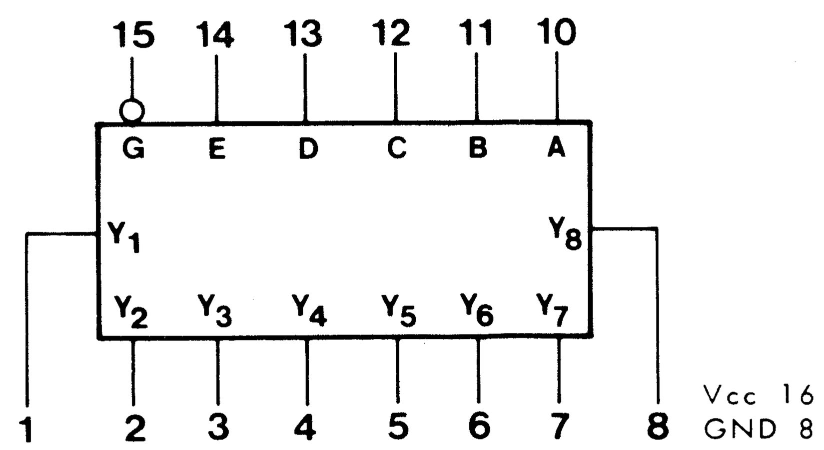
FUNCTION TABLES

COMPARING INPUTS				CASCADING INPUTS			OUTPUTS		
A3,B3	A2,B2	A1,B1	A0,B0	A>B	A<B	A=B	A>B	A<B	A=B
A3>B3	X	X	X	X	X	X	H	L	L
A3<B3	X	X	X	X	X	X	L	H	L
A3=B3	A2>B2	X	X	X	X	X	H	L	L
A3=B3	A2<B2	X	X	X	X	X	L	H	L
A3=B3	A2=B2	A1>B1	X	X	X	X	H	L	L
A3=B3	A2=B2	A1<B1	X	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0>B0	X	X	X	H	L	L
A3=B3	A2=B2	A1=B1	A0<B0	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	H	L	L	H	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	H	L	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	H	L	L	H
'85, 'S85									
A3=B3	A2=B2	A1=B1	A0=B0	X	X	H	L	L	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	L	L	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	L	H	H	L
'L85									
A3=B3	A2=B2	A1=B1	A0=B0	L	H	H	L	H	H
A3=B3	A2=B2	A1=B1	A0=B0	H	L	H	H	L	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	H	H	H	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	L	H	H	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	L	L	L	L

H=high level, L=low level; X=irrelevant

7488

256 BIT READ-ONLY MEMORY



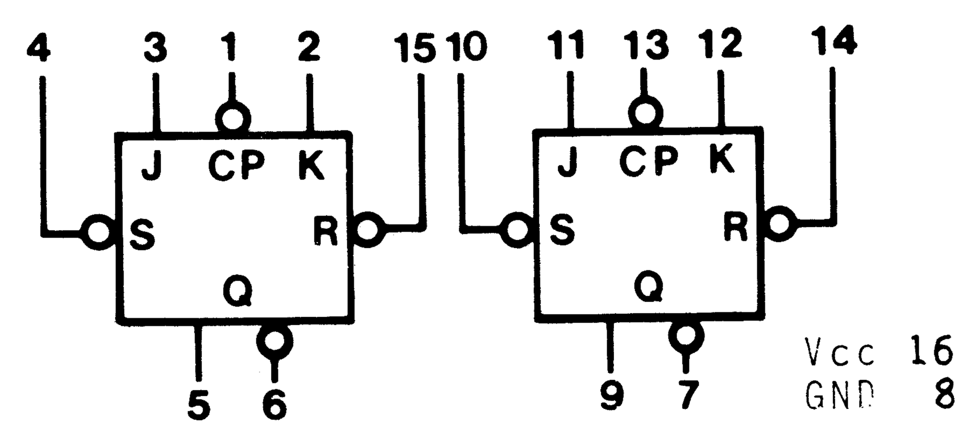
WORD-SELECT TABLE

WORD	E	D	C	B	A	WORD	E	D	C	B	A
	INPUTS						INPUTS				
0	L	L	L	L	L	16	H	L	L	L	L
1	L	L	L	L	H	17	H	L	L	L	H
2	L	L	L	H	L	18	H	L	L	H	L
3	L	L	L	H	H	19	H	L	L	H	H
4	L	L	H	L	L	20	H	L	H	L	L
5	L	L	H	L	H	21	H	L	H	L	H
6	L	L	H	H	L	22	H	L	H	H	L
7	L	L	H	H	H	23	H	L	H	H	H
8	L	H	L	L	L	24	H	H	L	L	L
9	L	H	L	L	H	25	H	H	L	L	H
10	L	H	L	H	L	26	H	H	L	H	L
11	L	H	L	H	H	27	H	H	L	H	H
12	L	H	H	L	L	28	H	H	H	L	L
13	L	H	H	L	H	29	H	H	H	L	H
14	L	H	H	H	L	30	H	H	H	H	L
15	L	H	H	H	H	31	H	H	H	H	H

74112

74S112

J-K NEGATIVE-EDGE-TRIGGERED FLIP-FLOPS

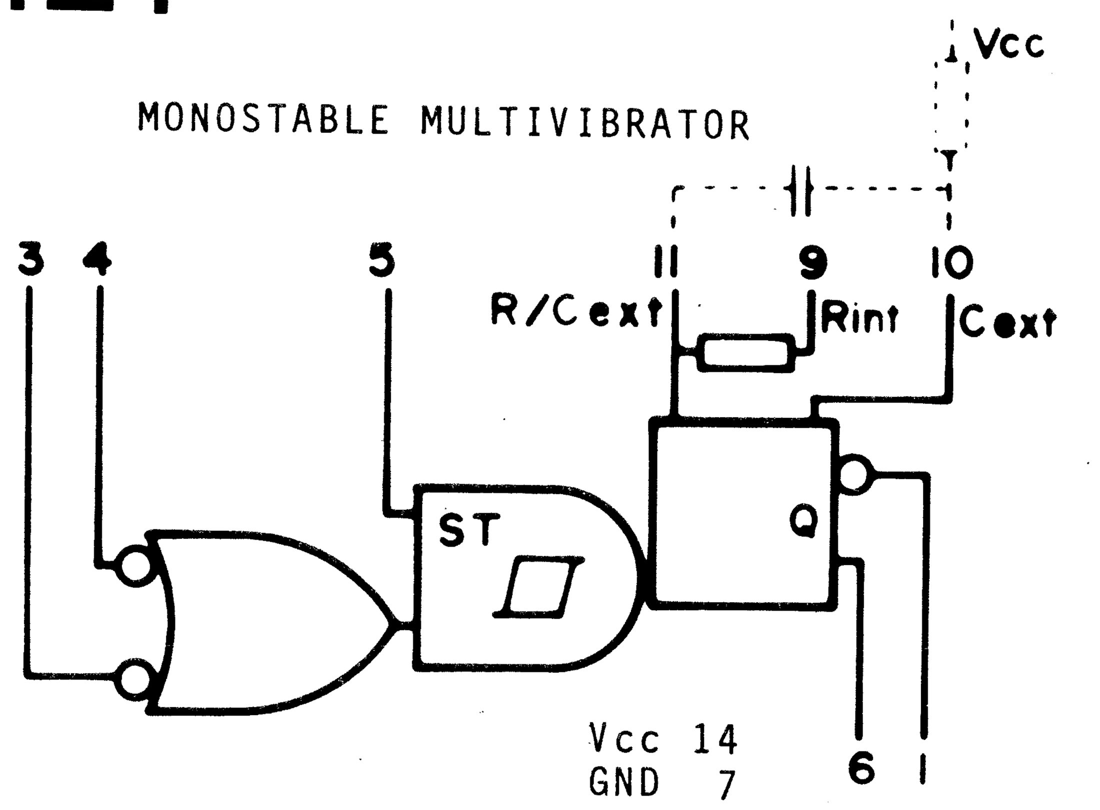


FUNCTION TABLE

INPUTS					OUTPUTS	
S	R	C	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	H	H
L	L	X	X	X	H	H
H	H	↓	L	L	Qo	Qo
H	H	↓	H	L	H	L
H	H	↓	L	H	L	H
H	H	↓	H	H	TOGGLE	
H	H	H	X	X	Q_0	$\overline{Q}_0$

74121

MONOSTABLE MULTIVIBRATOR



To use the internal timing resistor connect Rint to Vcc.

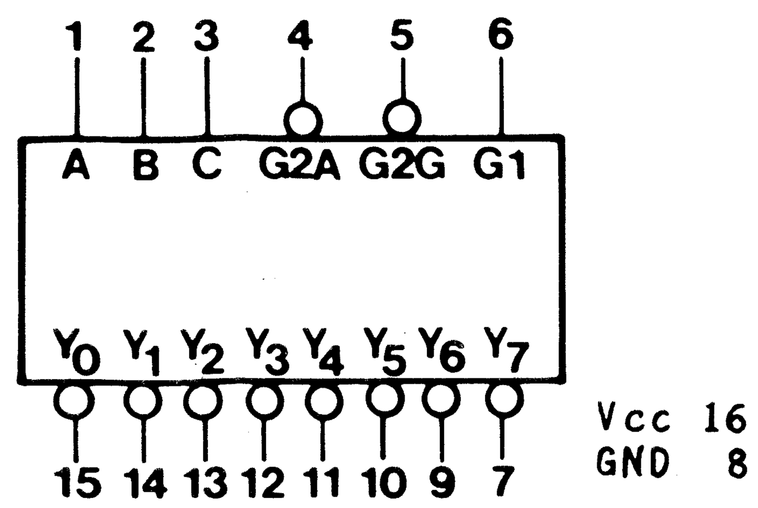
An external timing capacitor may be connected between Cext and Rext/Cext.

For accurate repeatable pulse widths, connect an external resistor between Rext/Cext and Vcc with Rint open-circuited.

To obtain variable pulse widths, connect external variable resistance between Rint or Rext/Cext and Vcc.

74138

DECODER



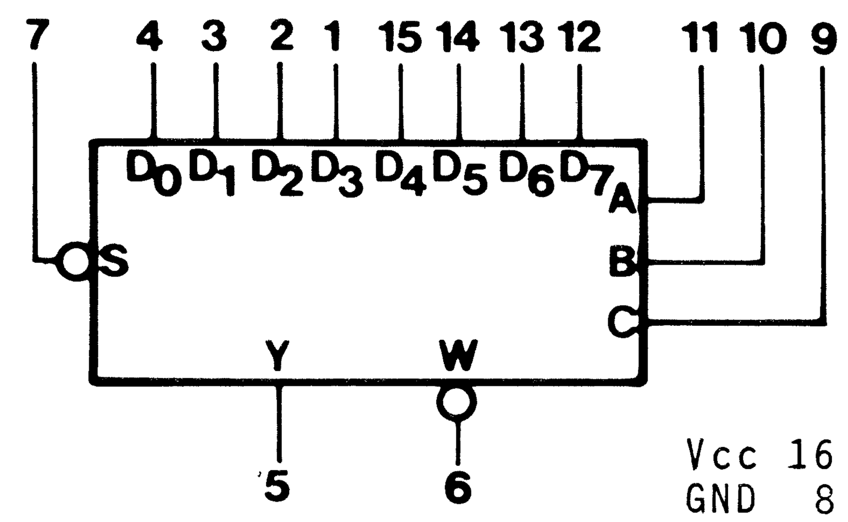
FUNCTION TABLE

INPUTS					OUTPUTS							
ENABLE		SELECT										
G1	'G2	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

'G2 = G2A+G2B

74151

DATA SELECTOR/MULTIPLEXER



FUNCTION TABLE

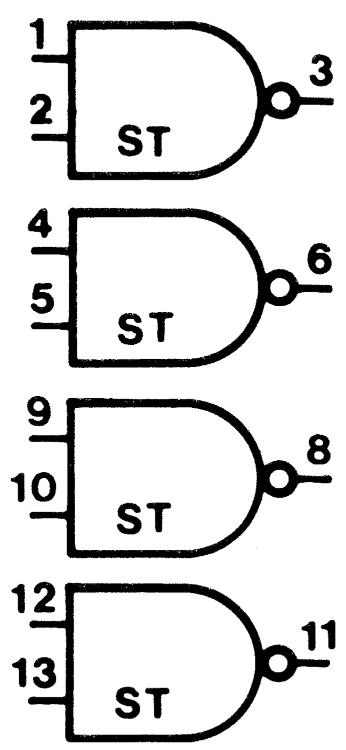
INPUTS				OUTPUTS	
SELECT			STROBE		
C	B	A		Y	W
X	X	X	H	L	H
L	L	L	L	D0	D0
L	L	H	L	D1	D1
L	H	L	L	D2	D2
L	H	H	L	D3	D3
H	L	L	L	D4	D4
H	L	H	L	D5	D5
H	H	L	L	D6	D6
H	H	H	L	D7	D7

D0,D1...D7 = the level of the D respective input

74132

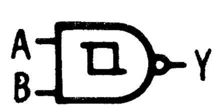
74132

POSITIVE NAND
SCHMITT TRIGGER



POSITIVE LOGIC:

Y=AB



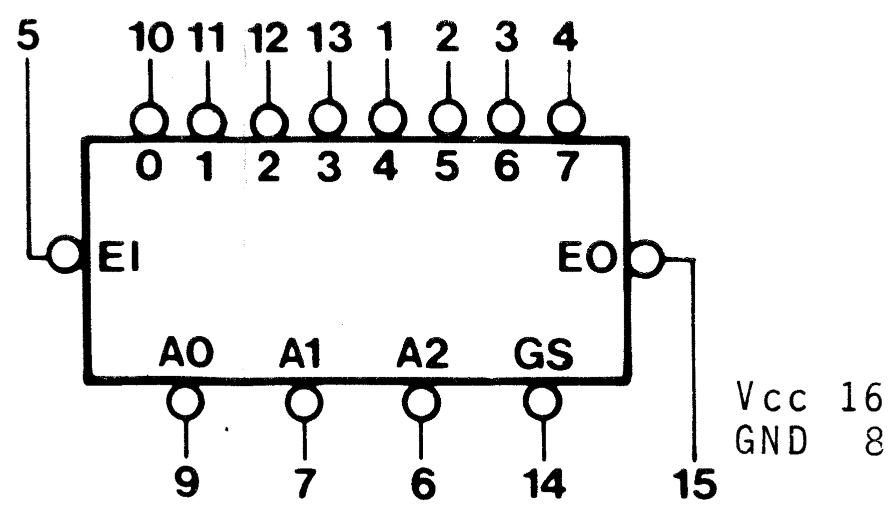
Vcc 14
GND 7

Positive going threshold V=1.7

Negative going threshold V=0.9

74148

8-LINE-TO-3-LINE PRIORITY ENCODER

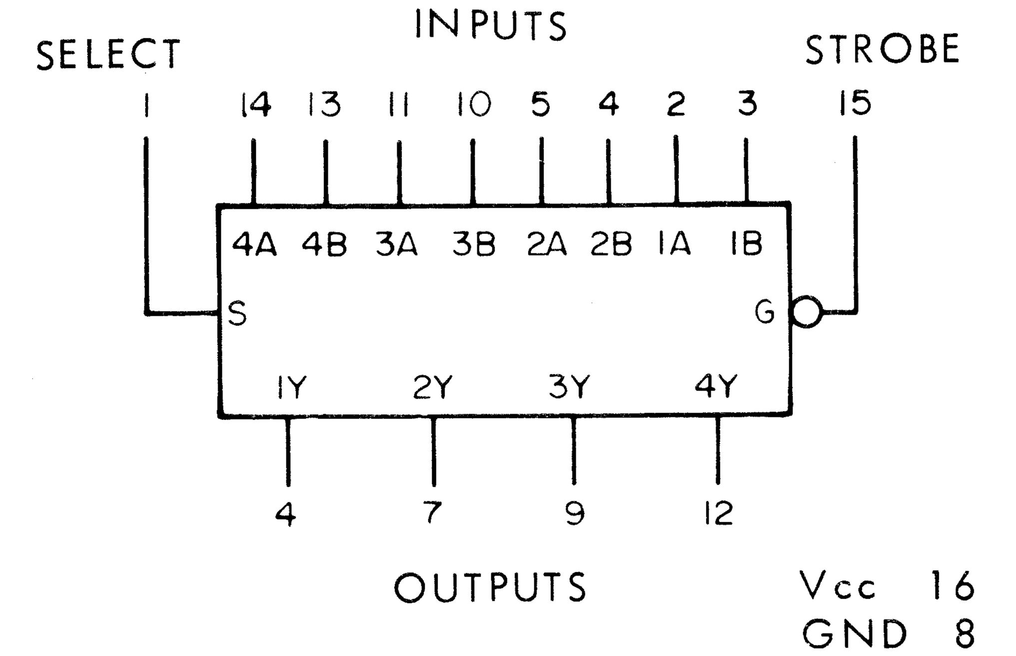


FUNCTION TABLE

INPUTS									OUTPUTS				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

74157

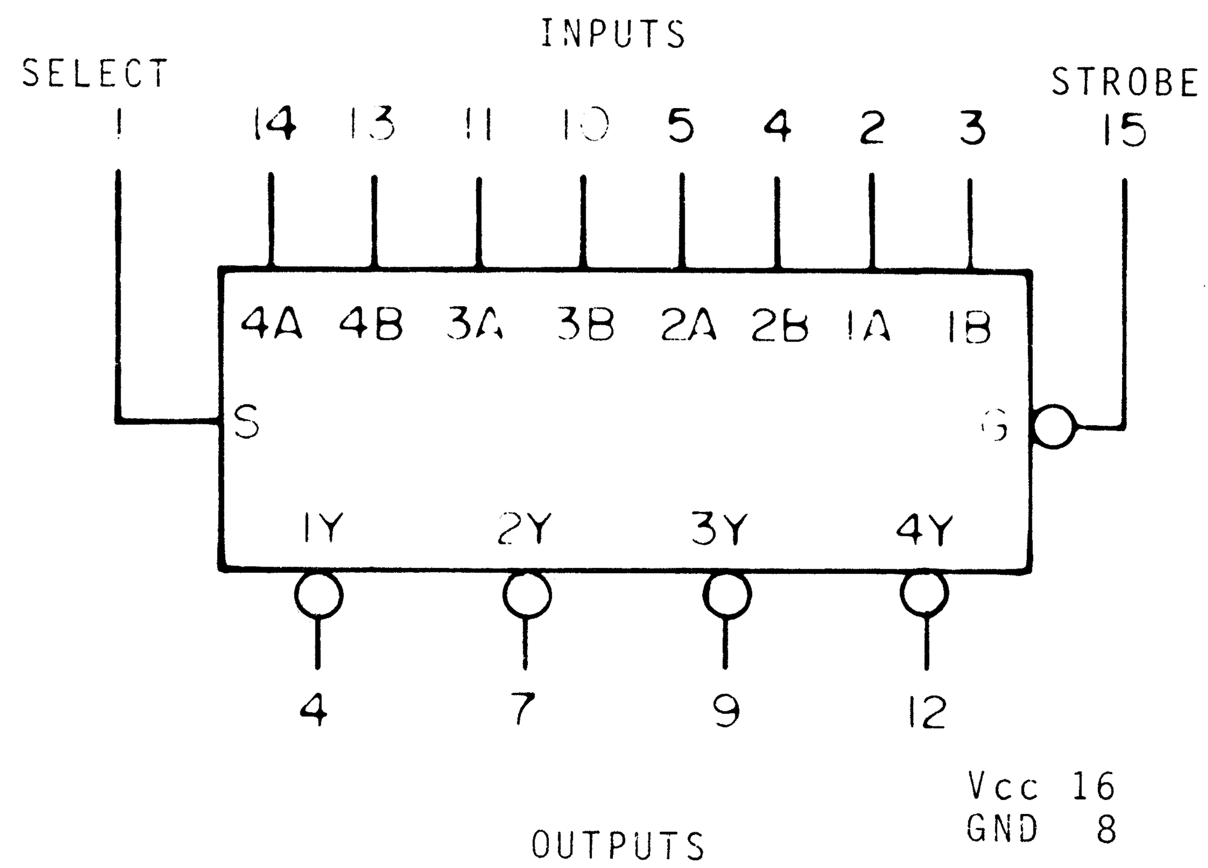
QUAD 2-to-1-LINE DATA SELECTORS/MULTIPLEXERS



74158

74158

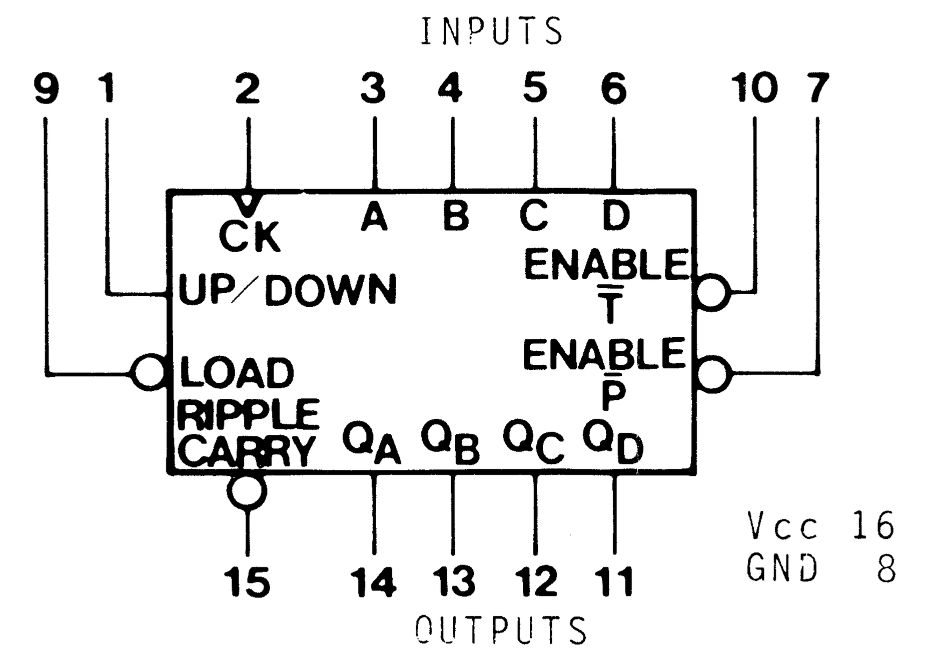
QUAD 2-to-1-LINE DATA SELECTORS/MULTIPLEXERS



74169

74169

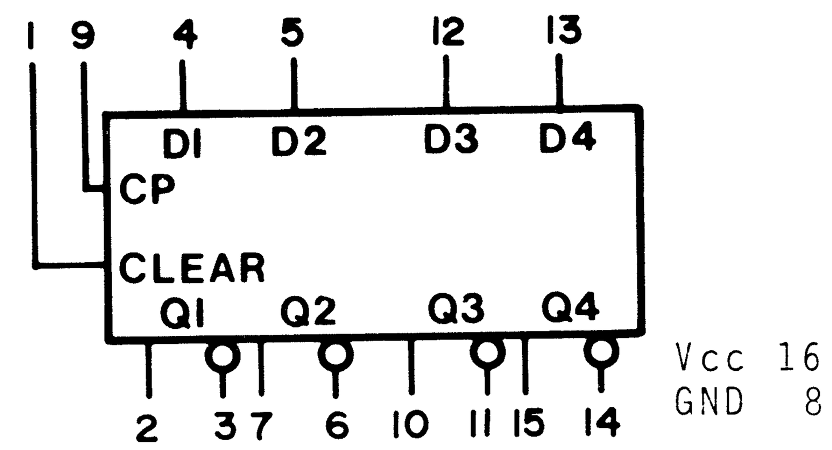
SYNCHRONOUS 4-BIT UP/DOWN COUNTERS



74175

74175

QUAD D-TYPE FLIP-FLOPS



FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\bar{Q}$
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	Qo

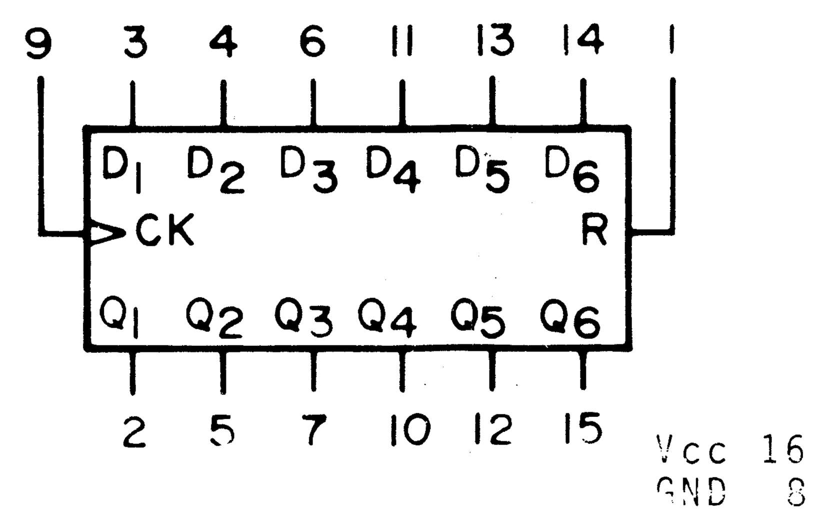
74161

(see 9310)

74174

74174

HEX D-TYPE FLIP-FLOPS WITH COMMON CLOCK AND RESET

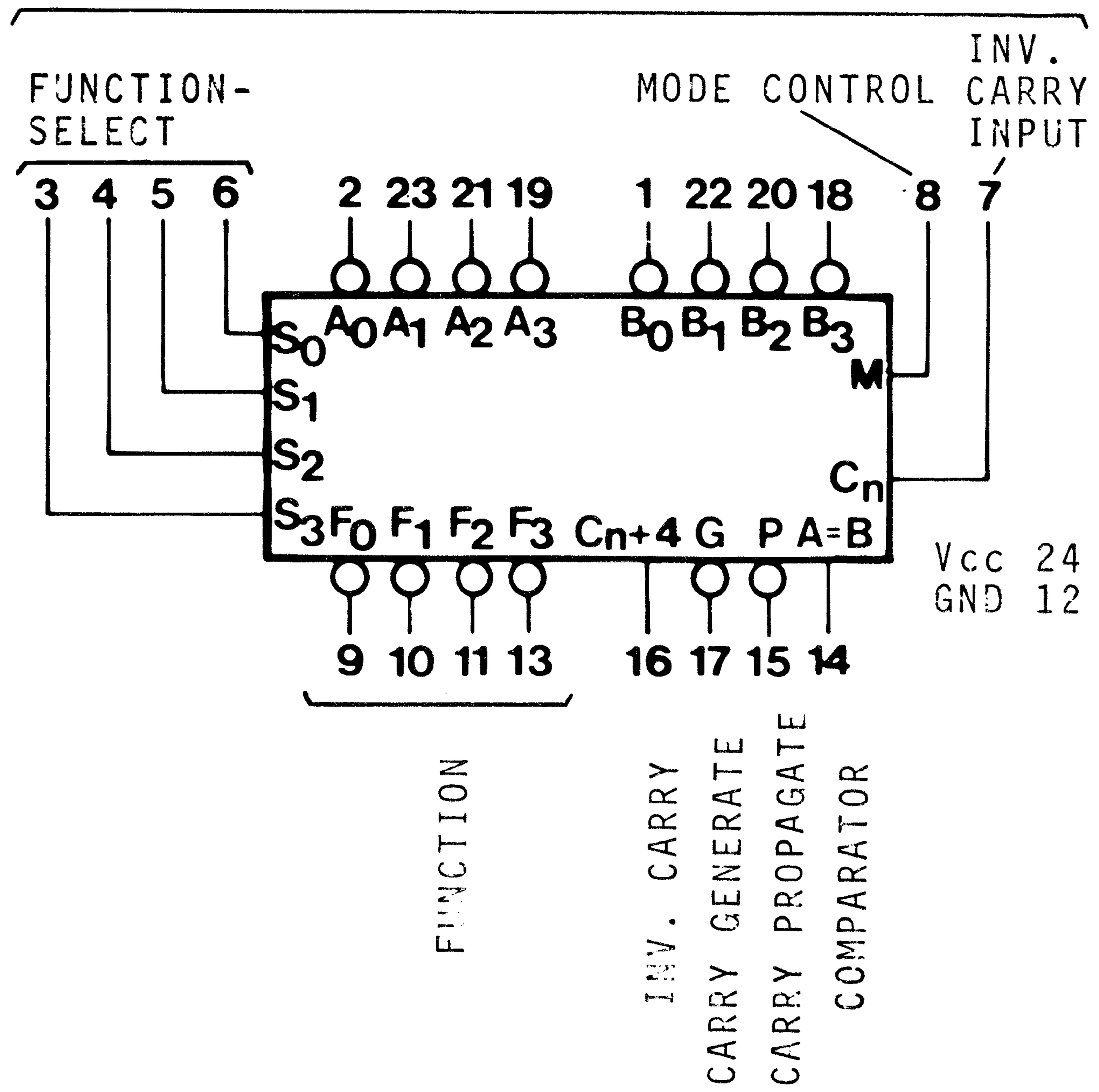


FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\bar{Q}t$
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	$\bar{Q}o$

X

74181

ARITHMETIC LOGIC UNIT



Vcc 24
GND 12

Table 1

SELECTION				ACTIVE-HIGH DATA		
				M = H LOGIC FUNCTIONS	M = L, ARITHMETIC OPERATIONS	
S3	S2	S1	S0		Cn = H (no carry)	Cn = L (with carry)
L	L	L	L	$F = \bar{A}$	F = A	F = A plus 1
L	L	L	H	$F = \overline{A + B}$	F = A + B	F = (A + B) plus 1
L	L	H	L	$F = \bar{A}B$	$F = A + \bar{B}$	$F = (A + \bar{B})$ plus 1
L	L	H	H	F = 0	F = minus 1 (2's compl)	F = zero
L	H	L	L	$F = \overline{AB}$	$F = A$ plus $A\bar{B}$	$F = A$ plus $A\bar{B}$ plus 1
L	H	L	H	$F = \bar{B}$	$F = (A + B)$ plus $A\bar{B}$	$F = (A + B)$ plus $A\bar{B}$ plus 1
L	H	H	L	$F = A \oplus B$	F = A minus B minus 1	F = A minus B
L	H	H	H	$F = A\bar{B}$	$F = A\bar{B}$ minus 1	$F = A\bar{B}$
H	L	L	L	$F = \bar{A} + B$	F = A plus AB	F = A plus AB plus 1
H	L	L	H	$F = \overline{A \oplus B}$	F = A plus B	F = A plus B plus 1
H	L	H	L	F = B	$F = (A + \bar{B})$ plus AB	$F = (A + \bar{B})$ plus AB plus 1
H	L	H	H	F = AB	F = AB minus 1	F = AB
H	H	L	L	F = 1	F = A plus A*	F = A plus A plus 1
H	H	L	H	$F = A + \bar{B}$	F = (A + B) plus A	F = (A + B) plus A plus 1
H	H	H	L	F = A + B	$F = (A + \bar{B})$ plus A	$F = (A + \bar{B})$ plus A plus 1
H	H	H	H	F = A	F = A minus 1	F = A

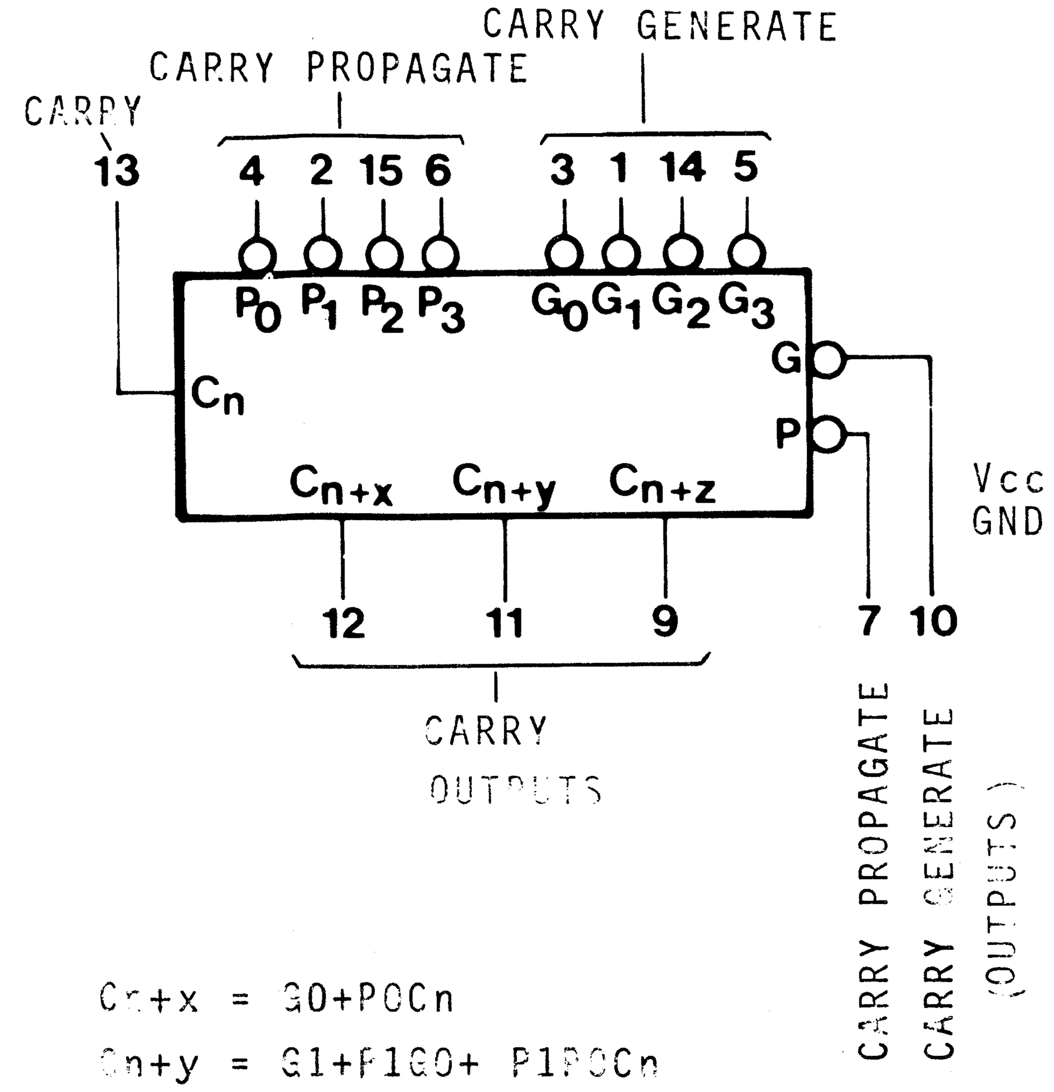
Table 2

SELECTION				ACTIVE-LOW DATA		
				M=H LOGIC FUNCTIONS	M = L, ARITHMETIC OPERATIONS	
S3	S2	S1	S0		Cn = L (no carry)	Cn = H (with carry)
L	L	L	L	$F = \bar{A}$	F = A minus 1	F = A
L	L	L	H	$F = \overline{AB}$	F = AB minus 1	F = AB
L	L	H	L	$F = \bar{A} + B$	$F = A\bar{B}$ minus 1	$F = A\bar{B}$
L	L	H	H	F = 1	F = minus 1 (2's comp)	F = zero
L	H	L	L	$F = \overline{A + B}$	$F = A$ plus $(A + \bar{B})$	$F = A$ plus $(A + \bar{B})$ plus 1
L	H	L	H	$F = \bar{B}$	$F = AB$ plus $(A + \bar{B})$	$F = AB$ plus $(A + \bar{B})$ plus 1
L	H	H	L	$F = \overline{A \oplus B}$	F = A minus B minus 1	F = A minus B
L	H	H	H	$F = A + \bar{B}$	$F = A + \bar{B}$	$F = (A + \bar{B})$ plus 1
H	L	L	L	$F = \bar{A}B$	F = A plus (A + B)	F = A plus (A + B) plus 1
H	L	L	H	$F = A \oplus B$	F = A plus B	F = A plus B plus 1
H	L	H	L	F = B	$F = A\bar{B}$ plus (A + B)	$F = A\bar{B}$ plus (A + B) plus 1
H	L	H	H	F = A + B	F = A + B	F = (A + B) plus 1
H	H	L	L	F = 0	F = A plus A*	F = A plus A plus 1
H	H	L	H	$F = A\bar{B}$	F = AB plus A	F = AB plus A plus 1
H	H	H	L	F = AB	$F = A\bar{B}$ plus A	$F = A\bar{B}$ plus A plus 1
H	H	H	H	F = A	F = A	F = A plus 1

* Each bit is shifted to the next more significant position.

74182

LOOK-AHEAD CARRY GENERATOR



Vcc 16
GND 8

$Cn+x = G0+P0Cn$

$Cn+y = G1+P1G0+P1P0Cn$

$Cn+z = G2+P2G1+P2P1G0+P2P1P0Cn$

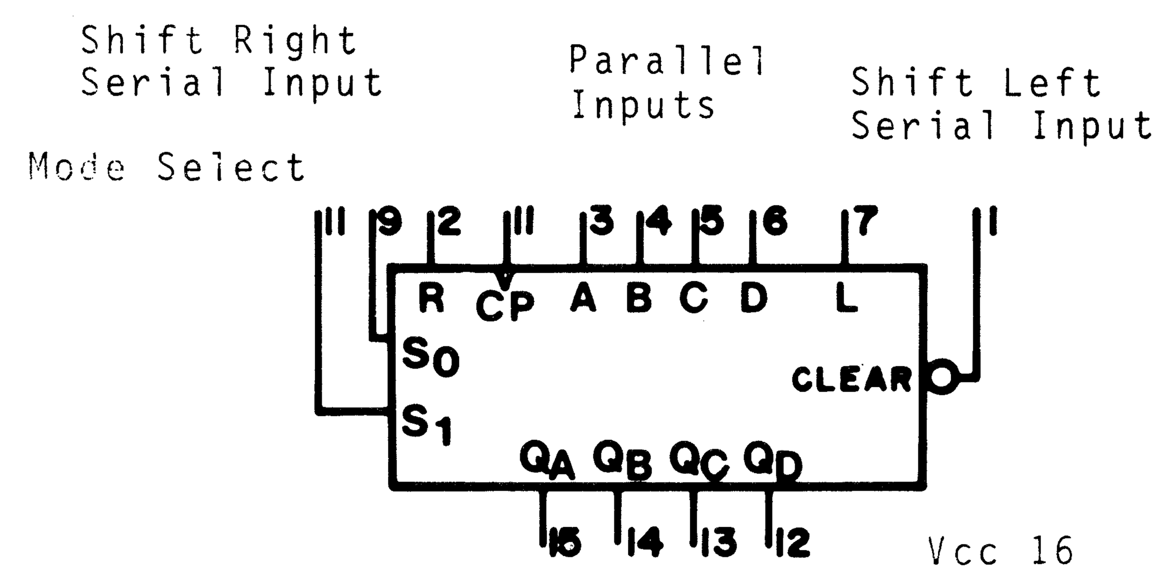
$G = G3(P3+G2)(P3+P2+G1)(P3+P2+P1+G0)$

$P = P3P2P1P0$

74194

74194

4-BIT SHIFT REGISTER

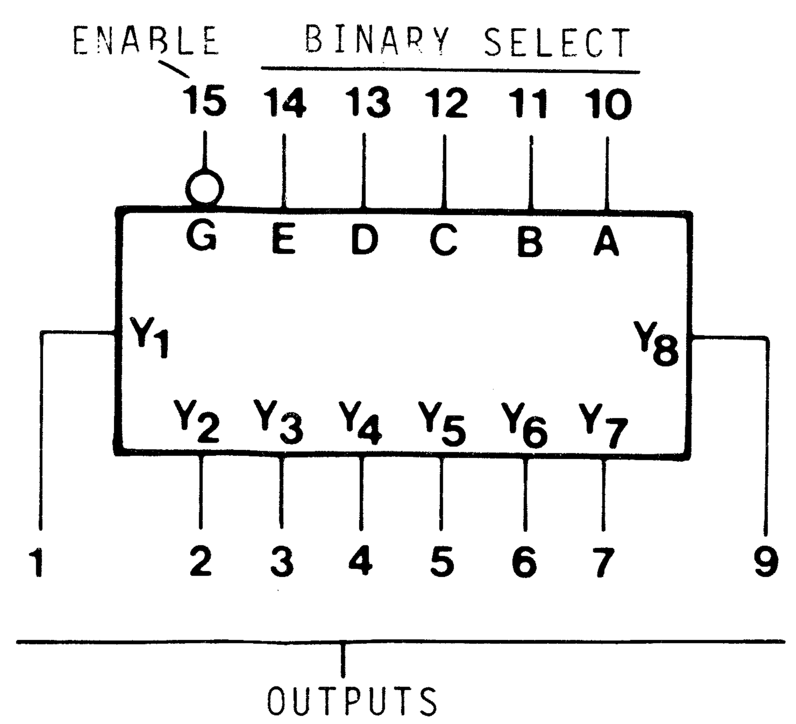


Vcc 16
GND 8

74188

74188A

256-BIT PROGRAMMABLE READ-ONLY MEMORY

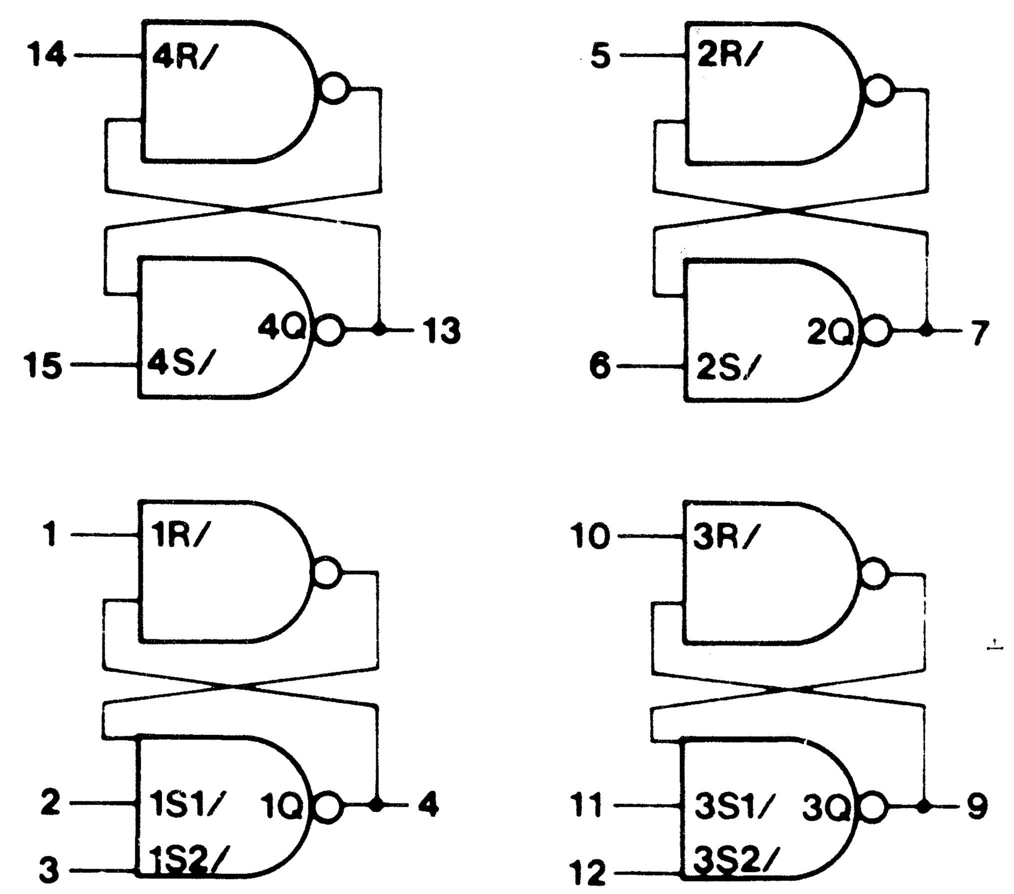


Vcc 16
GND 8

Organized as 32 Words of 8 Bits Each

74279

QUADRUPLE S-R LATCHES



FUNCTION TABLE

INPUTS S*	INPUTS R	OUTPUT Q
H	H	Qo
L	H	H
H	L	L
L	L	H'

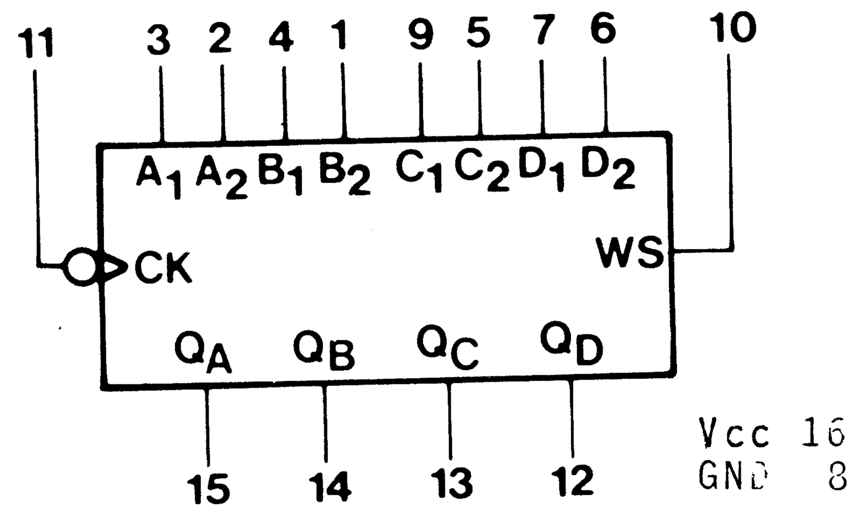
Vcc 16
GND 8

* For latches with double S inputs:
H = both S inputs high
L = one or both S inputs low

Qo = the level of Q before the indicated input conditions were established.

' This output level is pseudo stable; that is, it may not persist when the S and R inputs return to their inactive (high) level.

74298 QUADRUPLE 2-INPUT MULTIPLEXER WITH STORAGE



Vcc 16
GND 8

FUNCTION TABLE

INPUTS		OUTPUTS			
WORD SELECT	CLOCK	QA	QB	QC	QD
L	↓	a1	b1	c1	d1
H	↓	a2	b2	c2	d2
X	H	QA0	QB0	QC0	QD0

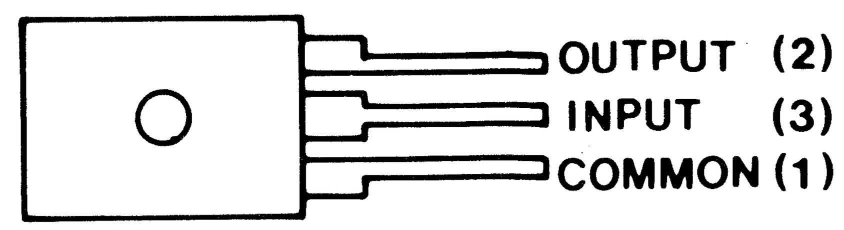
a1,a2,etc. = the level of steady-state input at A1,A2,etc.

QA0,QB0,etc. = the level of QA,QB,etc. entered on the most-recent ↓ transition of the clock input.

7905

7905 (-5V)

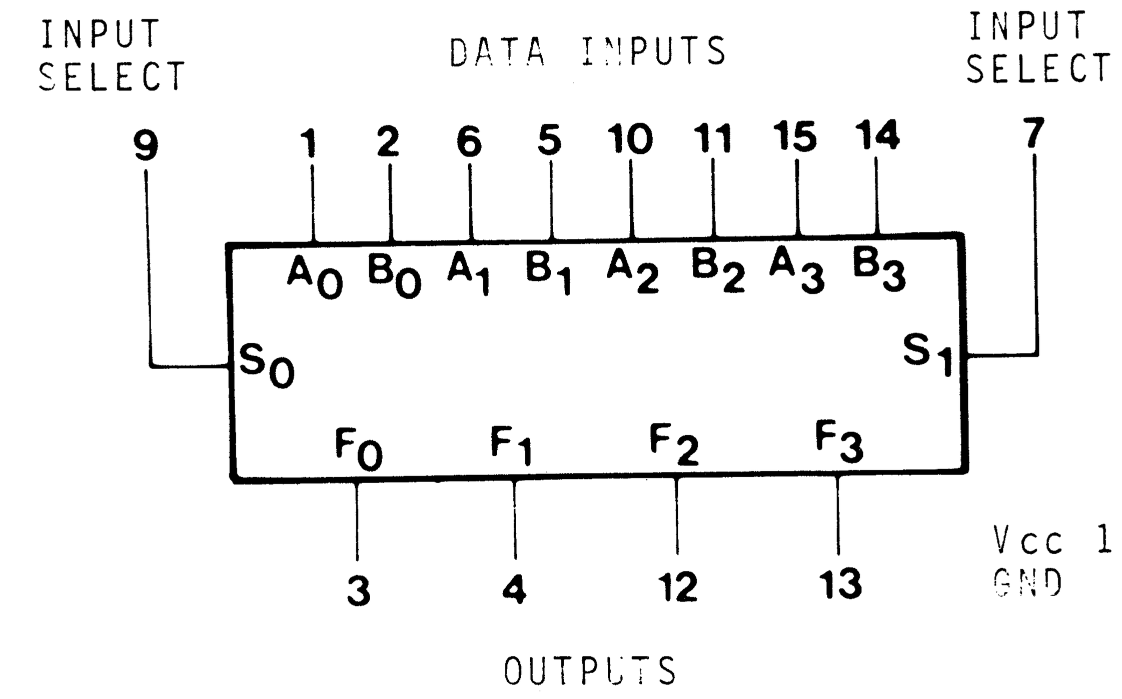
NEGATIVE VOLTAGE REGULATOR



8234

8234

QUAD 2-INPUT MULTIPLEXER



Vcc 16
GND 8

OUTPUTS

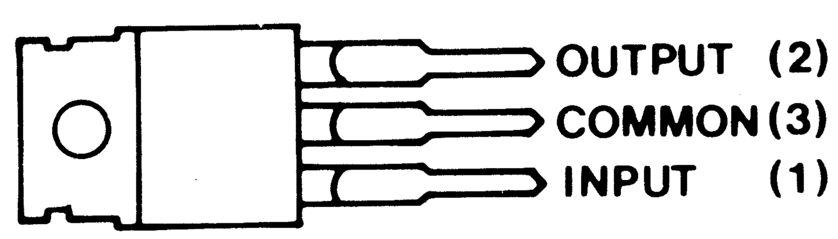
S0	S1	Selects
0	0	B
0	1	B
1	0	A
1	1	High Out

7812
7815
7912

7812C (+12V)

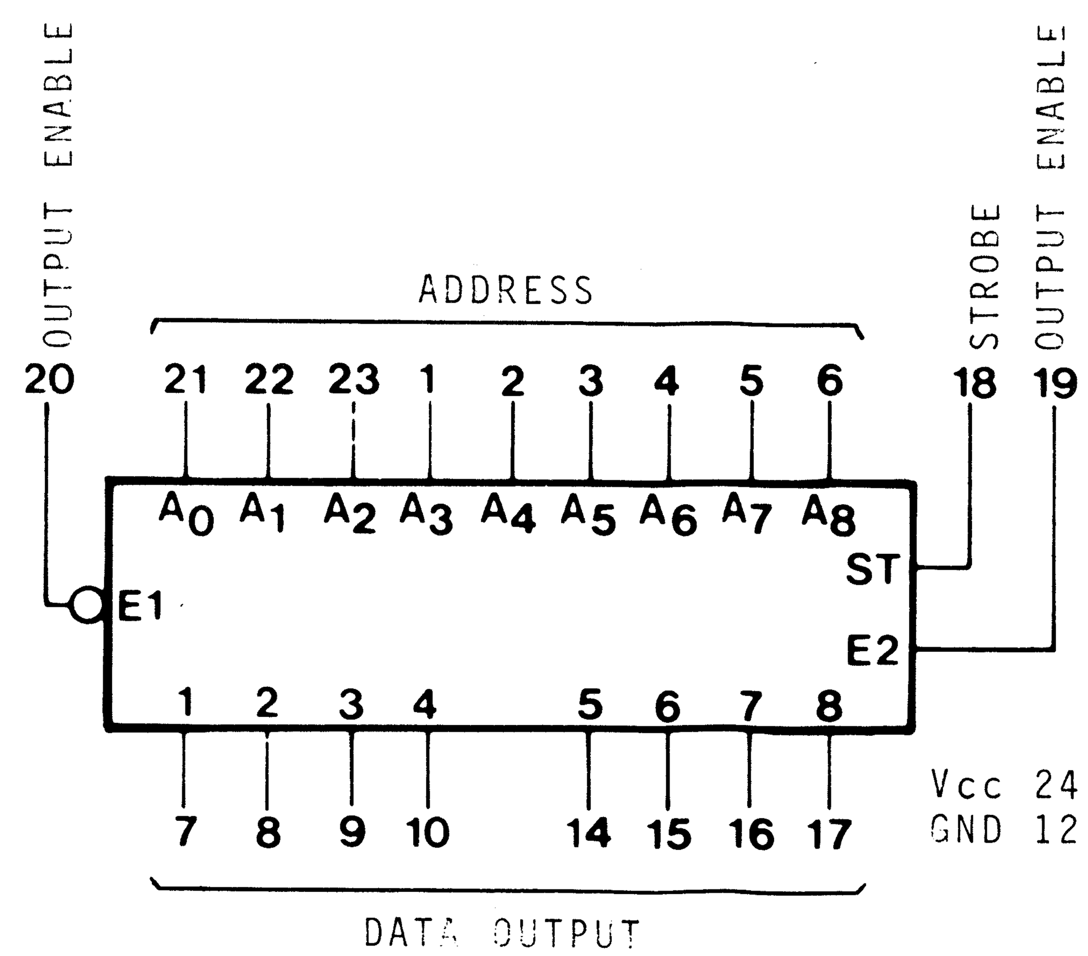
7815C (+15V)

POSITIVE VOLTAGE REGULATOR



8205

512x8 ROM WITH 3 STATE OUTPUTS



Vcc 24
GND 12

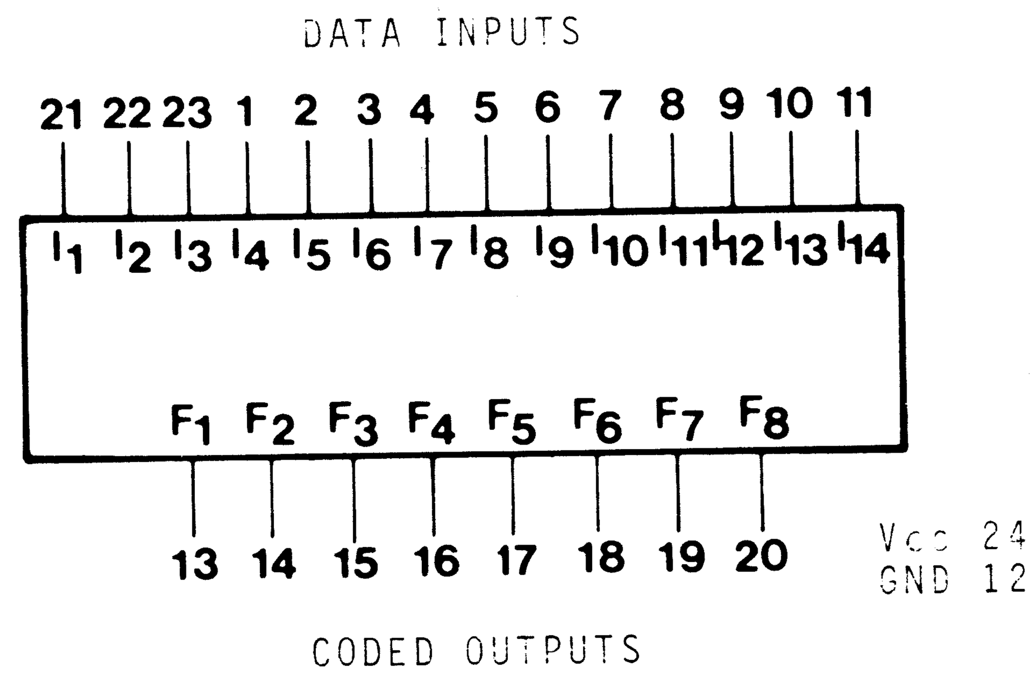
Strobe High E1/E2 gate the output

Strobe Low Last addressed word is stopped at the output

8576

8576

PLA DATA ENCODER



Vcc 24
GND 12

PLA = Programmable Logic Array

96 Input combinations are coded
(Refer to code table for specific unit)

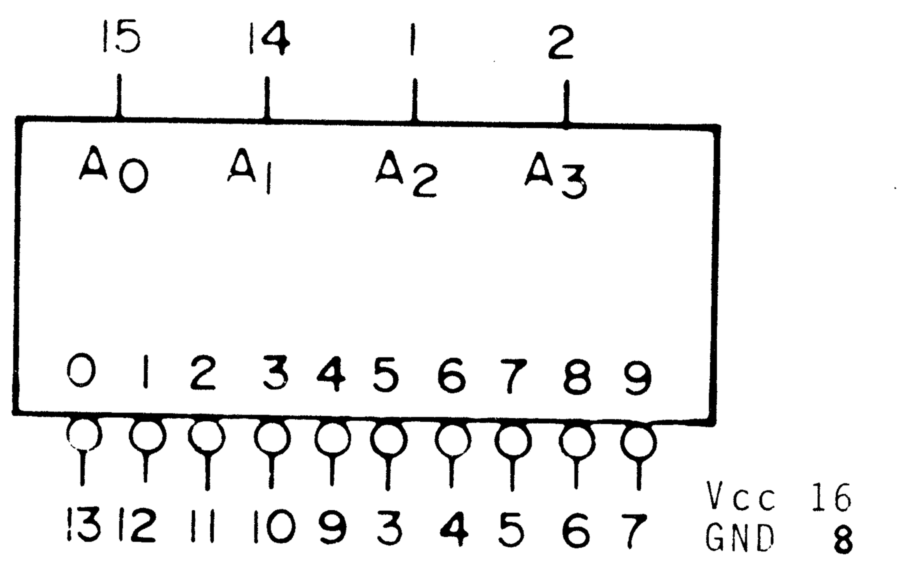
9301

9301

ONE-OF-TEN DECODER

Truth Table

A0	A1	A2	A3	Output Low
L	L	L	L	0
H	L	L	L	1
L	H	L	L	2
H	H	L	L	3
L	L	H	L	4
H	L	H	L	5
L	H	H	L	6
H	H	H	L	7
L	L	L	H	8
H	L	L	H	9

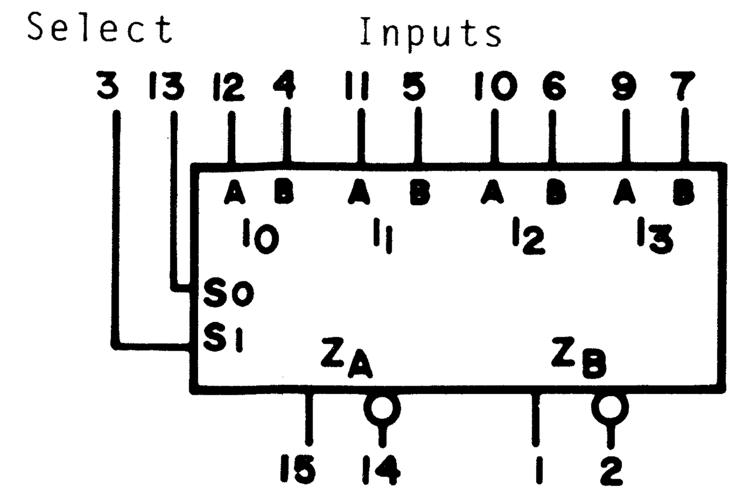


Vcc 16
GND 8

Multipurpose Decoder will accept four inputs and provide 10 mutually exclusive outputs.

9309

DUAL, 4-INPUT MULTIPLEXER

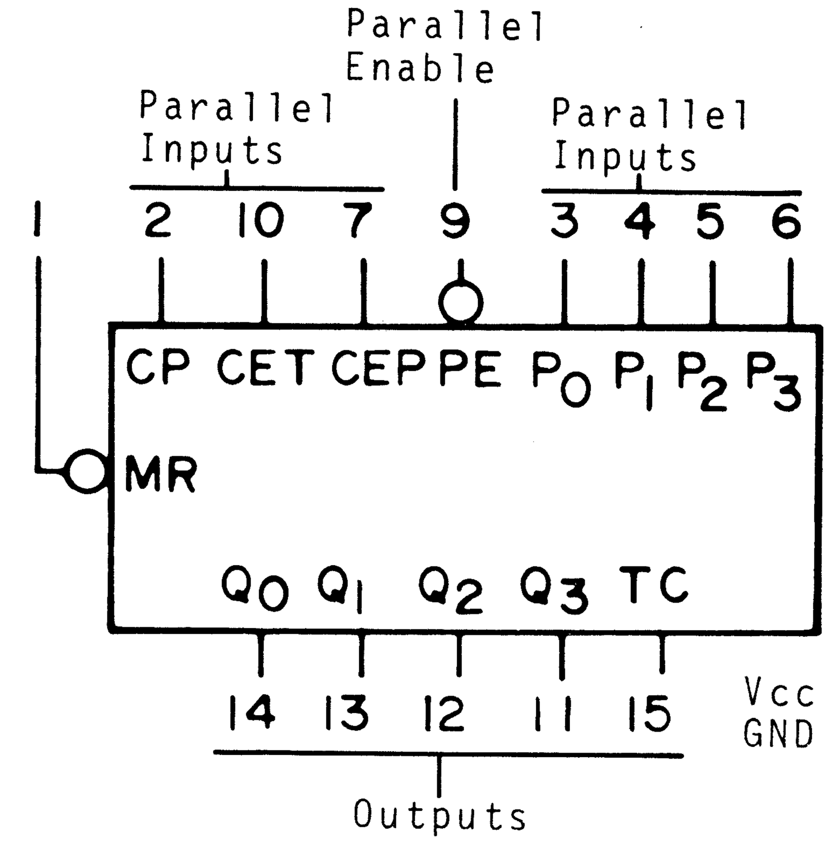


Vcc 16
GND 8

9316

9316

4-BIT BINARY COUNTER

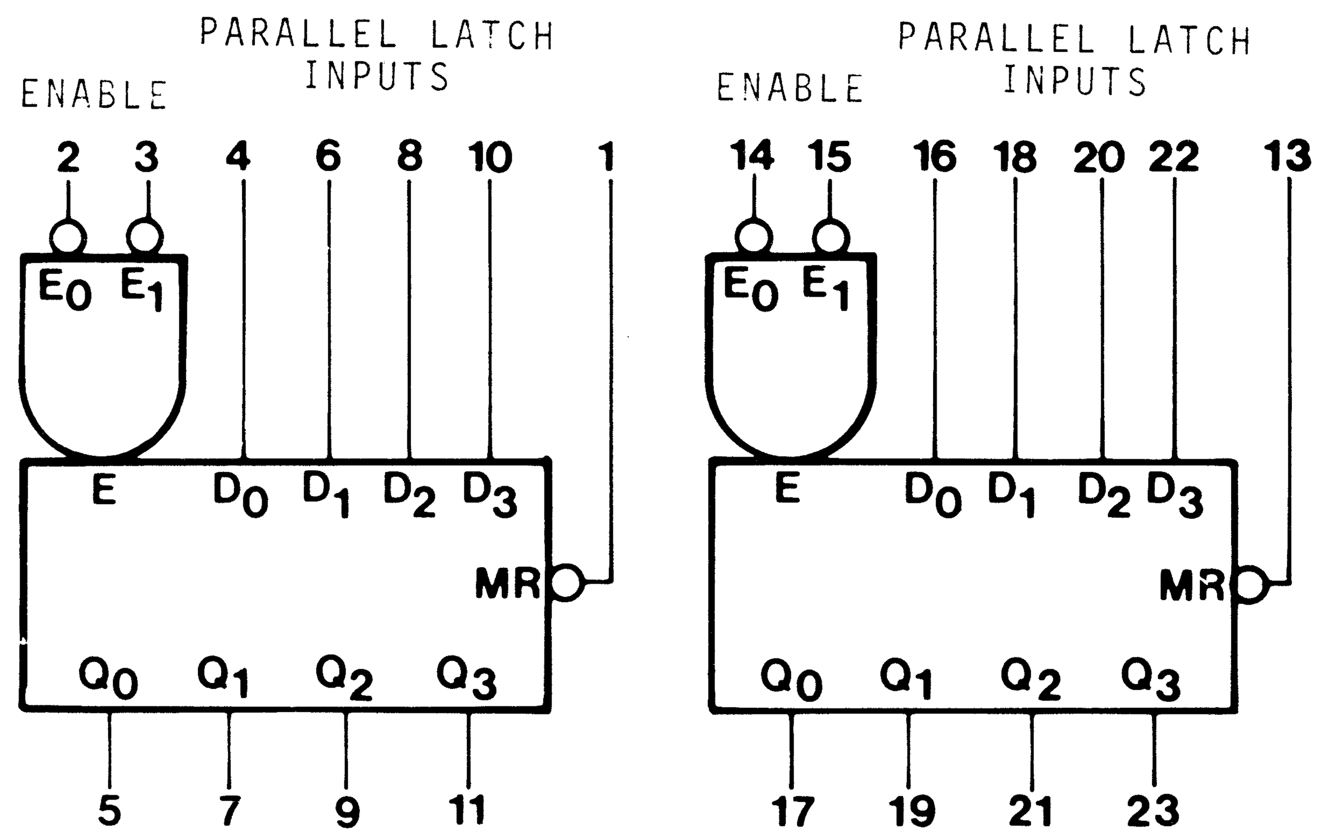


Vcc 16
GND 8

TC = $Q_0Q_1Q_2Q_3CEP$

9308

DUAL FOUR-BIT LATCH



FUNCTION TABLE

MR	E0	E1	D	Qn	OPERATION
H	L	L	L	L	Data Entry
H	L	L	H	H	Data Entry
H	L	H	X	Qn-1	Hold
H	H	L	X	Qn-1	Hold
H	H	H	X	Qn-1	Hold
L	X	X	X	L	Reset

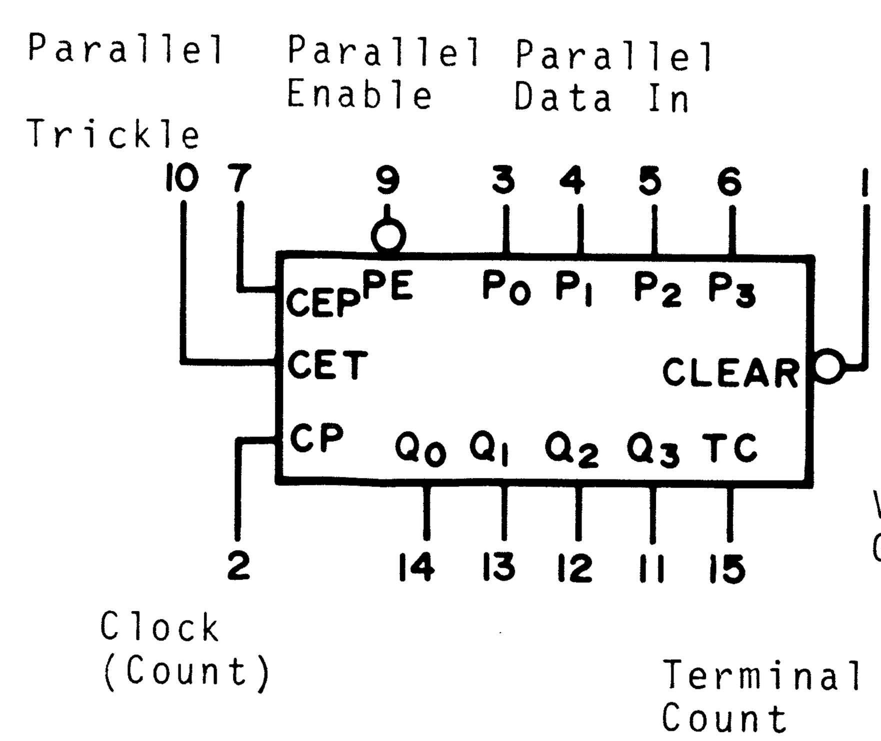
Vcc 24
GND 12

Qn-1 = Previous Output State
Qn = Present Output State

9310

9310, 74161

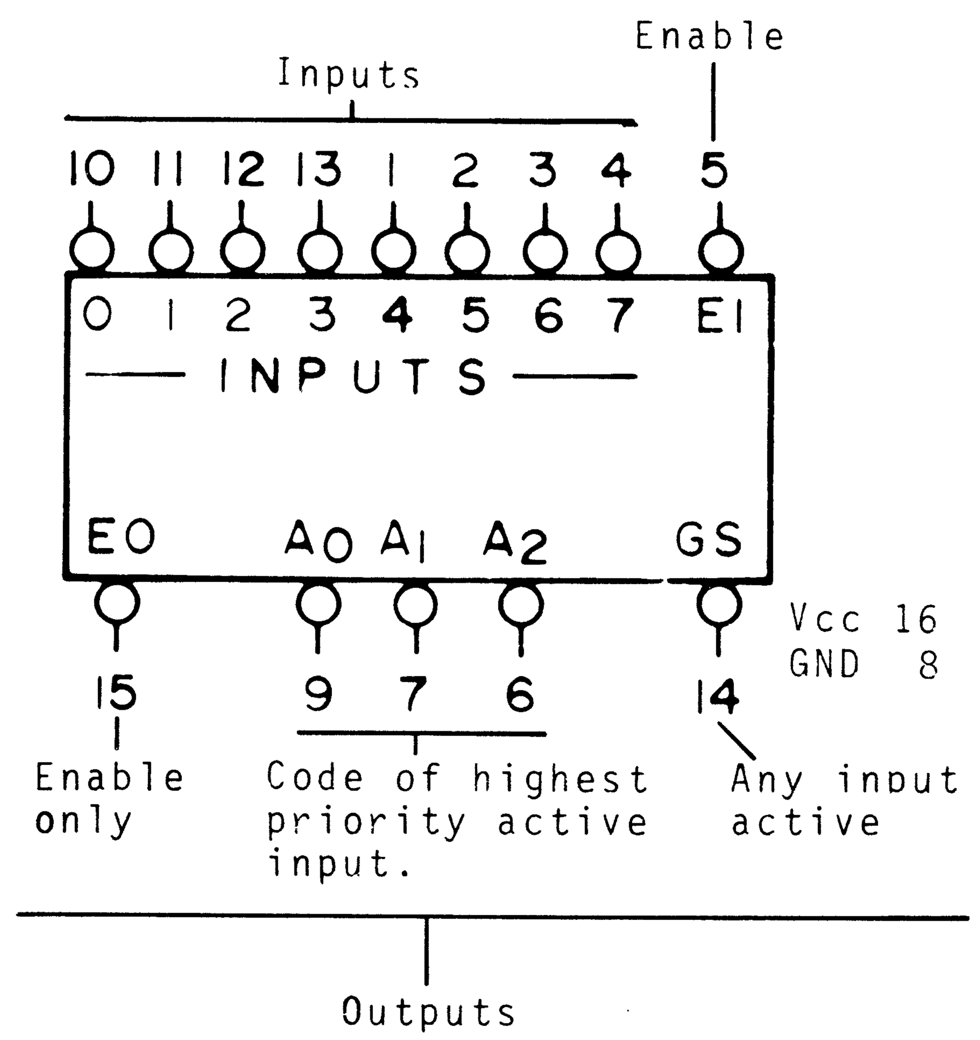
BCD COUNTER
BINARY COUNTER



Vcc 16
GND 8

9318

EIGHT-INPUT PRIORITY ENCODER

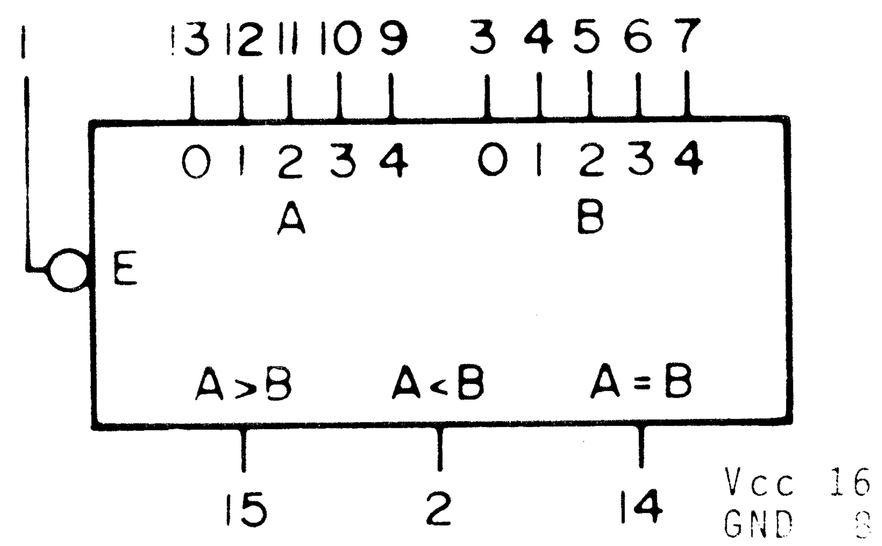


Vcc 16
GND 8

9318 is a Multipurpose Encoder disigned to accept eight inputs and produce a binary weighted code of the highest order input.

9324

5-BIT COMPARATOR



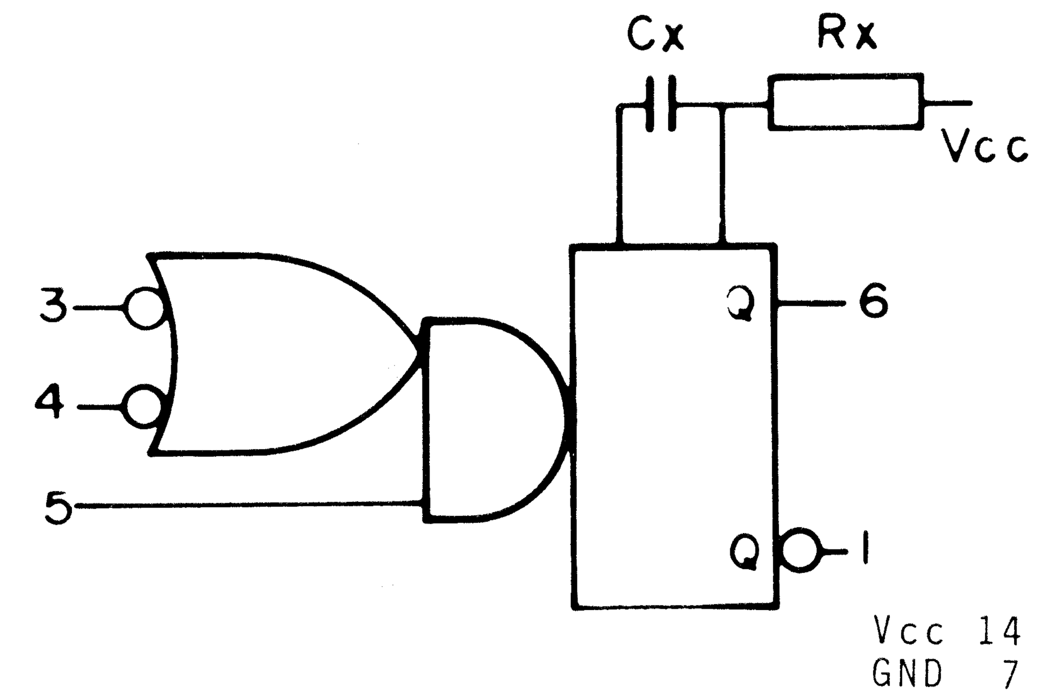
A High Speed Expandable Comparator that provides comparison between two 5-bit words and gives three outputs, "less than", "greator than" and "equal to". A HIGH level on the active LOW enable input forces all three outputs LOW.

93403

(see 7489)

9603

SINGLE SHOT WITH SCHMITT TRIGGER INPUT



FUNCTION TABLE						
t_n INPUT		→	t_{n+1} INPUT			OUTPUT
L	X	L	L	X	H	Trigger
X	L	L	X	L	H	Trigger
H	H	H	X	L	H	Trigger
H	H	H	L	X	H	Trigger

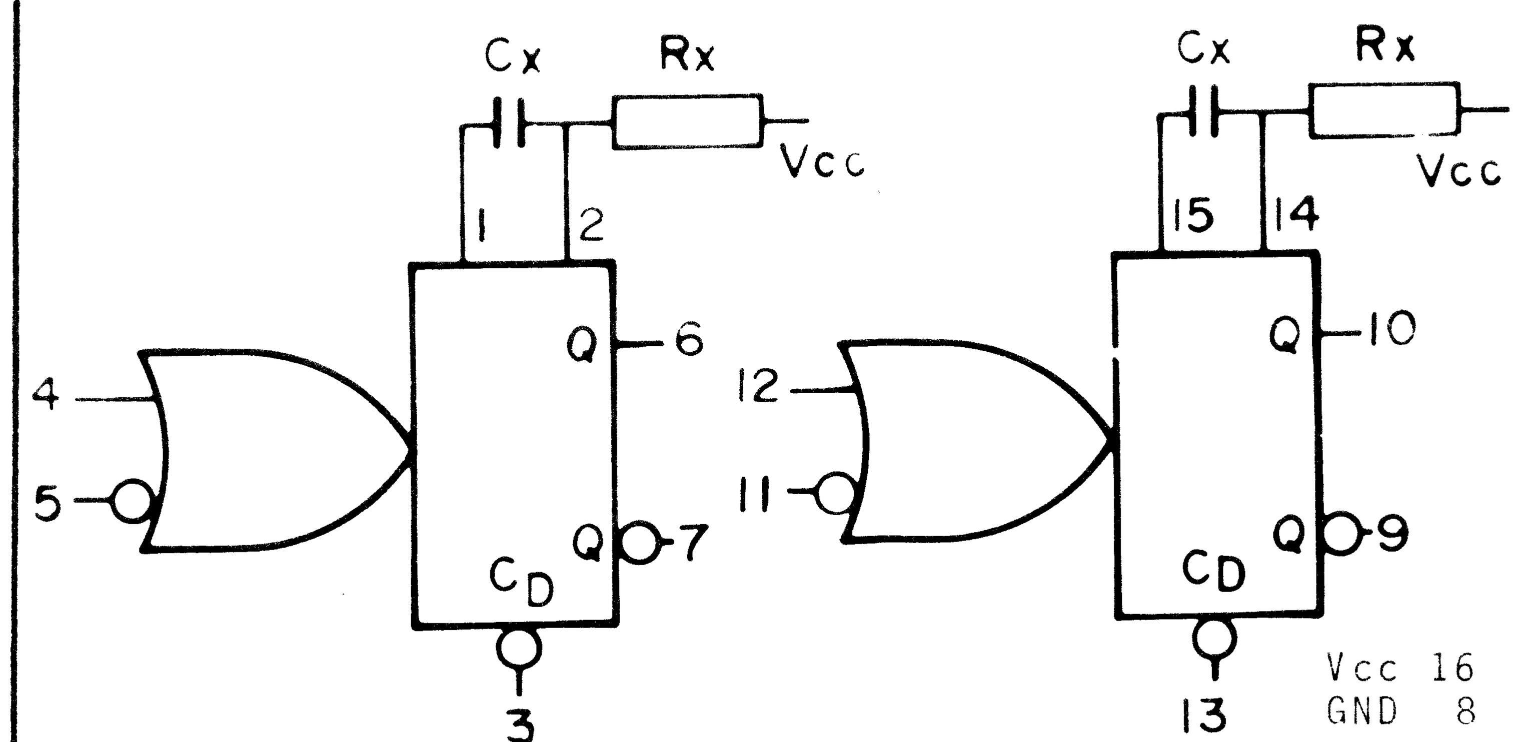
→ = Triggering Transition

9341

(see 74181)

9602

SINGLE SHOT



TRIGGERING FUNCTION TABLE			
PIN NO'S 5(11)	4(12)	3(13)	Operation
H-L	L	H	Trigger
H	L-H	H	Trigger
X	X	L	Reset

Dual Retriggerable, Resettable Monostable Multivibrator provides an output pulse whose duration and accuracy is a function of external timing components Ex and Rx.

ADL2011
CR202A

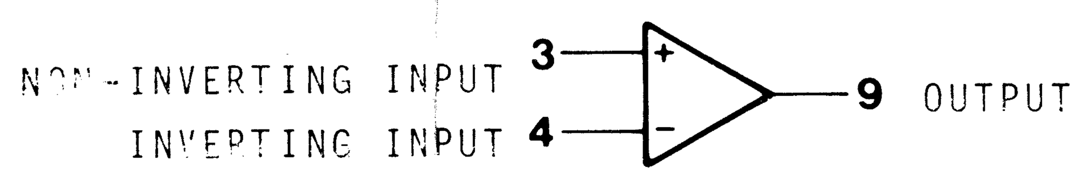
(see 74188)

REC 0613

(see 7404)

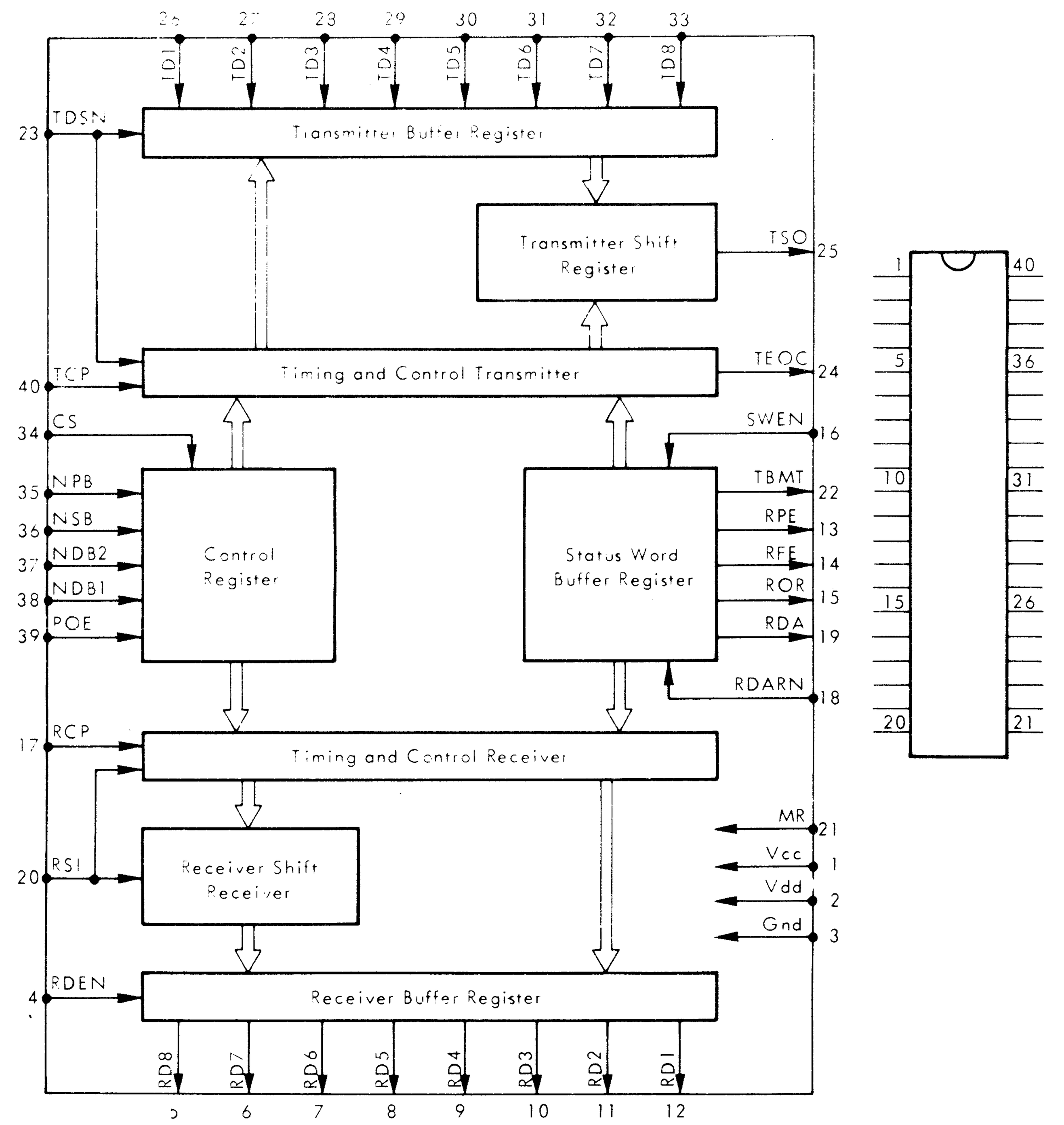
µA710DC

HIGH SPEED DIFFERENTIAL COMPARATOR



GND 2
+V 11
-V 6

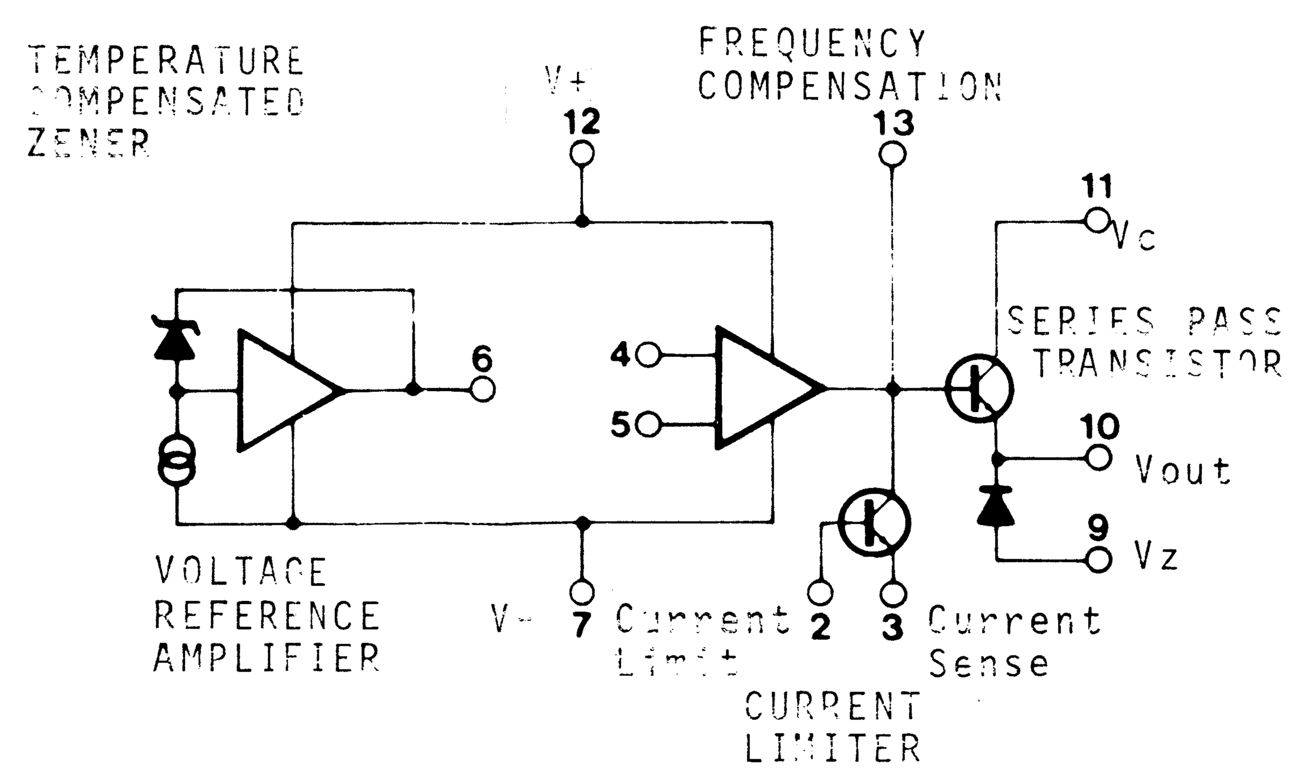
UART



µA 723

µA723

PRECISION VOLTAGE REGULATOR



APPENDIX B

I/O PROCESSOR

SERVICE MANUAL

Effective Pages February 1977

ii	through	iii
B.1	through	B-27

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

APPENDIX B

I/O PROCESSOR

B.1 GENERAL

The Input/Output Processor (IOP channel (Figure B-1) manages data transfers directly between memory and up to eight multiplexed control-unit/device channels. The IOP contains a pair of address/length control-word registers for each of its eight channels. At the beginning of a transfer for one device, the CPU program uses two WER instructions to load this register pair with the starting address and the block length (Figure B-2). The IOP logic then provides all GP Bus timing signals to control the data transfers directly between the memory and the CU.

B.2 For input or output data transfers, a CU signals that it is ready with a Break Request (BR) to the IOP. The IOP makes a Bus Request to obtain control of the GP Bus. The IOP then sends a simulated INR or OTR command to the CU to initiate one word transfer. The INR/OTR command is simulated in that it is generated by the IOP and is not a CPU programmed instruction. The IOP logic updates its control-word register for each data word. When the block length is counted down to One, the IOP sends End of Record (EOR) to the CU along with the last simulated INR/OTR data transfer. The data block transfer is ended with an SST command and status transfer between the CU and CPU.

B.3 The IOP is provided in two versions:

- version with card marked MX is for P852 only.
- version with card marked IOP is for P852/856/857.

The function and the logic is identical for both versions. The two cards differ in layout and in the location of components. Two sets of logic diagrams are thus provided to show the component locations for each version.

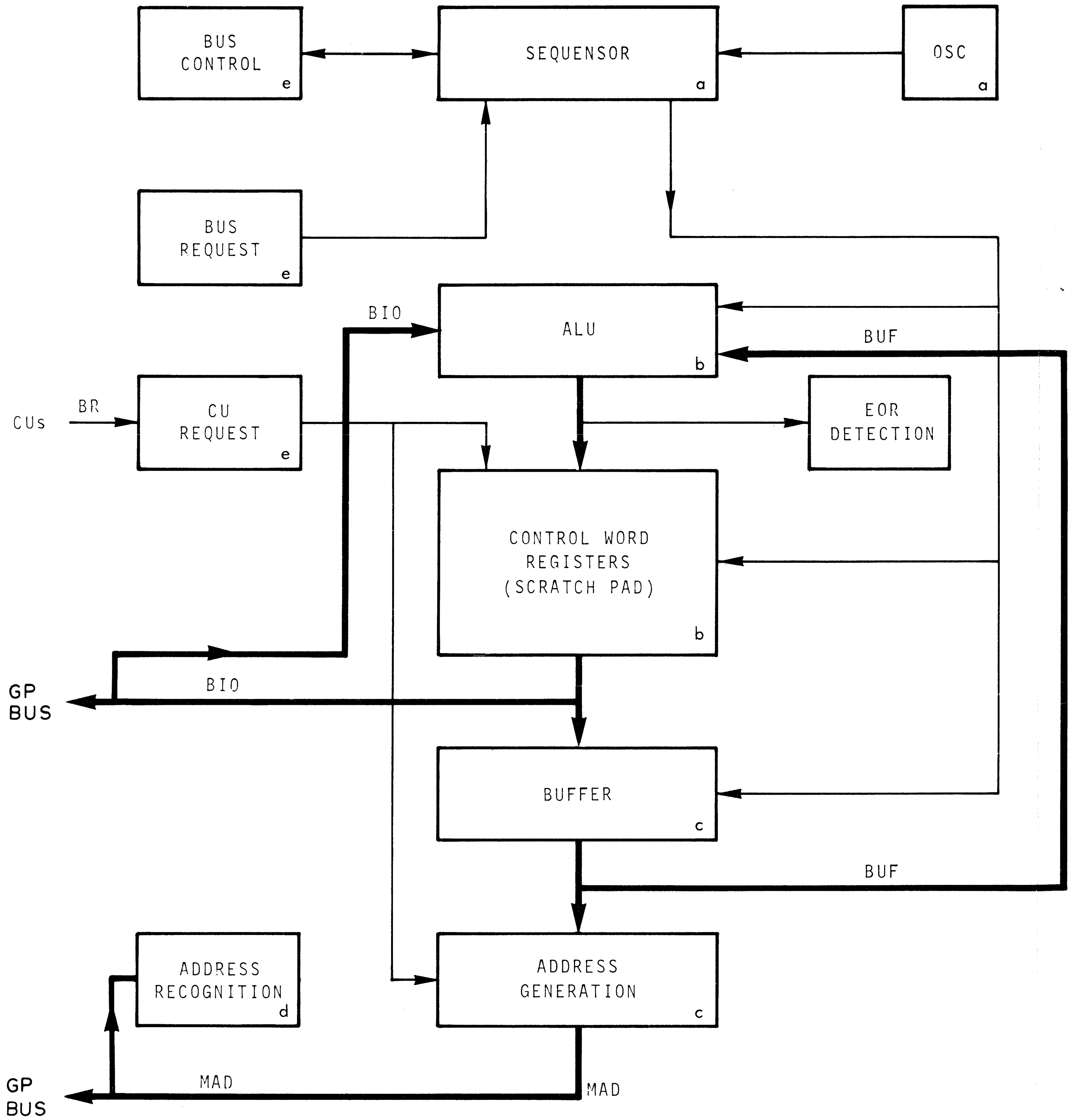


Figure B-1 I/O Processor Block Diagram

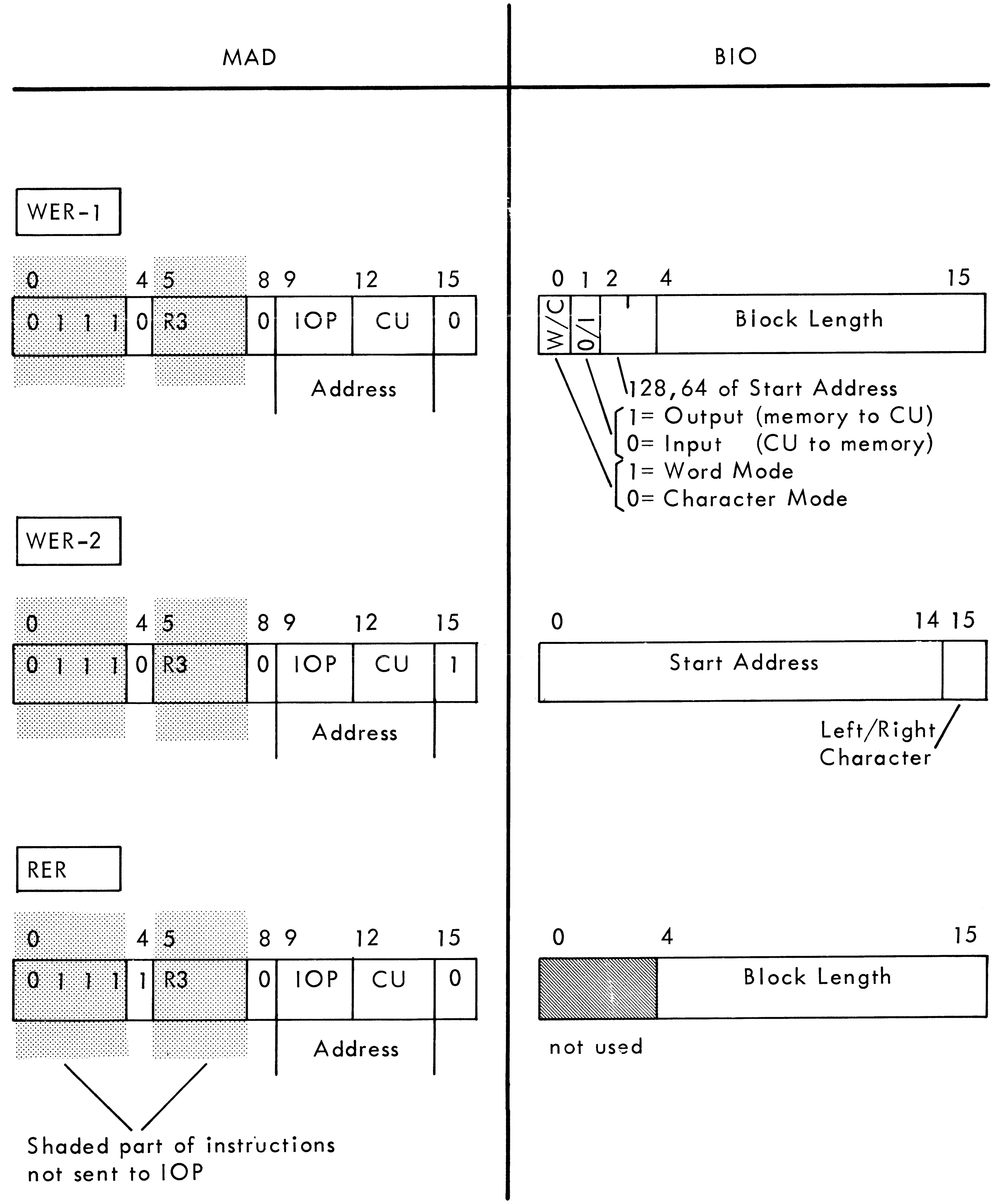


Figure B-1 I/O Processor Block Diagram

Figure B-2 Bus Content During CPU Mode (WER/RER)

B.4 CU/Device Priority

Priority of the eight CU/Device channels is established by the connection of the Break Request lines. Refer to paragraph B.18.

B.5 WER Instruction

Two Write External Register (WER) instructions (Figure B-2) are used to load the two control words for a device into a pair of IOP registers. Bits 04, 08-15 of the instructions are sent to the IOP on the MAD lines; bit-15 of each instruction specifies WER-1 or WER-2. The R3 field of each instruction specifies a CPU accumulator (A1-A7) that contains the block-length or start-address control word which is sent to the IOP on the BIO lines.

B.6 RER Instruction

A single Read External Register (RER) instruction (Figure B-2) is used when the CPU wants to test the remaining block length of an incomplete data-transfer operation. Bits 04, 08-15 of the instruction are sent to the IOP on the MAD lines; bit 15 = 0 accesses the control-word-1 (block length) register in the IOP. The R3 field of the instruction specifies the CPU accumulator (A1-A7) where the control-word information placed on the BIO lines is to be loaded (bits 4-15).

B.7 Logic Description and Diagrams

The IOP logic is described in the sequence of its operation, in the section IOP Operating Modes. Operation of some of the more complex logic units (sequensor, ALU, scratchpad) is given also in the section Functional Units. Logic diagrams (Figure B-9, sheets a-e) are provided at the end of the logic description. These diagrams are referenced on the block diagram and in the text by the sheet number, for example: "logic c" refers to Figure B-9, sheet c.

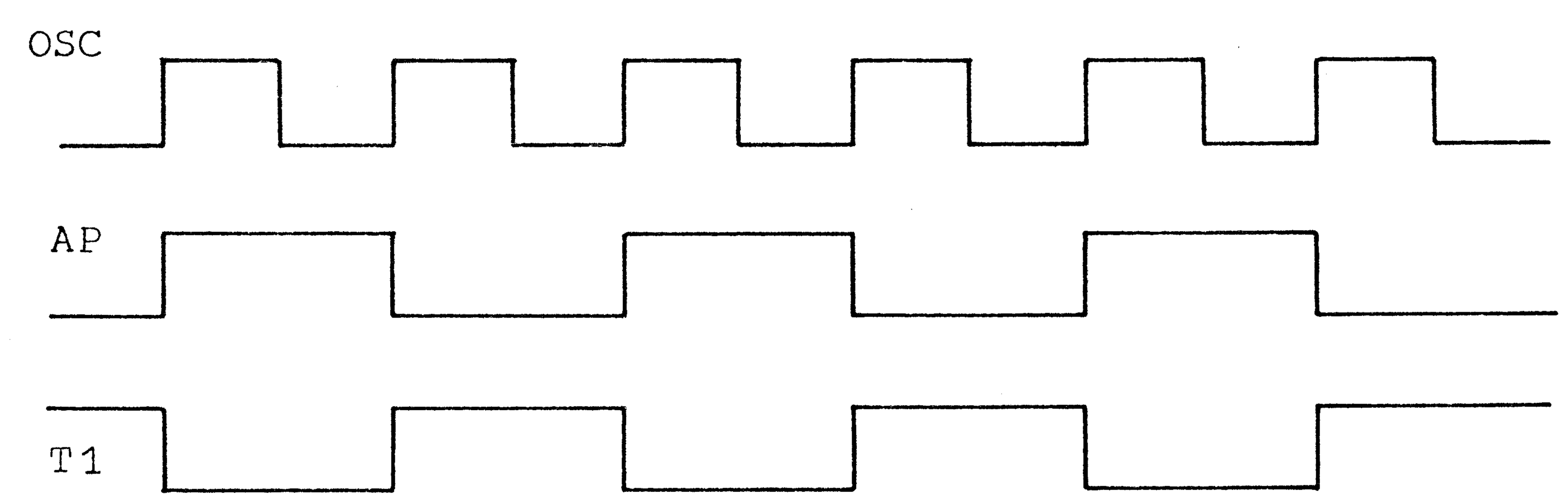
B.8 IOP OPERATING MODES

The IOP operates in three modes:

- Sampling Mode -- The IOP sequensor logic monitors the IOP status and tests for CU Break Requests or CPU commands.
- CPU Mode -- Set upon receipt of a CPU command (WER or RER instruction). Control-word register information is transferred between the CPU and IOP, with the CPU as master to control the GP BUS.
- Exchange Mode -- Set upon receipt of a Break Request from a CU (with Bus Obtained. One data word is transferred directly between the CU and memory, with the IOP as master to control the GP Bus.

B.9 Sampling Mode

The IOP waits in Sampling Mode whenever there is no instruction or data-exchange operation being performed. The sequensor logic is set to the Scan cycle (Figure B-3); timing signals AP and T1 are repeated continuously. The AP signal is used to test every 200ns for instructions from the CPU and Break Requests from the Control Units. Detection of the address-recognized signal AK indicates a CPU command, and the IOP sets CPU mode with signal NCPU. Detection of a Break Request sets the IOP to exchange mode. The ENB flip-flop (set at the end of an Exchange operation) remains set during Sampling Mode (logic e) to enable a Bus request if a BR is detected.



(BFYBS) AP. ENB
(NCPUZ1) AP. AK.$\overline{NCPU}$
(NCW1Z1) AP. MSL.BSYL. $\overline{TMP}$

asynchr : (BUSRZ1) BR
(MSLZ1) BUSR. OKI
(BSYLZ1) MSL. $\overline{BSY}$. $\overline{TRM}$. $\overline{TPM}$

Figure B-3 Scan-Cycle Timing

B.10 CPU Mode

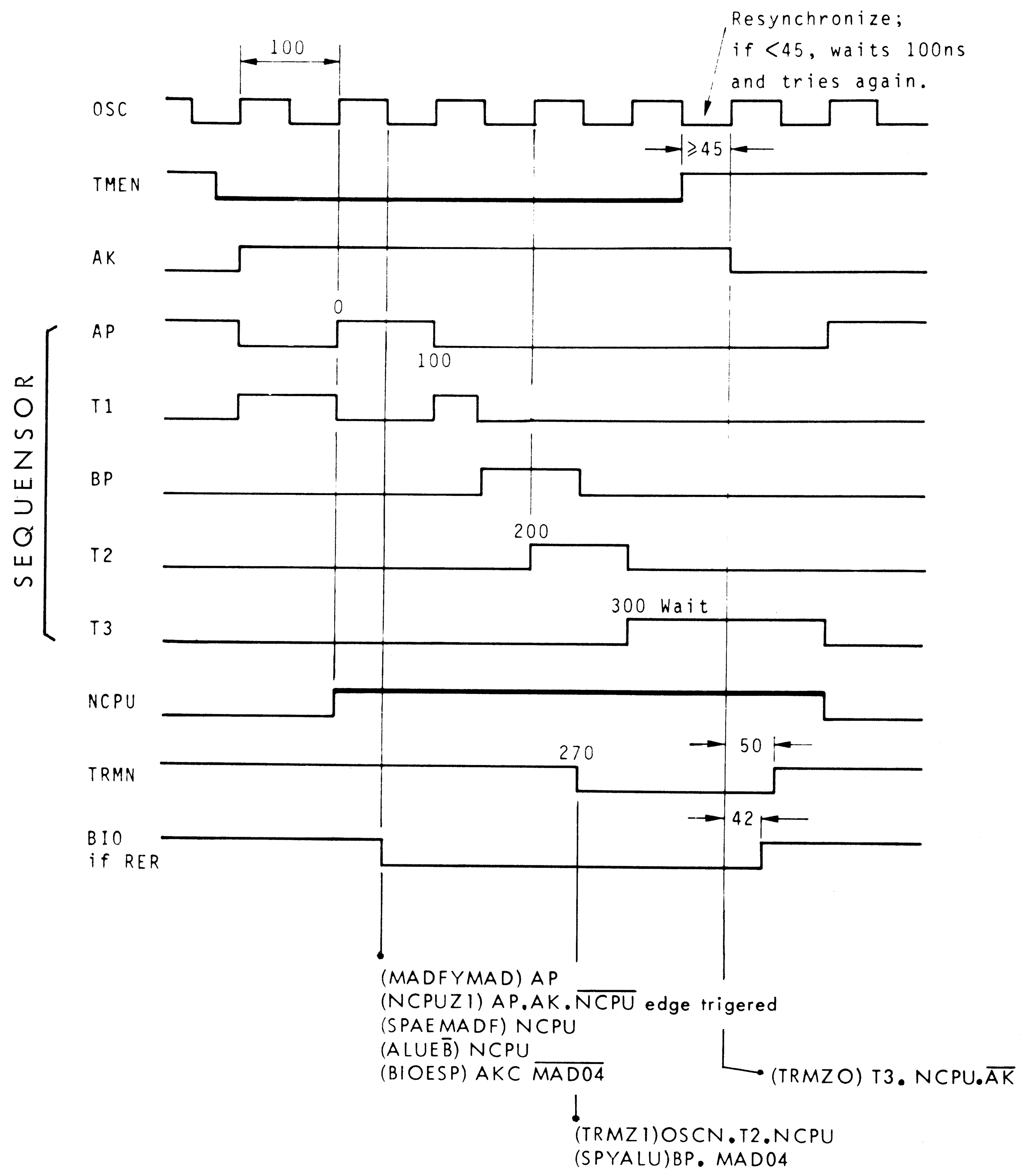
The CPU mode is used by the IOP logic to perform the WER or RER instruction from the CPU. The CPU sends part of the instruction word (including the IOP address) to the IOP (Figure B-2), with timing signal TMEN. The CPU mode is set when the IOP, in Sampling Mode, detects its address-recognized signal AK.

B.11 Addressing. A WER/RER instruction can address up to 256 external registers with MAD 08-15 (Figure B-2). With bit 08 = 0 for IOP operations, up to 128 external registers can be specified, with 16 registers for each IOP. Bits 12-14 address the register pair for a specific CU/Device channel; bit 15 indicates register-1 or register-2 for the selected device, corresponding to the first or second control word for the channel.

B.12 CPU-Cycle Operation. CPU-cycle timing is shown in Figure B-4. The IOP compares the address on MAD08-11 with its own address code set by U-links (logic d). An address compare (AKC) is set into the AKCF flip-flop on the rising edge of OSC. The AK signal sets flip-flop NCPU on the rising edge of AP (logic d), if NCPU is not already set. This constraint avoids repeating a CPU cycle. NCPU and AK are used by the Sequensor CPU-cycle: AP-T1-BP-T2-T3.

B.13 For an RER instruction (MAD04 = 1), AK activates BIOVALN (logic d) to gate the control-word register contents onto the BIO lines to the CPU (Scratchpad, logic b). The active RCWN signal inhibits writing into the scratchpad.

B.14 For a WER instruction (MAD04 = 0), BIOVALN and RCWN are blocked (logic d) and WCW1N is conditioned. The Arithmetic unit (ALU, logic b) is set to logic operation $\bar{B}$ by the selection signals CW2, 1 = 1, 0. The control word on the BIO lines is thus connected through the ALU B-input to the scratchpad. The control word is clocked into the addressed scratchpad register by the BP pulse during the CPU-cycle timing.



Note : Times shown in ns

Figure B-4 CPU-Cycle Timing

B.15 The scratchpad-address multiplexer (SPA0-3N) is switched by NCPU so that the MAD lines 12-15 select the register address. In the middle of T2 time, TRMX is set and the timing-response signal TRMN is sent back to the CPU (logic e).

B.16 The IOP waits with sequensor-cycle T3 until TMEN from the CPU is terminated. AK is reset on the next OSC after TMEN drops. The loss of AK activates APJN (logic a) which resets TRMX, dropping TRMN, and enables AP to be set on the next OSC. The CPU Mode is finished when AP is set and the IOP is switched to the Sampling Mode.

B.17 Exchange Mode

The Exchange Mode is used by the IOP logic to perform a data transfer between a CU and memory. The exchange can be either input (CU to memory) or output (memory to CU). The IOP operates as System Master to obtain control of the GP Bus and control the operation. The operation is performed in two logic sequensor cycles: CW1 and CW2. The Exchange Mode is set when the IOP, in Sampling Mode, detects a Break Request (BR) from one of the CUs (Figure B-5).

B.18 Break Requests. The CU priority is established when the Break Request (BR) lines from the CUs are connected to the IOP, with BR00 the highest priority and BR07 the lowest. The BR lines are examined whenever Enable flip-flop ENB is set (logic e). ENB is reset at the start of an Exchange operation (T1 of CW1, Figure B-6) to prevent a higher-priority BR from altering the conditions after an operation has started. ENB is set near the end of the Exchange, and remains set during the Sampling Mode, to enable a BR to initiate a Bus-Request sequence.

B.19 A Bus Request is initiated if any BR is active when ENB is set. This may happen any time during Sampling Mode, or in the middle of the previous Exchange cycle CW2. Starting a new Bus Request before the previous Exchange has been completed enables data-exchange operations to be linked together to save time.

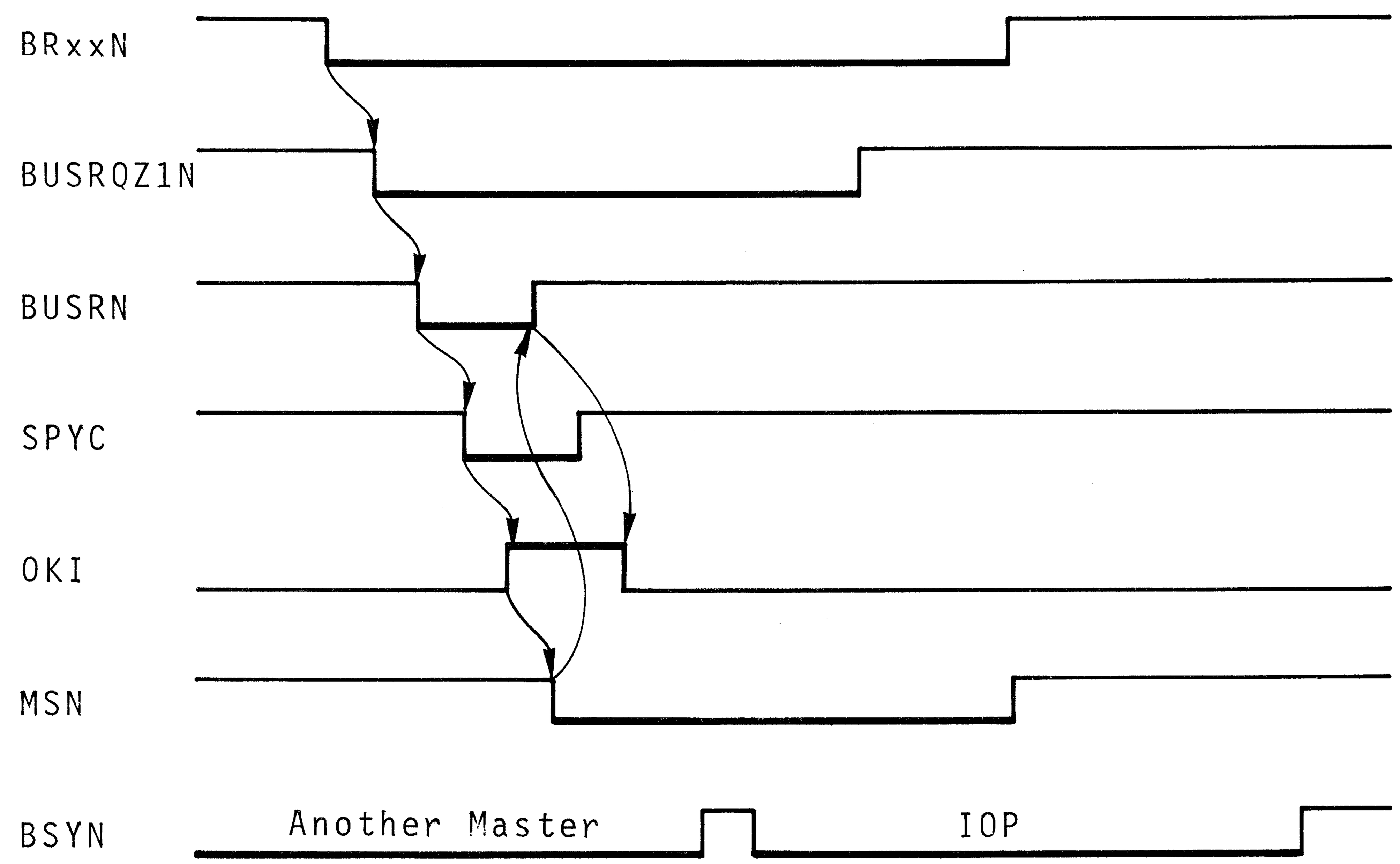


Figure B-5 Bus Request Sequence

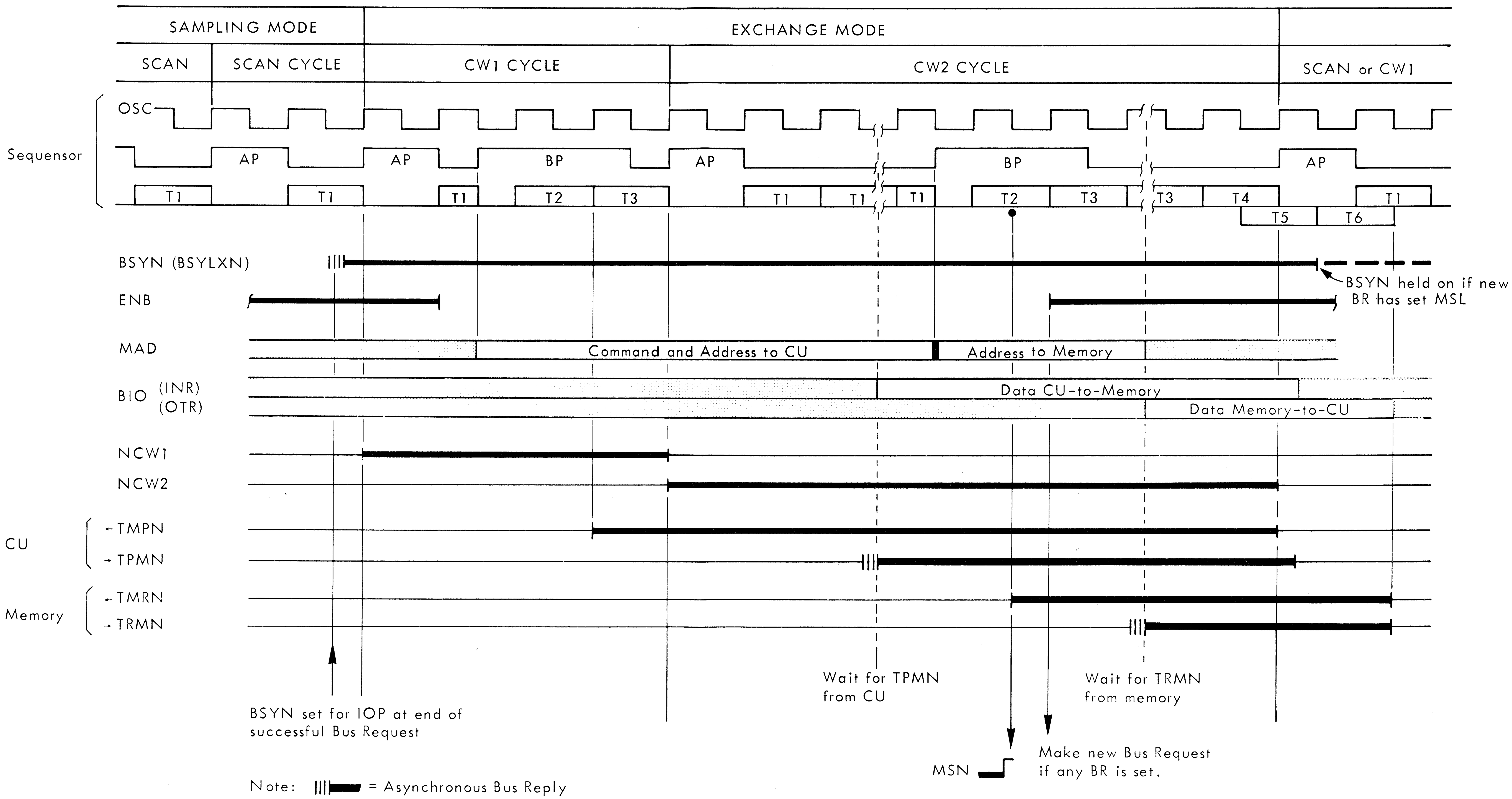


Figure B-6 Exchange-Cycle Timing (CW1/CW2)

B.20 With ENB set, any active BR signal is set into the FBR register (logic e) by the leading edge of AP, which occurs every 200ns during Sampling Mode, or at the end of Exchange cycle CW2. The priority encoder circuit (74148) indicates the code of the highest-priority BR stored in FBR. This code (BRENC0-2) is used to select the external-register (scratchpad) address.

B.21 Bus Request Logic. The Bus Control sequence, with the IOP as Master, is shown in Figure B-5. BUSRQZ1N from the detected BR is gated into the Bus-Request logic (logic e) as BUSRQ if no other operation is active (BUSRQN high). BUSRQ sends BUSRN to request the GP Bus.

B.22 The active-low SPYC response (scan priority chain) sets the OKA flip-flop. OKI is received if no higher-priority unit takes control of the Bus. OKI sets the MSL (master selected local) flip-flop, while OKAN blocks the sending of OKO to the next unit on the Bus. MSL sends the Master-Selected signal MSN onto the Bus, and terminates BUSRN. The CPU responds to the end of BUSRN by dropping SPYC and OKI.

B.23 The IOP may have to wait, with MSL set, until the Bus becomes free of any current operation (TRMN, TPMN, and BSYN all inactive). When the Bus is free, MSL sets BSYLX which puts BSYN onto the Bus to take Bus control for the IOP. The IOP switches to the CW1 cycle at the first AP after BSYLX is set.

B.24 Register Addressing. Two control-word registers must be accessed in the scratchpad during the Exchange Mode. The registers contain data transfer information (control and address) and must be updated during the operation. The scratchpad-address multiplexer (logic b) is switched by $\overline{\text{NCPU}}$ so that the BR priority encoder and NCW1 select the register address. The three most-significant bits (SPA1-3) are selected by BRENC2-0N; the least-significant bit (SPA0) is selected by NCW1 (1 for cycle CW1 and 0 for cycle CW2).

B.25 First Control Word. The first control-word sequence (Figure B-6) begins with the first AP pulse following the setting of BSYLX. The sequensor CW1 cycle is AP-T1-BP-T2-T3, with AP of the CW2 cycle following directly after T3. The NCW1 flip-flop (logic d) is set by the leading edge of AP after BSYLX is set.

B.26 With NCW1 set, the ALU mode selected is A minus 1 (for bits 04-15 only). The first control word is gated from the scratchpad into the buffer register (BUF, logic c) by the leading edge of BP. The BUF contents are immediately applied to the ALU where the block-length is decremented and bits 00-03 are transferred directly. If the block length is decremented to One by this operation, ENDN is active. At the end of BP, the updated control word is gated back into the scratchpad.

B.27 The Input/Output control bit (ALU01N) and the ENDN signal are set into a two-bit register (logic b) as BUFIN and BUFEOR. These two bits are written into the auxiliary scratchpad as SPINN and SPEORN, and are used to minimize delay in the generation of the MAD signals to the CU.

B.28 The IOP command to the CU is placed on the MAD lines (Figure B-7) via the MADSL multiplexer (logic c). The Input/Output control (SPIN) and EOR (SPEOR) bits are sent on MAD04 and 03 respectively. The IOP address (MXAD0-2) placed on MAD10-12 is taken directly from the IOP address U-links. The CU address from the priority encoder (BRENC0-2) is sent on MAD13-15.

B.29 The IOP generates Bus timing signal TMPN (logic e) at the end of time T2 (the first OSC after flip-flop TMPX is set). TMPN goes active 160ns after the MAD lines are set. The CU uses TMPN to validate the address and control information on the MAD lines. The CW1 cycle for the first control word is completed at the end of T3, before the CU has responded to TMPN.

B.30 Second Control Word. The second control-word sequence (Figure B-6) begins with the AP pulse immediately following T3 of CW1. The sequensor CW2 cycle is AP-T1* -BP-T2-T3* -T4-T5-T6, with a waiting loop before BP and before T4. Flip-flop NCW2 (logic d) is conditioned by the TMPN signal (TMPXA) and set on the leading edge of AP.

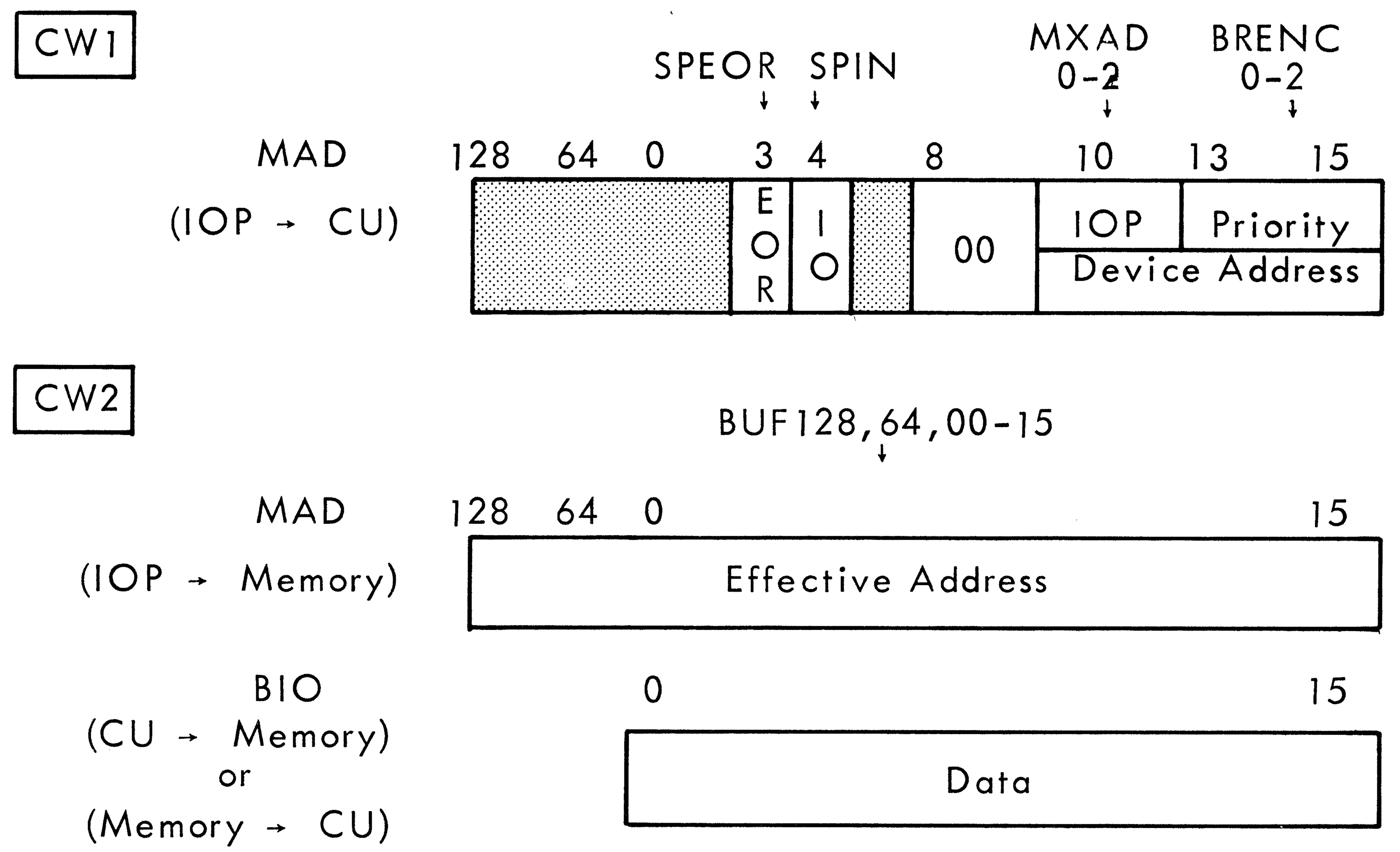


Figure B-7 Bus Content During Exchange Mode (CW1/CW2)

B.31 The second control word is available at the scratchpad output (logic b) as soon as NCW1 goes off. With the control flip-flops NCW1-NCW2 at 0-1, the ALU mode selected is A plus 1 for incrementing the effective character address. To realize A plus 2 for updating the effective word address, CW2WN forces a carry-in to bit 15, resulting in the least-significant bit set to 0 for the even-character word addressing.

B.32 The IOP waits for the CU response TPMN, with the sequensor blocked at T1 before the generation of BP. The CU receives TMPN and its address and command on the MAD lines (sent at the end of CW1), places data on the BIO lines (for an input transfer), drops its BR for that exchange, and sends TPMN back to the IOP. TPMN is clocked into TPMNRA (logic a) on the leading edge of OSC preceeding BP to ensure the guard time of BIO for input transfers.

B.33 Before the IOP starts the memory transfer, it ensures that any preceeding memory response signal TRMN is inactive (important in the case of linked exchanges). The BP pulse gates the 18-bit effective address (control word 2) from the scratchpad into the buffer (logic c). This memory address is switched via the MADSL multiplexer onto the MAD lines. The selection signal MADSEL is set at BP of CW2. At the same time, the read/write and word/character control signals are sent to the memory via Bus signals WRITE and CHA. These two control bits were read from the first control word and stared as BUFIN (logic b) and BUFCH (logic c) during CW1.

B.34 The IOP sets TMRX (logic e) in the middle of time T2 to send TMRN to the memory. The memory uses TMRN to validate the address on MAD (from the IOP) and the data on BIO (from the CU, if input transfer). The IOP must now wait for memory response TRMN.

B.35 The Bus control flip-flops OKA and MSL (logic e) are reset as a result of TMRN dropping MSN and freeing the Bus for a new selection. The reset occurs when TMRX sets the MSLRF flip-flop, and generates MSLRN. The reset MSLN flip-flop in turn resets MSLRF. BSYN is held active until time T6 so that the IOP maintains Bus control until its operation is complete. Enable flip-flop ENB (logic e) is set on the first OSC after T2 to permit acceptance of a new BR from any of the CUs.

B.36 If any BR is active when ENB is set, a new Bus Request sequence is initiated (paragraph B-20). If OKI is received without being blocked by another unit on the Bus, another IOP Exchange cycle (CW1-CW2) is linked to the end of this exchange. MSL is again set by OKI and holds BYSN on at time T6 instead of allowing it to be reset as in a normal ending.

B.37 The IOP waits for the memory response TRMN, with the sequensor blocked at T3 before the generation of T4. The TMPN signal to the CU remains active during the memory-transfer operation to enable the BIO-line data at the CU while TMRN enables BIO at the memory.

B.38 The TRMN response from memory is set into flip-flop TRMNRA (logic a) on any leading or trailing edge of OSC. Time T4 is then set on the next leading edge of OSC to continue the CW2 cycle. At time T4, flip-flops TMPX and TMPENB (logic e) are reset and TMPN to the CU drops (providing suitable guard time for the CU to read data on the BIO lines). The CU drops TPMN after TMPN goes inactive.

B.39 For read operations (output, memory to CU): the memory validates the data on the BIO lines with TRMN, the IOP validates the BIO data at the CU with the trailing edge of TMPN. For write operations (input, CU to memory): the CU validates the BIO-line data with TPMN, the IOP validates the BIO data at the memory with the leading edge of TMRN.

B.40 FUNCTIONAL UNITS

A block diagram of the IOP is given in Figure B-1. The control words are loaded into the Control Word Registers (scratchpad), via the ALU, by WER instructions. The CPU reads a Control Word from the scratchpad by means of an RER instruction. During data-transfer operations (Exchange Mode), the Control Words are accessed and updated via the processing loop: scratchpad--buffer--ALU--scratchpad. The address-generation logic sends the CU address/command and then the memory address for each data transfer.

B.41 Sequensor

The sequensor (logic a) is driven from a constantly-running oscillator to provide the IOP timing signals AP, BP, T1, T2, T3, T4, T5, T6. The oscillator frequency at QUARTZ is 20 MHz and OSC is 10 MHz. Sequensor cycles for the different Operating Modes are:

Operating Mode	Seq. Cycle	
Sampling	Scan	AP-T1, repeated until CPU or Exchange Mode is set.
CPU	CPU	AP-T1-BP-T2-T3*, waiting at T3 until the first OSC after TMEN is received from CPU.
Exchange	CW1	AP-T1-BP-T2-T3, followed by the CW2 cycle.
Exchange	CW2	AP-T1*-BP-T2-T3*-T4-T5-T6, with waiting loop before BP and T4. AP of next cycle (Scan/CW1) follows T4, overlapping T5-T6.

Signals AP, T1-T4, T6 are clocked on the rising edge of OSC. Signals BP and T5 are clocked on the falling edge of OSC and thus shifted one-half OSC cycle from the other sequensor signals. All sequensor timing signals have a duration of one or more 100ns OSC cycles, as follows:

- AP, T2, T4, T5, T6 are always 100ns.
- BP is 100ns (CPU cycle) or 200ns (CW1/CW2 cycles).
- T1 or T3 may last for one or more multiples of 100ns during the waiting conditions for BP or T4 during the CW2 cycle.

B.42 The conditions for the Sequensor timing signals are:

- AP (first pulse of every cycle)

Condition	
Scan T1	Start of Scan or CW1 cycle, depending on conditions for BP.
CPU T3.AKN	Start of Scan cycle; Sequensor waits at T3 of CPU cycle until AKN indicates the end of the CPU command, to avoid branching back into a CPU cycle if TMEN is delayed.
CW1 T3	Start of CW2 cycle always follows T3 of CW1.
CW2 T4	Start of Scan or new CW1 cycle follows T4, although T5-T6 of CW2 are not complete.

- T1

AP	T1 follows AP for all cycles
$\overline{BP}$	Wait at T1 until BP is set (CW2 cycle)

- BP

Cycle	Selection S0 NCW1	Selection S1 NCPU		Set (BPJ)	Reset (BPK)	
CW2	0	0	10	TPMNRA	T3	Wait at T1 until TPMN received
CW1	1	0	11	1	T3	
CPU	1	1	12	1	T2	

BPJ enabled by BPJE2N: SCANN.T1.(CW2+TRMN.TMPENB)

- T2 follows BP (CPU, CW1, CW2 cycles)
- T3 follows T2 and stays on until AP (CPU, CW1 cycles) or T4 (CW2 cycle) is set.
- T4 follows T3 directly (CPU, CW1 cycles) or waits for receipt of TRMN (CW2 cycle).
- T5 (CW2 cycle) ends the command to the CU.
- T6 (CW2 cycle) ensures data guard time for ending the memory transfer.

B.43 Arithmetic Unit (ALU)

The ALU (logic b) is used to load control words from the CPU and to update the control words during Exchange Mode. The type 9341 or 74181 ALU circuits operate in one of three modes, controlled by flip-flops NCW1-NCW2, as follows:

Operating Mode	Control NCW1,2 = CWIN,2		Selection S3,2,1,0-CIN-CE	Operation
CPU	0 0	1 0	L H L H L H	$\overline{B}$
Exch-CW1	1 0	0 0	L L L L L L	A - 1*
Exch-CW2	0 1	1 1	H H H H H H	A + 1

* Decrement bits 04-15 only; carry-in to bit 03 blocked by $\overline{NCW2}$ so that bits 03-00 are transferred without change.

- CPU mode loads the WER control words via ALU operand B from the BIO lines to the scratchpad.
- Exchange-Mode/CW1-cycle decrements the block-length control word in the processing loop, via ALU operand A.
- Exchange-Mode/CW2-cycle increments the effective-address control word in the processing loop, via ALU operand A.

B.44 Scratchpad

The 16-word scratchpad (logic b) stores the two control-words for each of the eight Control Units (Figure B-8).

B.45 Scratchpad Addressing. A type 74157 multiplexer circuit provides the scratchpad-addressing signals SPA0-3N, as follows:

Operating Mode	S-Input	Address Source for SPA0-3N
CPU	NCPU	MAD15-12RF from CPU (MAD lines); active low.
Exch-CW1	$\overline{NCPU}$	NCW1, BRENC2-0N; control-word-1 specified by the CU priority-level code.
Exch-CW2	$\overline{NCPU}$	$\overline{NCW1}$, BRENC2-0N; control-word-2 specified by the CU priority-level code.

B.45 128-64, EOR Bits. A type 74157 selector (SPIN) is used to transfer some bits from one control word to the other. The SPIN selector is controlled by signal SPA0N, as follows:

Cycle	Generation of SPA0N
CPU	MAD15 = 0, MAD15RF-high, SPA0N-high: first control word.
CPU	MAD15 = 1, MAD15RF-low, SPA0N-low: second control word.
CW1	NCW1-high, SPA0N-high: first control word.
CW2	NCW1-low, SPA0N-low: second control word.

SPIN Selection	Source IC	Source IB	Source IA
SPA0N-high	ENDN	ALU03N	ALU02N
SPA0N-low	ALU02N	ALU64N	ALU128N
Output:	SPIN02	SPIN64	SPIN128

- During the first WER instruction (SPA0N-high), address bits 128-64 are loaded via SPIN into scratchpad bits 128-64 for the second control word.
- During Exchange-Mode/CW1-cycle (SPA0N-high), the decremented block length is tested for a count of ONE, and the resulting ENDN bit is loaded via SPIN into scratchpad bit 02.

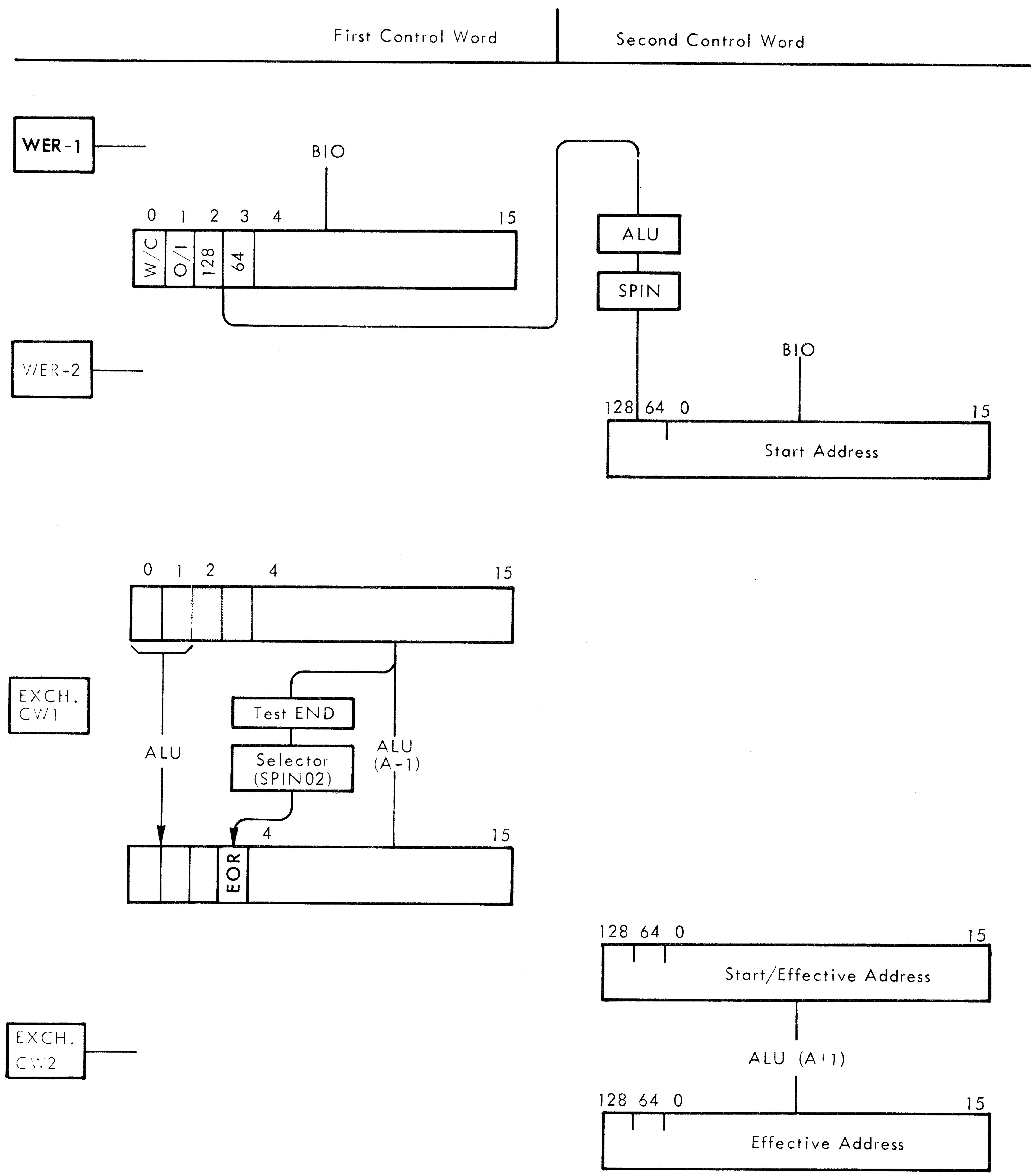


Figure B-8 Scratchpad Contents (IOP)

- During operations with the second control word (SPA0N-low), the effective address is loaded from BIO (via ALU $\overline{B}$) or incremented through the processing loop (via ALU A + 1).

B.47 In/Out, EOR bits. The Input/Output and End-of-Record (EOR) control bits are sent to the CU on the MAD lines during Exchange Mode, cycle CW1 (Figure B-7). These two bits are obtained via the ALU during the processing loop of the CW1 cycle: the In/Out bit from ALU01N; the EOR bit from testing block-length bits 04-15 for One (ENDN). ALU01N and ENDN are gated via the type 7475 circuit (as BUFIN and BUFEOR) and loaded into auxiliary scratchpad as SPINN and SPEORN. These two command bits are gated onto MAD04 and 03 along with the control-unit address code during the CW1 cycle (which is held until the TPMN response during the CW2 cycle).

B.48 Other Logic Units

The remaining logic units are described with the IOP Operating Modes. The following list is a guide to the uses of the different logic units.

Logic		Text Paragraph	
Address Recognition	d	CPU-Cycle	B.12
CU Request logic	e	Exch/Break Requests	B.18
Bus Request logic	e	Exch/Bus Request Exch/2nd CW	B.21 B.35
Execution Cycle logic	d	Exch/1st CW Exch/2nd CW	B.25 B.30
Bus Control logic	e	Exch/1st CW Exch/2nd CW	B.29 B.34
Buffer	c	Exch/2nd CW	B.32

B.49 PHYSICAL

The Units are contained on printed-circuit cards (Figure B-10) of the standard P852/856/857 system size. The parts lists are provided in Table B-1. The IOP interfaces the other system elements (CPU, control units, and memory) via the GP Bus, via connector-3.

B.50 U-Links

Adjustable U-links are provided for decoding the IOP address (logic d). The locations of the U-links are shown on Figure B-10.

B.51 Break-Request (BR) Signals

The BR signal connections are shown in Figure B-11. The internal BR signals, from CUs in the same chassis with the IOP, are attached to connector-4B. The external BR signals, from CUs in the extension chassis, are attached to connector 5, and are strapped from connector-4A to 4-B. Priority of the control units is determined by the order of the connections to connector-4B.

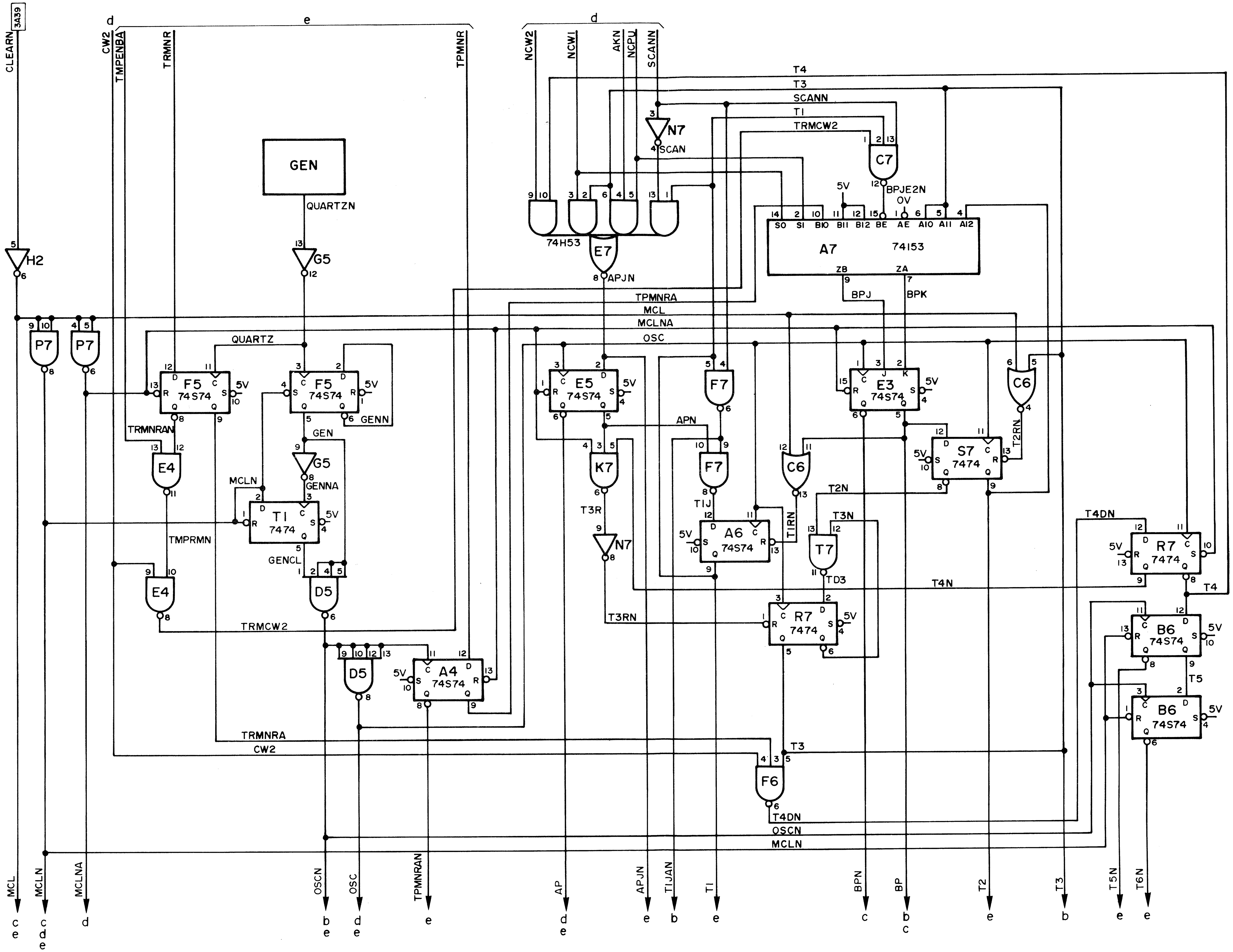


Figure B-9a Clock/Sequensor (MX)

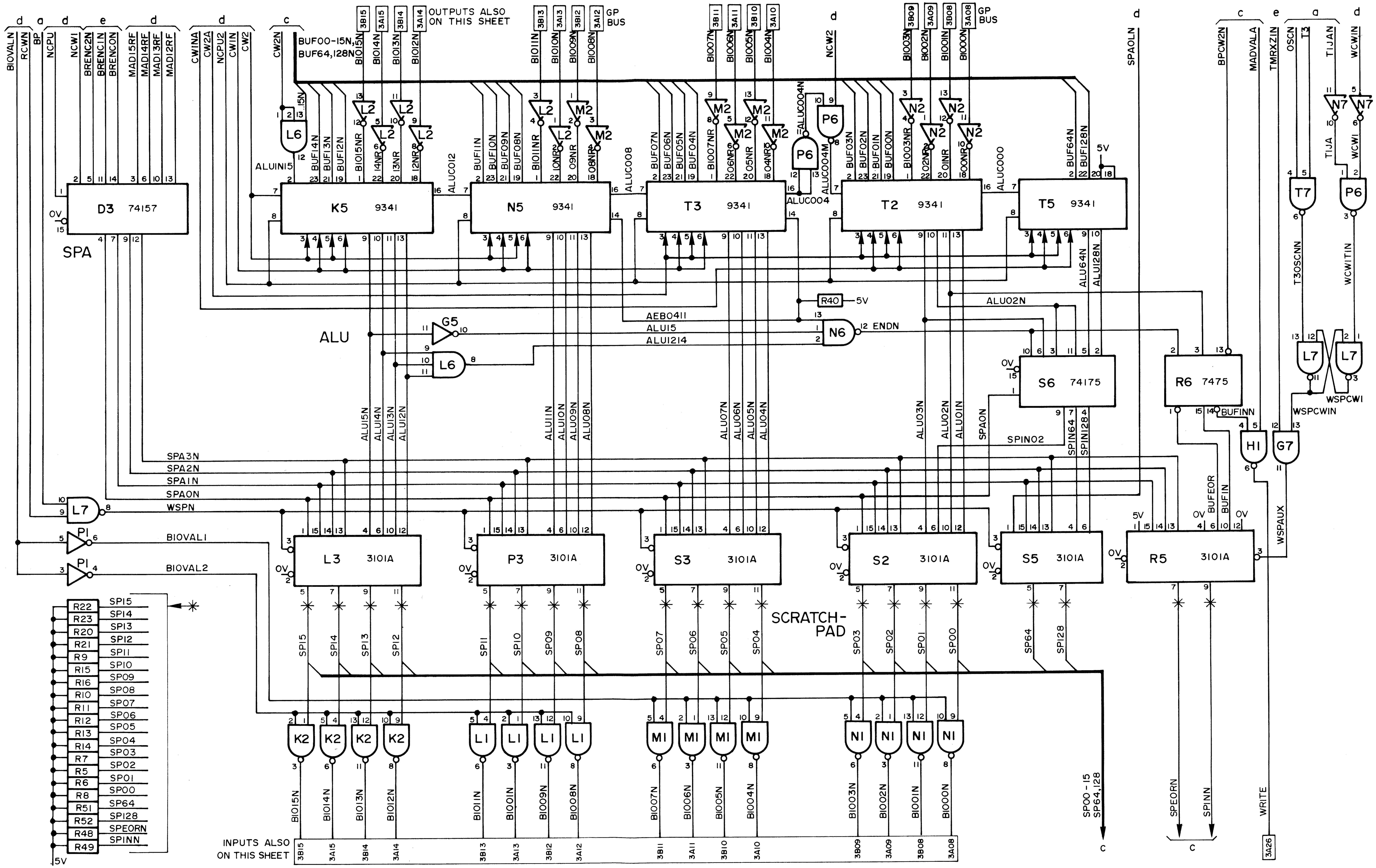


Figure B-9b ALU, Scratchpad (MX)

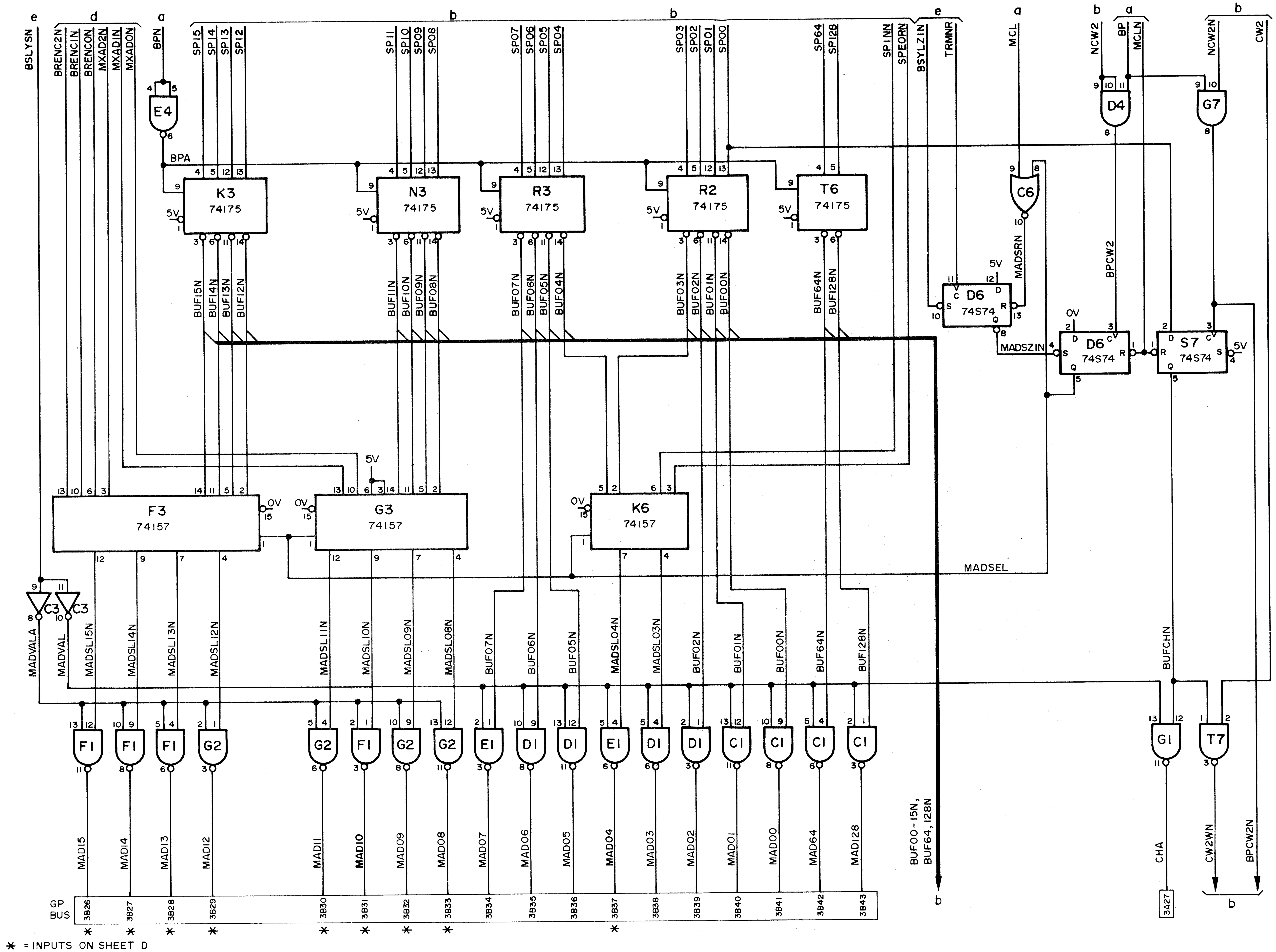


Figure B-9c Buffer, Address Generation (MX)

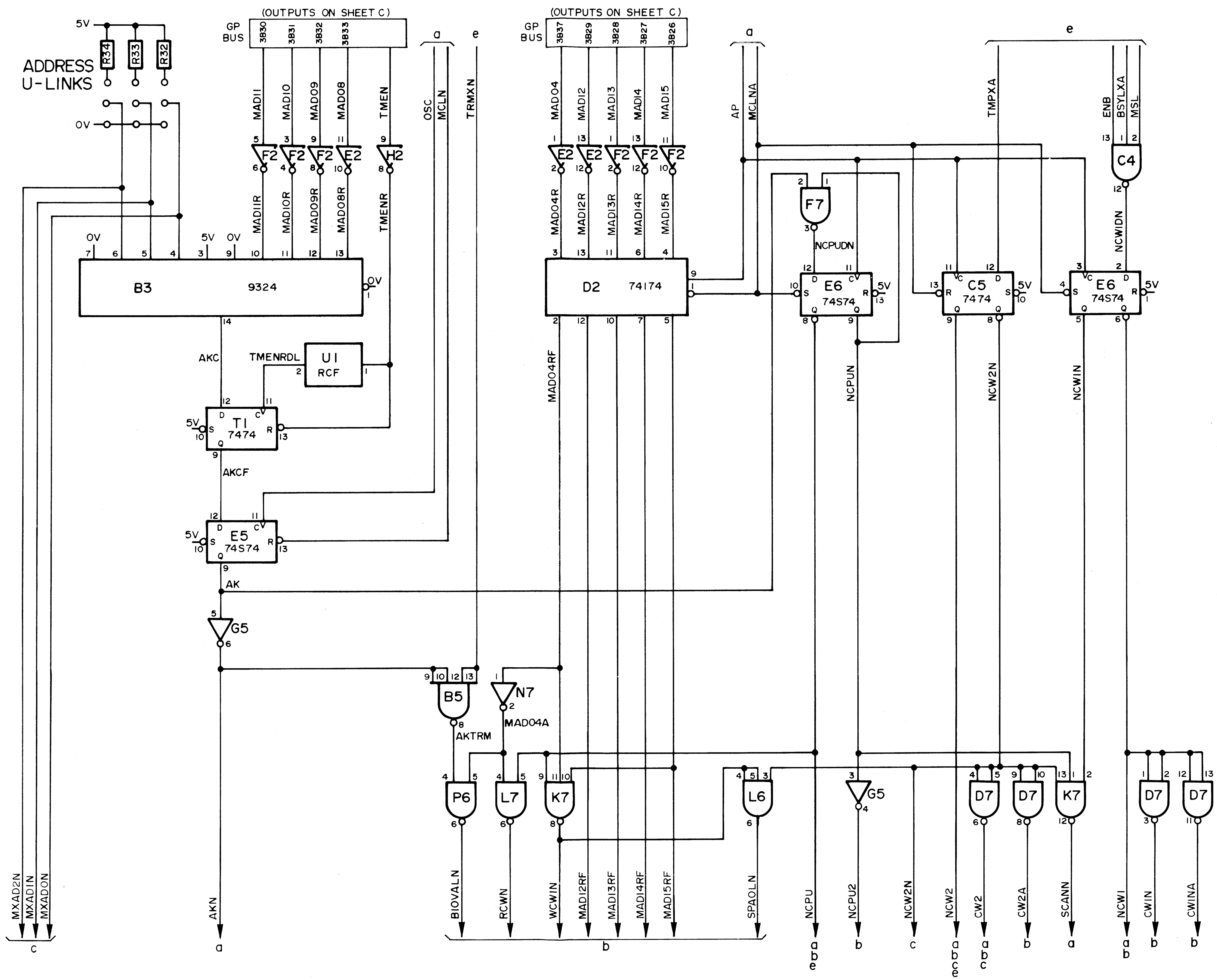


Figure B-9d Address Recognition, Execution Cycle (MX)

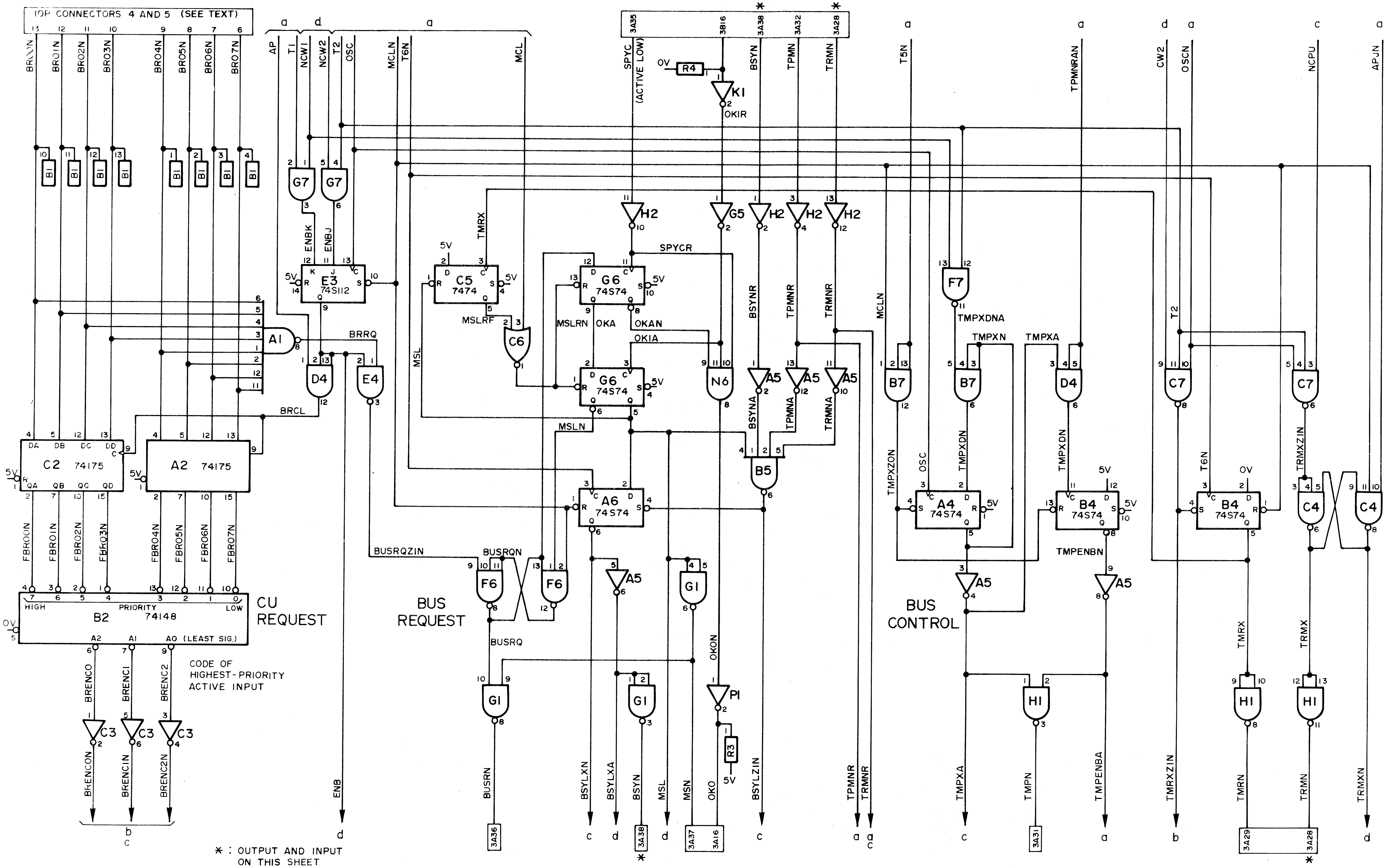


Figure B-9e Input/Output Control (MX)

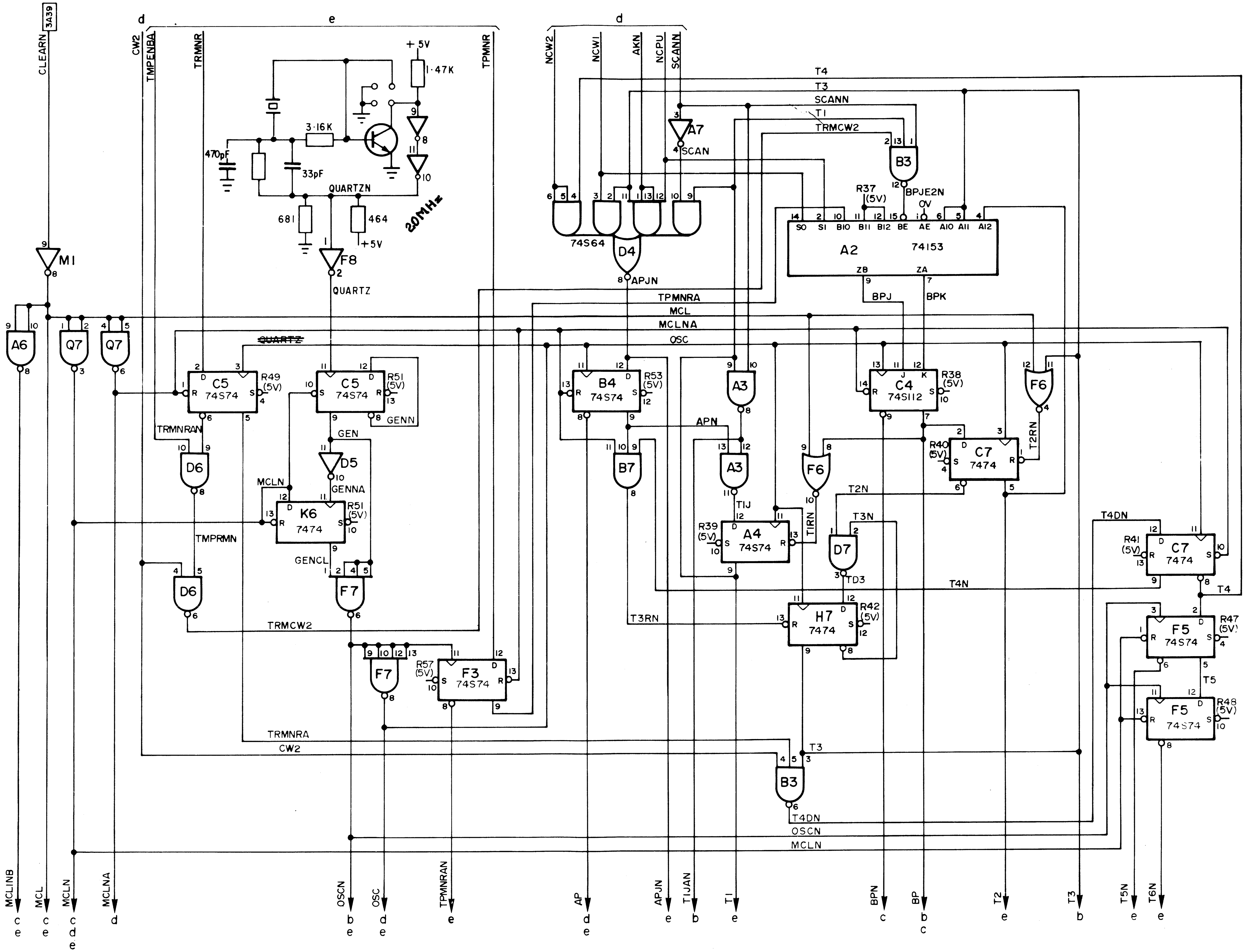


Figure B-9a Clock/Sequensor (IOP)

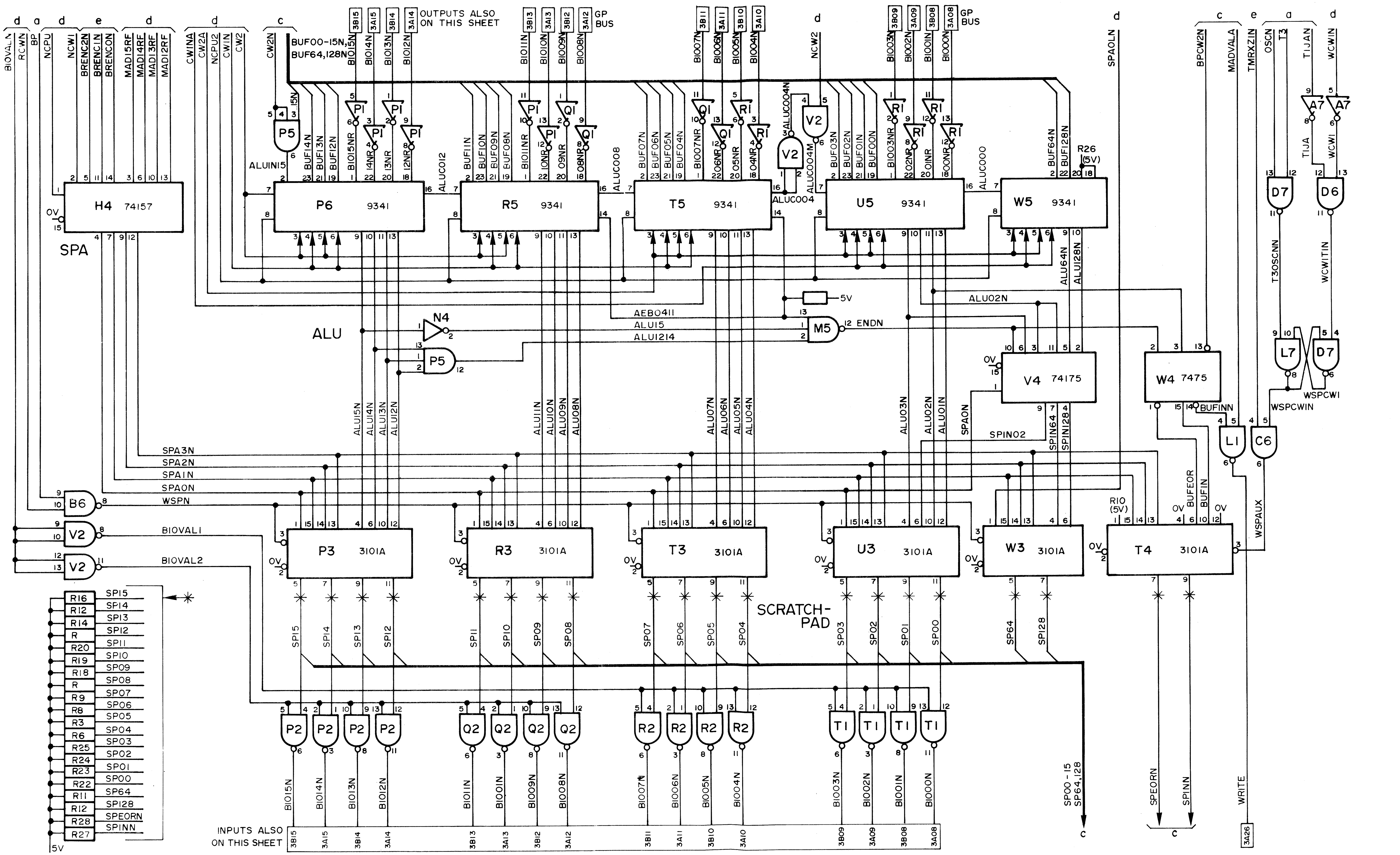


Figure B-9b ALU, Scratchpad (IOP)

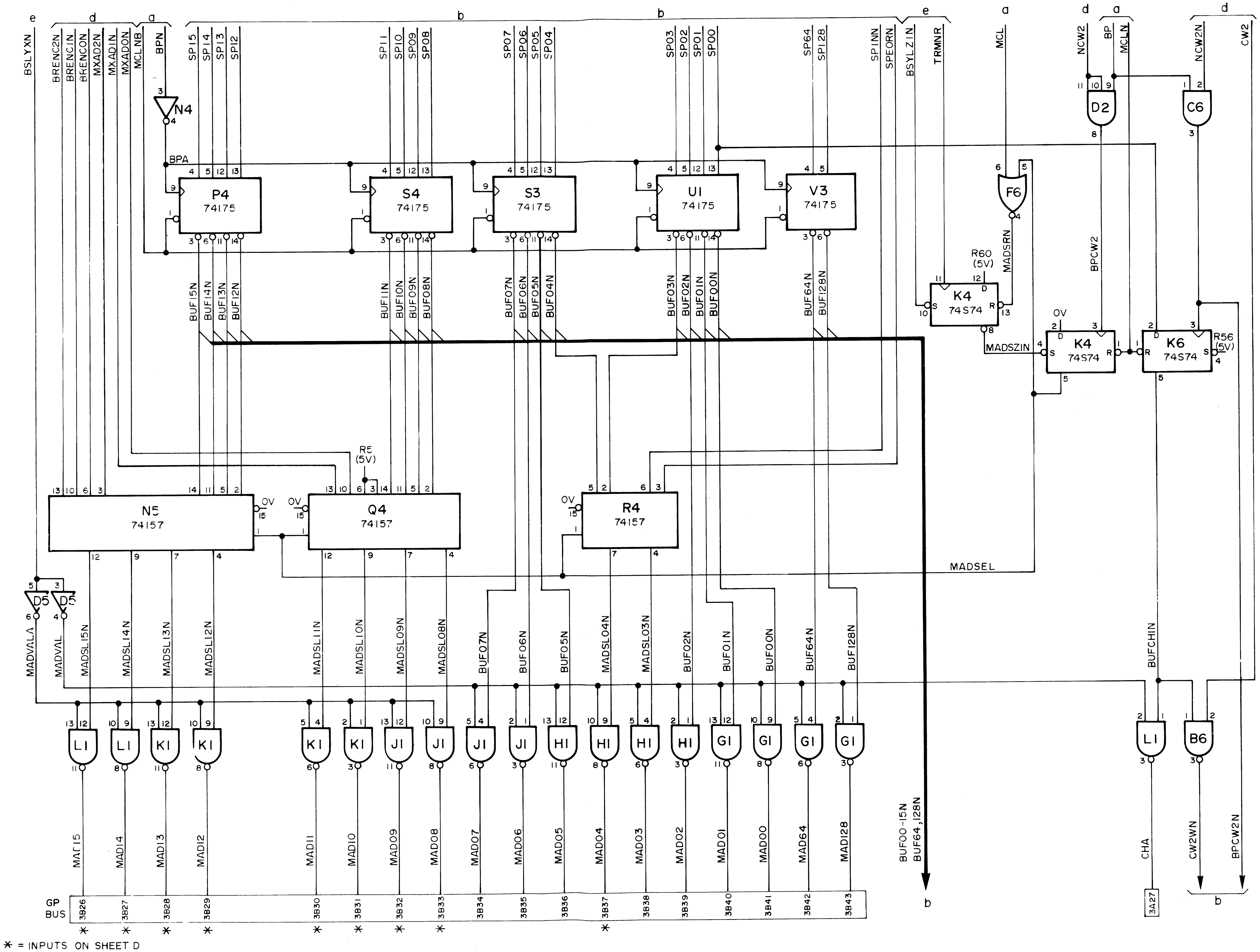


Figure B-9c Buffer, Address Generation (IOP)

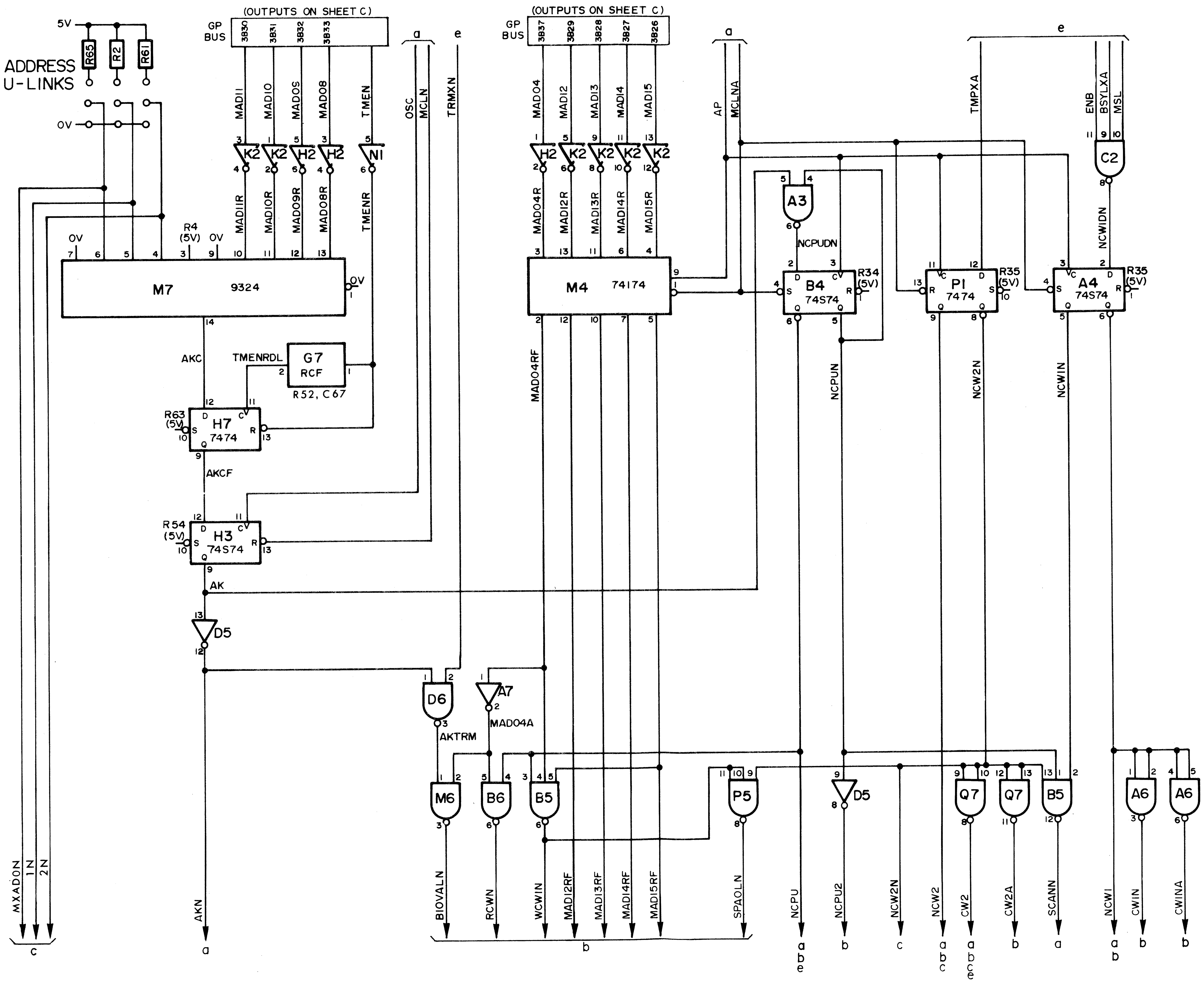


Figure B-9d Address Recognition, Execution Cycle (IOP)

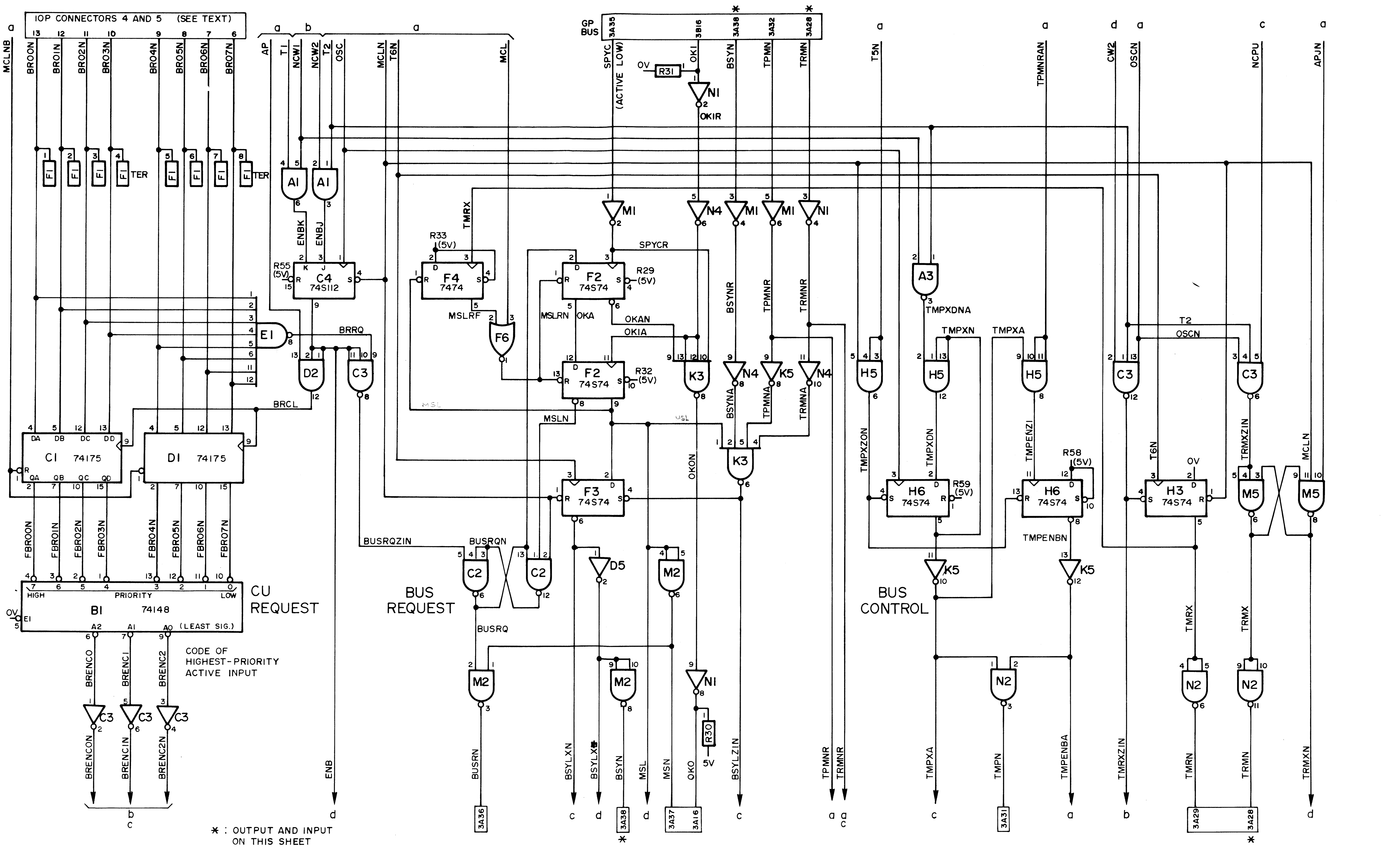


Figure B-9e Input/Output Control (IOP)

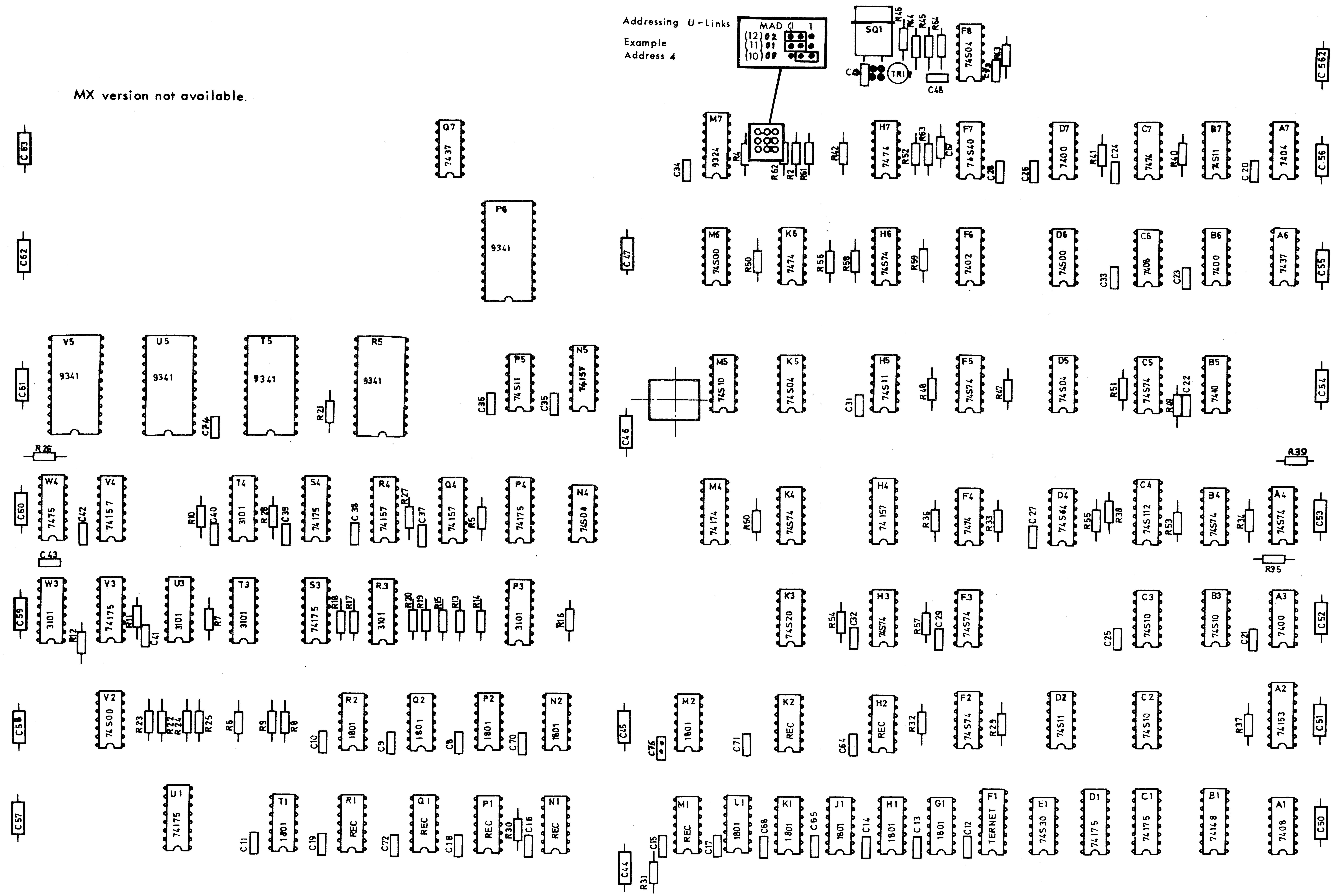


Figure B-10 I/O Processor Layout (IOP)

Table B-1a Multiplex Card Parts List

Reference	Description	12NC Code
	Printed circuit	5111 100 05473
L3, P3, R5, S2, S3, S5.	Integrated circuit 3101A	
F7, L7, T7.	Integrated circuit 7400	
C6.	Integrated circuit 7402	
N7.	Integrated circuit 7404	
G7.	Integrated circuit 7408	
K7.	Integrated circuit 7410	
D7, P7.	Integrated circuit 7437	
C5, R7, S7, T1.	Integrated circuit 7474	
R6.	Integrated circuit 7475	
B2.	Integrated circuit 74148	
A7.	Integrated circuit 74153	
D3, F3, G3, K6, S6.	Integrated circuit 74157	
D2.	Integrated circuit 74174	
A2, C2, K3, N3, R2, R3, T6.	Integrated circuit 74175	
B3.	Integrated circuit 9324	
K5, N5, T2, T3, T5.	Integrated circuit 74181	
C1, D1, E1, F1, G1, G2, H1, K2, L1, M1, N1.	Integrated circuit 1801	
E2, F2, H2, K1, L2, M2, N2.	Integrated circuit REC 0613	
A1.	Integrated circuit 74H30	
E7.	Integrated circuit 74H53	
E4, P6.	Integrated circuit 74S00	
A5, C3, G5, P1.	Integrated circuit 74S04	
C4, C7, F6, N6.	Integrated circuit 74S10	
B7, D4, L6.	Integrated circuit 74S11	
B5.	Integrated circuit 74S20	
D5.	Integrated circuit 74S40	
A4, A6, B4, B6, D6, E5, E6, F5, G6.	Integrated circuit 74S74	
E3.	Integrated circuit 74S112	
B1.	Ternet	
R43.	Resistor 215Ω, 1/8W, ±1%.	
R44.	Resistor 316Ω, 1/8W, ±1%.	
R45.	Resistor 2.15KΩ, 1/8W, ±1%.	
R46.	Resistor 3.16KΩ, 1/8W, ±1%.	
R3.	Resistor 330Ω, 1/4W, ±5%.	
R40.	Resistor 470Ω, 1/4W, ±5%.	
R4.	Resistor 510Ω, 1/4W, ±5%.	
R5-16, 20-23, 51, 52.	Resistor 560Ω, 1/4W, ±5%.	
R17, 24, 31-39, 47-50, 53-58.	Resistor 1KΩ, 1/4W, ±5%.	
R2, 19, 28; 29, 30, 42, 61, 62.	Resistor 3.3KΩ, 1/4W, ±5%.	
R1, 18, 25, 26, 27, 41, 59, 60.	Resistor 5.6KΩ, 1/4W, ±5%.	
R63.	Resistor 100Ω, 1/8W, ±1%.	

Table B-1a contd.

Reference	Description	12NC Code
C48.	Capacitor 68pF, 63V, 2%, ceramic.	
C2, 11, 19, 20, 22, 23, 25, 27-29, 31, 34, 35, 37, 39, 40, 43, 45-49, 51-56, 58, 61-66, 68-71, 74-80.	Capacitor 10nF, ceramic.	
C10, 38, 67.	Capacitor 68µF, 16V, CTS13.	
C1, 3-9, 12-18, 21, 24, 26, 30, 32, 33, 41, 42, 50, 57, 59, 60, 72, 73, 81.	Capacitor 3.3µF, 16V, CTS13.	
C82.	Capacitor 560pF, 10%.	
L1, L2.	Inductance.	
10.	U Link DCW06.	

Table B-1b IOP Parts List

Reference	Description	12NC Code
	Printed circuit	5111 100 06052
A3,B6,D7.	Integrated circuit 7400	
F6.	Integrated circuit 7402	
A7.	Integrated circuit 7404	
A1,C6.	Integrated circuit 7408	
B5.	Integrated circuit 7410	
A6,Q7.	Integrated circuit 7437	
C7,F4,H7,K6.	Integrated circuit 7474	
W4.	Integrated circuit 7475	
B1.	Integrated circuit 74148	
A2.	Integrated circuit 74153	
H4,N5,Q4,R4,V4.	Integrated circuit 74157	
M4.	Integrated circuit 74174	
C1,D1,P4,S3,S4,U1,V3.	Integrated circuit 74175	
P6,R5,T5,U5,V5.	Integrated circuit 74181 (9341)	
M7.	Integrated circuit 9324	
P3,R3,T3,T4,U3,W3.	Integrated circuit 3101A	
G1,H1,J1,K1,L1,M2,N2,P2,Q2,R2,T1.	Integrated circuit 1801	
H2,K2,M1,N1,P1,Q1,R1.	Integrated circuit REC 0613	
D6,M6,V2.	Integrated circuit 74S00	
D5,F8,K5,N4.	Integrated circuit 74S04	
B3,C2,C3,M5.	Integrated circuit 74S10	
B7,D2,H5,P5.	Integrated circuit 74S11	
K3.	Integrated circuit 74S20	
E1.	Integrated circuit 74S30	
F7.	Integrated circuit 74S40	
D4.	Integrated circuit 74S64	
A4,B4,C5,F2,F3,F5,H3, H6,K4.	Integrated circuit 74S74	
C4.	Integrated circuit 74S112	
R43.	Resistor 464Ω, 1/8W, 1%.	
R64.	Resistor 681Ω, 1/8W, 1%.	
R44.	Resistor 1.47KΩ, 1/8W, 1%.	
R45.	Resistor 2.15KΩ, 1/8W, 1%.	
R46.	Resistor 3.16KΩ, 1/8W, 1%.	
R52.	Resistor 100Ω, 1/8W, 1%.	
R30.	Resistor 330Ω, 1/4W, 5%.	
R31.	Resistor 510Ω, 1/4W, 5%.	
R6-9,11-20,22-25,27,28.	Resistor 560Ω, 1/4W, 5%.	
R2,4,5,10,21,26,29,32-42,47-51,53-63.	Resistor 1KΩ, 1/4W, 5%.	
C44-47.	Capacitor 47μF, 10V, electro.	
C50-63,562.	Capacitor 10μF, 25V, electro.	
C67.	Capacitor 560pF, 12V, 1%, micropocco.	
C49.	Capacitor 470pF, ceramic.	
C48.	Capacitor 33pF, ceramic.	
C8-29,31-43,64,65,68,70-75.	Capacitor 10nF, ceramic.	
F1	Ternet Resistor	
TR1	Transistor BSX20	
Y1	Quartz 20MHz QA60A U-Link DCW06	

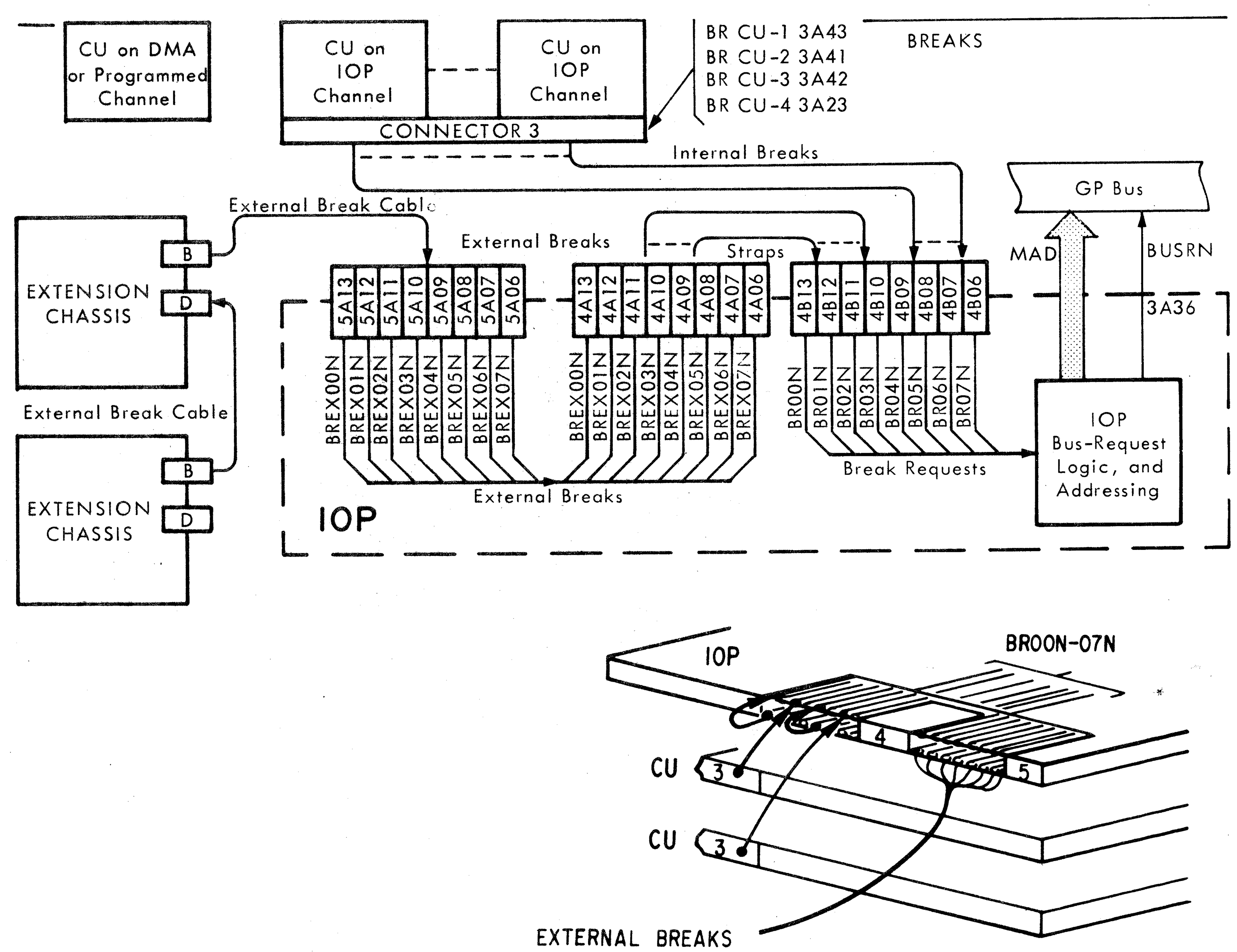


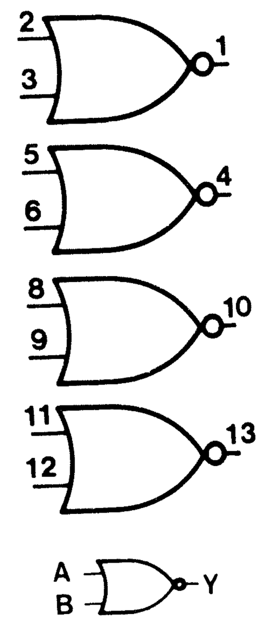
Figure B-11 Break-Request Signal Connections (IOP)

1801

(see 7400)

7402

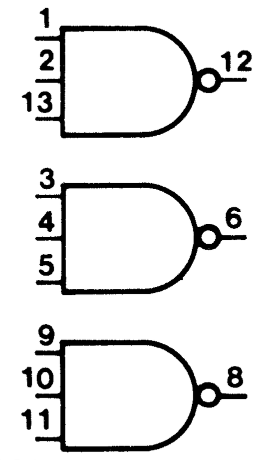
QUADRUPLE 2-INPUT
POSITIVE-NOR GATE



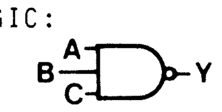
Vcc 14
GND 7

7410

7410



POSITIVE LOGIC:
$Y=\overline{ABC}$

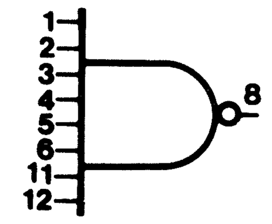


Vcc 14
GND 7

7430

7430

8-INPUT NAND GATE

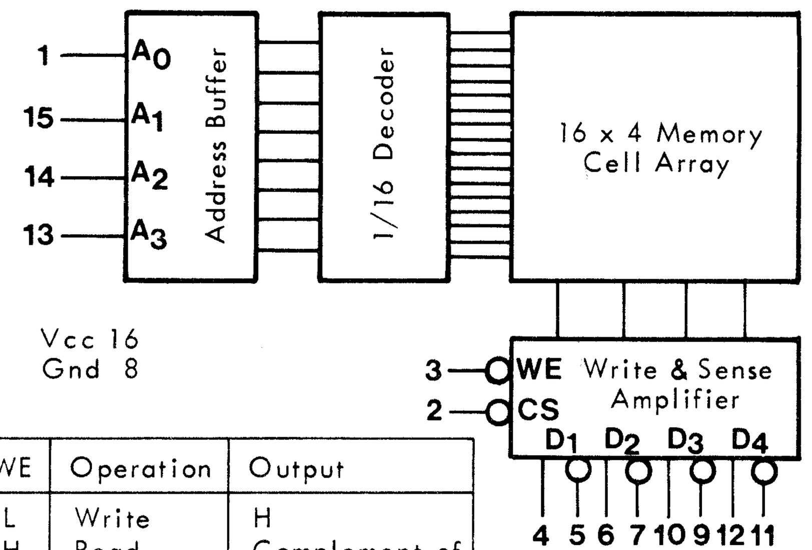


Vcc 14
GND 7

N.C. 9,10,13

3101

3101 HIGH SPEED FULLY DECODED 64 BIT MEMORY

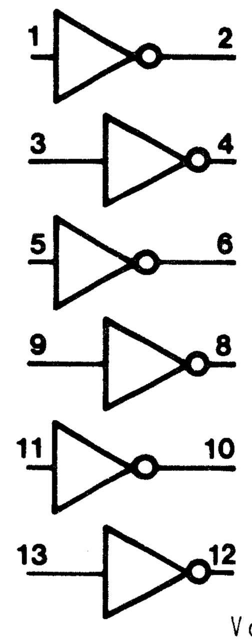


Vcc 16
Gnd 8

CS	WE	Operation	Output
L	L	Write	H
L	H	Read	Complement of Written Data
H	L		H
H	H		H

7404

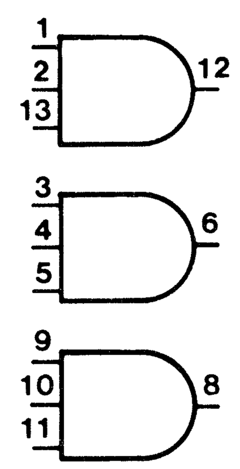
7404, S04, H04
7406 o.c. 30V
7414 sh Tris
7416 o.c. 15V



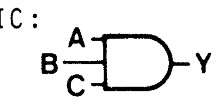
Vcc 14
GND 7

7411

7411



POSITIVE LOGIC:
Y=ABC



Vcc 14
GND 7

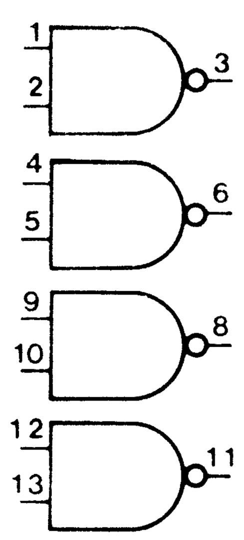
7437

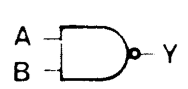
(see 7400)

7400

7400
S00
H00
03[1]
37[2]
38
1801[1]

1 open collector
2 buffer





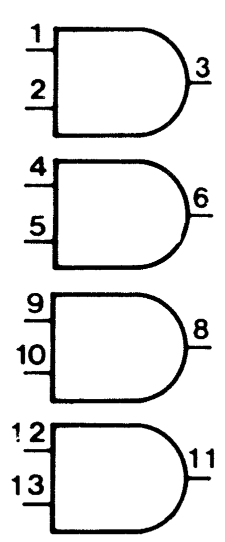
Y=AB

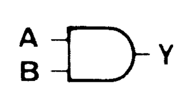
Vcc 14
GND 7

7408

7408
09[1]

1 open collector





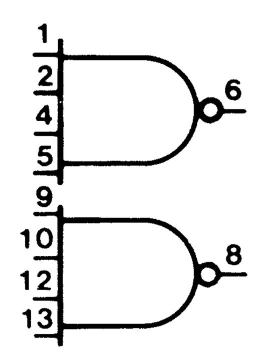
Y=AB

Vcc 14
GND 7

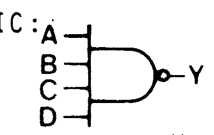
7420

7420 7440

BUFFER



POSITIVE LOGIC:
$Y=\overline{ABCD}$



3,11 N.C

Vcc 14
GND 7

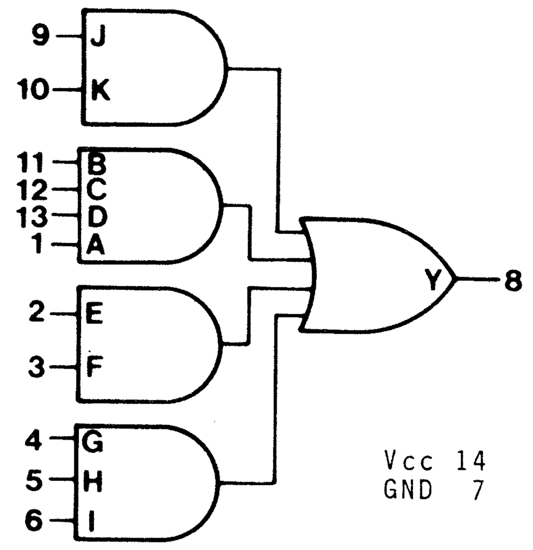
7440

(see 7420)

Figure B-12a IOP IC Guide

7464

4-2-3-2-INPUT AND-OR-INVERT GATES



Vcc 14
GND 7

FUNCTION TABLE

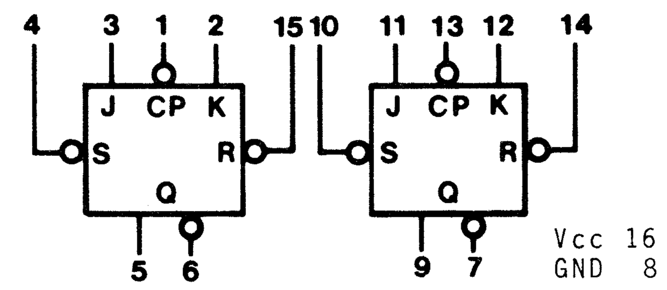
INPUTS					OUTPUTS	
PRESET	CLEAR	CLOCK	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H	H
H	H	↑	L	L	Qo	$\overline{Q}$o
H	H	↑	H	L	H	L
H	H	↑	L	H	L	H
H	H	↑	H	H	TOGGLE	
H	H	L	X	X	Qo	$\overline{Q}$o

Positive Logic: $Y=\overline{ABCD+EF+GHI+JK}$

74112

74S112

J-K NEGATIVE-EDGE-TRIGGERED FLIP-FLOPS



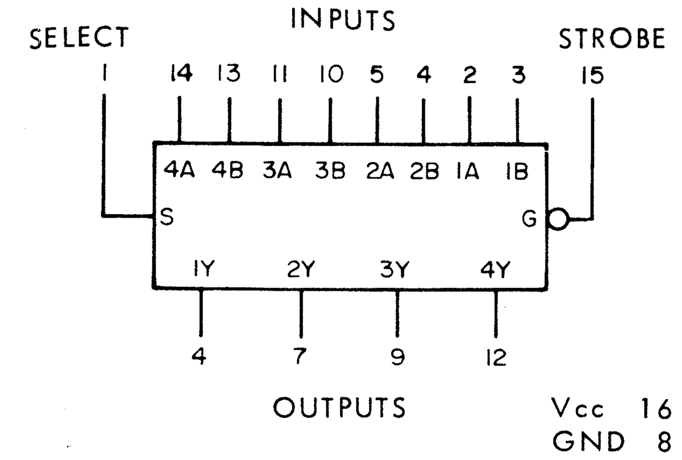
Vcc 16
GND 8

FUNCTION TABLE

INPUTS					OUTPUTS	
S	R	C	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H	H
H	H	↓	L	L	Qo	$\overline{Q}$o
H	H	↓	H	L	H	L
H	H	↓	L	H	L	H
H	H	↓	H	H	TOGGLE	
H	H	H	X	X	Qo	$\overline{Q}$o

74157

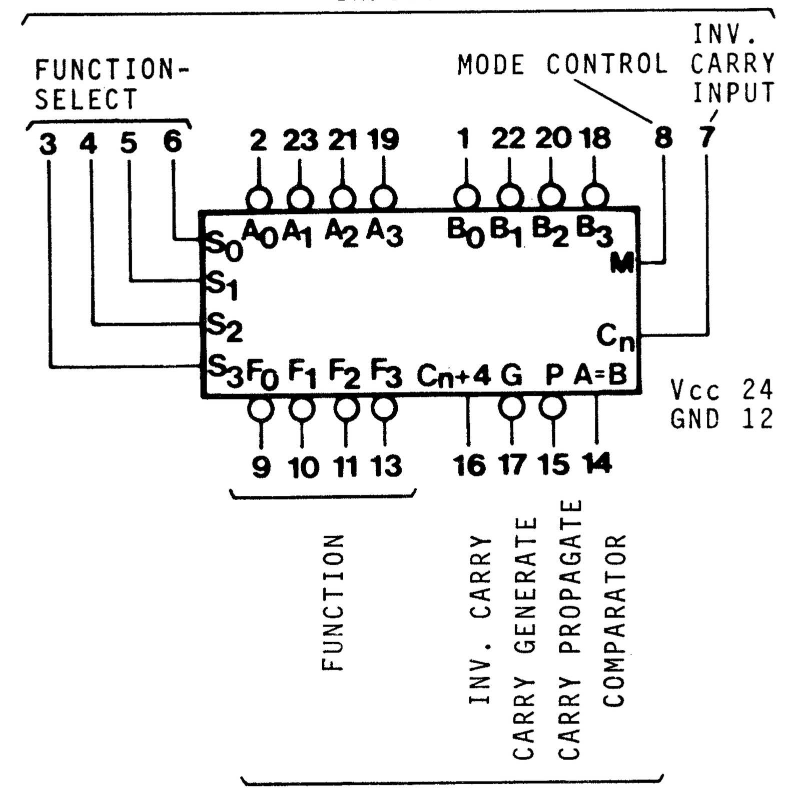
QUAD 2-to-1-LINE DATA SELECTORS/MULTIPLEXERS



Vcc 16
GND 8

74181

ARITHMETIC LOGIC UNIT



Vcc 24
GND 12

Table 1

SELECTION				ACTIVE-HIGH DATA		
				M=H LOGIC FUNCTIONS	M = L, ARITHMETIC OPERATIONS Cn = H (no carry)	M = L, ARITHMETIC OPERATIONS Cn = L (with carry)
S3	S2	S1	S0			
L	L	L	L	$F=\overline{A}$	F = A	F = A plus 1
L	L	L	H	$F=\overline{A+B}$	F = A + B	F = (A + B) plus 1
L	L	H	L	$F=\overline{A}B$	$F=A+\overline{B}$	$F=(A+\overline{B})$ plus 1
L	L	H	H	F = 0	F = minus 1 (2's compl)	F = zero
L	H	L	L	$F=\overline{AB}$	F = A plus $A\overline{B}$	F = A plus $A\overline{B}$ plus 1
L	H	L	H	$F=\overline{B}$	F = (A + B) plus $A\overline{B}$	F = (A + B) plus $A\overline{B}$ plus 1
L	H	H	L	$F=A\oplus B$	F = A minus B minus 1	F = A minus B
L	H	H	H	$F=A\overline{B}$	$F=A\overline{B}$ minus 1	$F=A\overline{B}$
H	L	L	L	$F=\overline{A}+B$	F = A plus AB	F = A plus AB plus 1
H	L	L	H	$F=\overline{A\oplus B}$	F = A plus B	F = A plus B plus 1
H	L	H	L	F = B	$F=(A+\overline{B})$ plus AB	$F=(A+\overline{B})$ plus AB plus 1
H	L	H	H	F = AB	F = AB minus 1	F = AB
H	H	L	L	F = 1	F = A plus A*	F = A plus A plus 1
H	H	L	H	$F=A+\overline{B}$	F = (A + B) plus A	F = (A + B) plus A plus 1
H	H	H	L	F = A + B	$F=(A+\overline{B})$ plus A	$F=(A+\overline{B})$ plus A plus 1
H	H	H	H	F = A	F = A minus 1	F = A

Table 2

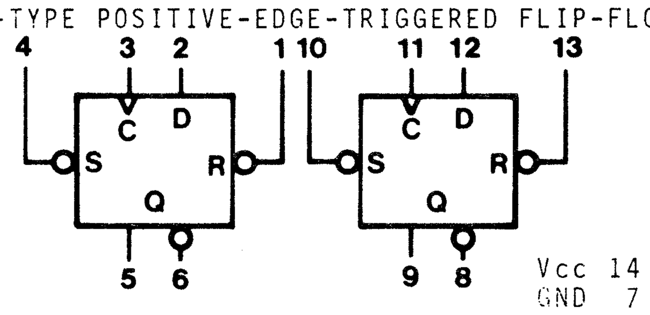
SELECTION				ACTIVE-LOW DATA		
				M=H LOGIC FUNCTIONS	M = L, ARITHMETIC OPERATIONS Cn = L (no carry)	M = L, ARITHMETIC OPERATIONS Cn = H (with carry)
S3	S2	S1	S0			
L	L	L	L	$F=\overline{A}$	F = A minus 1	F = A
L	L	L	H	$F=\overline{AB}$	F = AB minus 1	F = AB
L	L	H	L	$F=\overline{A}+B$	$F=A\overline{B}$ minus 1	$F=A\overline{B}$
L	L	H	H	F = 1	F = minus 1 (2's comp)	F = zero
L	H	L	L	$F=\overline{A+B}$	$F=A$ plus $(A+\overline{B})$	$F=A$ plus $(A+\overline{B})$ plus 1
L	H	L	H	$F=\overline{B}$	$F=AB$ plus $(A+\overline{B})$	$F=AB$ plus $(A+\overline{B})$ plus 1
L	H	H	L	$F=\overline{A\oplus B}$	F = A minus B minus 1	F = A minus B
L	H	H	H	$F=A+\overline{B}$	$F=A+\overline{B}$	$F=(A+\overline{B})$ plus 1
H	L	L	L	$F=\overline{A}B$	F = A plus (A + B)	F = A plus (A + B) plus 1
H	L	L	H	$F=A\oplus B$	F = A plus B	F = A plus B plus 1
H	L	H	L	F = B	$F=A\overline{B}$ plus (A + B)	$F=A\overline{B}$ plus (A + B) plus 1
H	L	H	H	F = A + B	F = A + B	F = (A + B) plus 1
H	H	L	L	F = 0	F = A plus A*	F = A plus A plus 1
H	H	L	H	$F=A\overline{B}$	F = AB plus A	F = AB plus A plus 1
H	H	H	L	F = AB	$F=A\overline{B}$ plus A	$F=A\overline{B}$ plus A plus 1
H	H	H	H	F = A	F = A	F = A plus 1

* Each bit is shifted to the next more significant position.

7474

7474, H74, S74

DUAL D-TYPE POSITIVE-EDGE-TRIGGERED FLIP-FLOPS



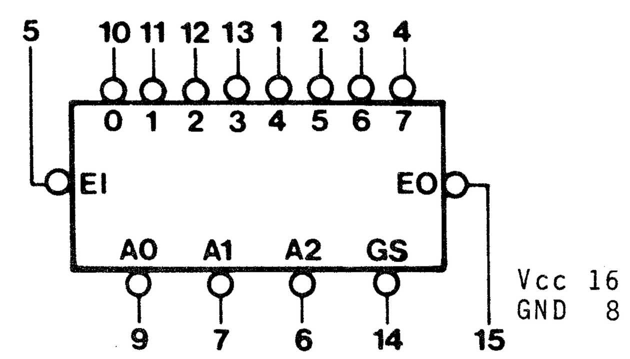
Vcc 14
GND 7

FUNCTION TABLE

INPUTS				OUTPUTS	
S	R	C	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Qo	$\overline{Q}$o

74148

8-LINE-TO-3-LINE PRIORITY ENCODER



Vcc 16
GND 8

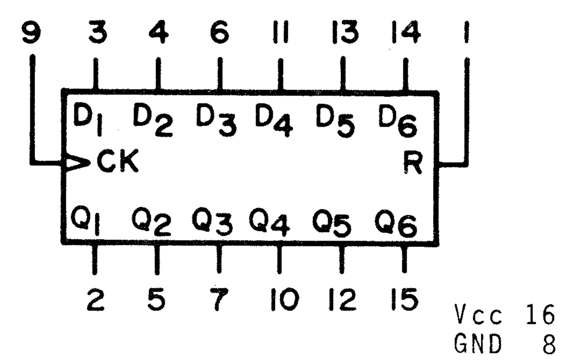
FUNCTION TABLE

INPUTS									OUTPUTS				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

74174

74174

HEX D-TYPE FLIP-FLOPS WITH COMMON CLOCK AND RESET



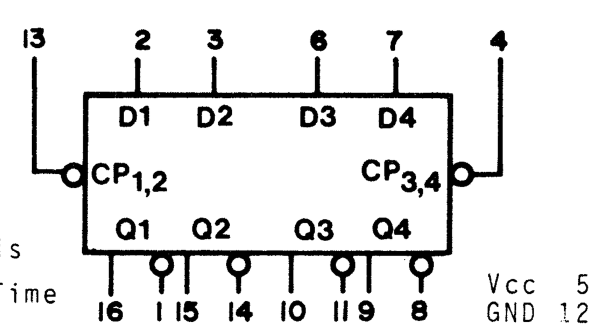
Vcc 16
GND 8

FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\overline{Q}$t
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	$\overline{Q}$o

7475

7475

4-BIT LATCH



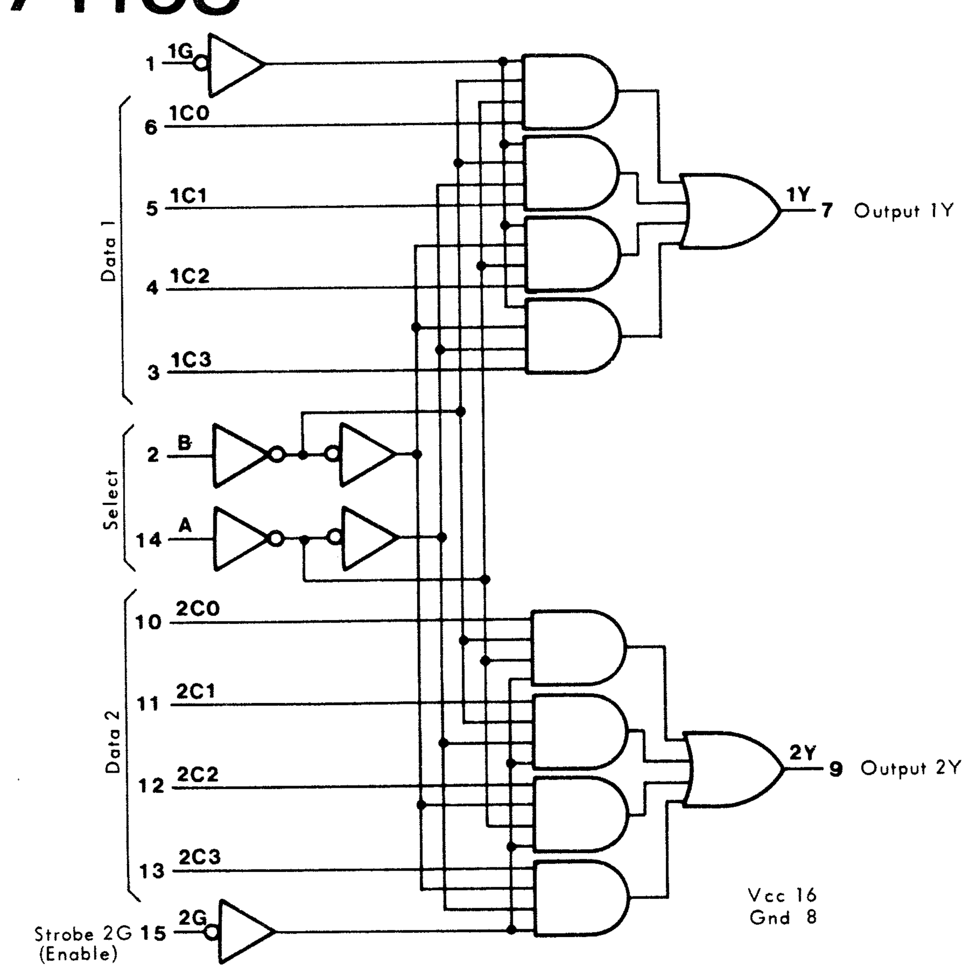
CP is Store Time

Vcc 5
GND 12

Q follows D when CP is high

74153

DUAL 4-LINE-TO-1-LINE DATA SELECTOR/MULTIPLEXER

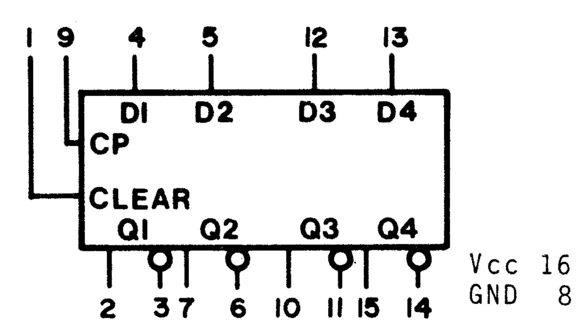


Vcc 16
Gnd 8

74175

74175

QUAD D-TYPE FLIP-FLOPS

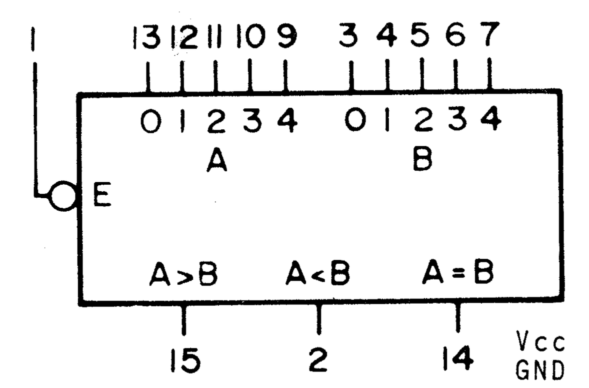


Vcc 16
GND 8

FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\overline{Q}$
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	Qo

9324

5-BIT COMPARATOR



Vcc 16
GND 8

A High Speed Expandable Comparator that provides comparison between two 5-bit words and gives three outputs, "less than", "greator than" and "equal to". A HIGH level on the active LOW enable input forces all three outputs LOW.

REC 0613

(see 7404)

APPENDIX C

MEMORY MANAGEMENT UNIT

SERVICE MANUAL

Effective Pages February 1977

ii
C.1 through C.19

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

APPENDIX C

MEMORY MANAGEMENT UNIT

C.1 GENERAL

The Memory Management Unit (MMU) is a hardware option which provides memory addressing and protection facilities for the P857 system. The MMU is a system slave which operates under CPU Bus control. The three main functions of the MMU are:

- Address translation which extends memory addressing up to 128k words.
- Page Usage Timing for dynamic program relocation on a page basis.
- Full memory protection.

C.2 Address Translation

The primary function of the MMU is to extend memory addressing up to 128k physical words while retaining the 32k-word logical address selection from the CPU instruction (Figures C-1 and C-2).

- A sixteen-segment table is pre-loaded with page addresses by a single Table Load instruction.
- Each CPU/memory transfer via the MMU (MMU translation) then uses the four most-significant address lines (S0-3) to select a table segment, and thus a page address, while the 12 least-significant address bits select one of the 2k words within the page.
- A Table Store instruction can be used by software to access the 16-word segment table for testing or dynamic relocation.

C.3 Page Usage Timer

The function of the Page Usage Timer is to keep track of which pages have not been accessed within a certain Page Residency Parameter (Window Interval). This parameter is based on the number of MMU translations and is derived from a software constant and an MMU hardware constant of 1024 (2^{10}).

- An eight-bit (0-255) software constant is loaded into the MMU by a WER instruction. This interval time count is used as a software constant common to all 16 page timers.
- Each MMU translation resets the timer for the addressed page, and is counted as an "interval" time unit.
- Each time the interval count is complete, a single page timer (located in the MMU scratchpad) is incremented, and the next sequential page timer is selected in preparation for the succeeding page-timer count.
- A non-addressed page thus counts to overflow after a number of MMU translations equal to the software constant times the hardware constant 1024 (2^{10}), as follows:
  -- Window interval (count 0-255).
  -- Sequential count through all 16 page timers (2^4).
  -- Full six-bit page timer (2^6).

C.4 Memory Protection

Each of the 16 words loaded into the segment table by the TL instruction includes, in addition to the page address, two memory-protection bits established by software.

- Bit 6 (E) -- The Page Error bit is set to restrict the page to System Mode. If an address translation in User Mode (FU set) attempts to access the page, the translation is blocked and MFAULTN is sent to the CPU.
- Bit 7 (R) -- The Read-Only Page bit is set to restrict the page to read operations only. If an address translation in User Mode attempts a write operation on this page, the translation is blocked and MFAULTN is sent to the CPU.

C.5 SOFTWARE CONSIDERATIONS

C.6 System/User Operations

The main memory is divided into a System part and a User part, with the System part always located at the beginning of the memory (lowest addresses). The System part contains the supervision programs (monitors) and their associated working areas (tables, buffers, etc.). The User part contains the user's running programs or real-time tasks, etc. The CPU operates in either System Mode or User Mode, and certain instructions are restricted to System Mode only.

C.7 The system part of memory is located in the beginning of memory and is limited to the first 32k word positions. Addressing the System part of memory is done with the 16-bit logical address directly from the CPU, and the MMU is not used. However, the System is able to extend its addressing by using the extended instructions (EL, ES, MVSU, MVUS) which use MMU translation.

C.8 The maximum length of a single user program is 32k words, divided into pages of 2k words. The CPU addresses a user location with the 16-bit logical address. The MMU translates the first four bits into a six-bit physical page address (which was previously loaded into the MMU). The remaining 12 bits from the CPU then select one of 2k words within the selected page.

C.9 System mode instructions used by the MMU:

- Table Load (TL) loads 16 physical page addresses and protection bits from memory into the MMU segment table. This must be done prior to address translations through the MMU.
- Table Store (TS) stores the 16-word segment table (including page addresses, protection bits, and page timers) from the MMU into memory.
- Write External Register (WER) loads a Window-Interval software constant into the MMU for use with the page timers. The WER instruction is not exclusive to MMU operations.

Address translation is usually performed by the MMU while the CPU operates in User Mode, and the MMU does memory-protection checking. There are, however, four System-Mode instructions which are used for address translation operations:

- Extended Load (EL) and Extended Store (ES) are used by the System to access the User area of memory to read or write a single word.
- Move Table User-to-System (MVUS) and System-to-User (MVSU) are used by the System to transfer blocks of data between the User area of memory and the System area.

The only difference the MMU notices for these System-Mode address translation operations is that the user-mode bit (FU) is not active, and the MMU does not do the memory-protection checking.

C.10 The Operating System must load the segment table every time a new User program is given control. The Operating System is also responsible for sending the 18-bit physical addresses of the User's I/O buffers to the I/O channels (I/O Processors or DMA controllers).

C.11 Page Replacement Facilities

C.12 Window Interval. The required window interval count (N) to obtain a specific page residency time (T) is: the time (T), divided by the average memory access time (θ), divided by the hardware constant (1024).

$$T = 1024 . \theta . N \quad \text{or} \quad N = T \div \theta \div 1024$$

For example, if a page residency time of 100ms is desired and the average memory access time (θ) is 2µsec, the window interval count (N) is:

100ms ÷ 2µsec ÷ 1024 = 50

Set into MMU by WER instruction.

Hardware constant (count through 16 timers of six bits each)

Average memory access time: produces one count each time the MMU is accessed for address translation.

C.13 Modified Page. Bit M (08) is set by the MMU whenever a write operation (Store instruction) is performed on a specific page. This feature indicates to the Operating System when a page needs to be swapped out before a new program segment is loaded at the same place. If not necessary, a direct over-writing is possible, resulting in a large saving of time.

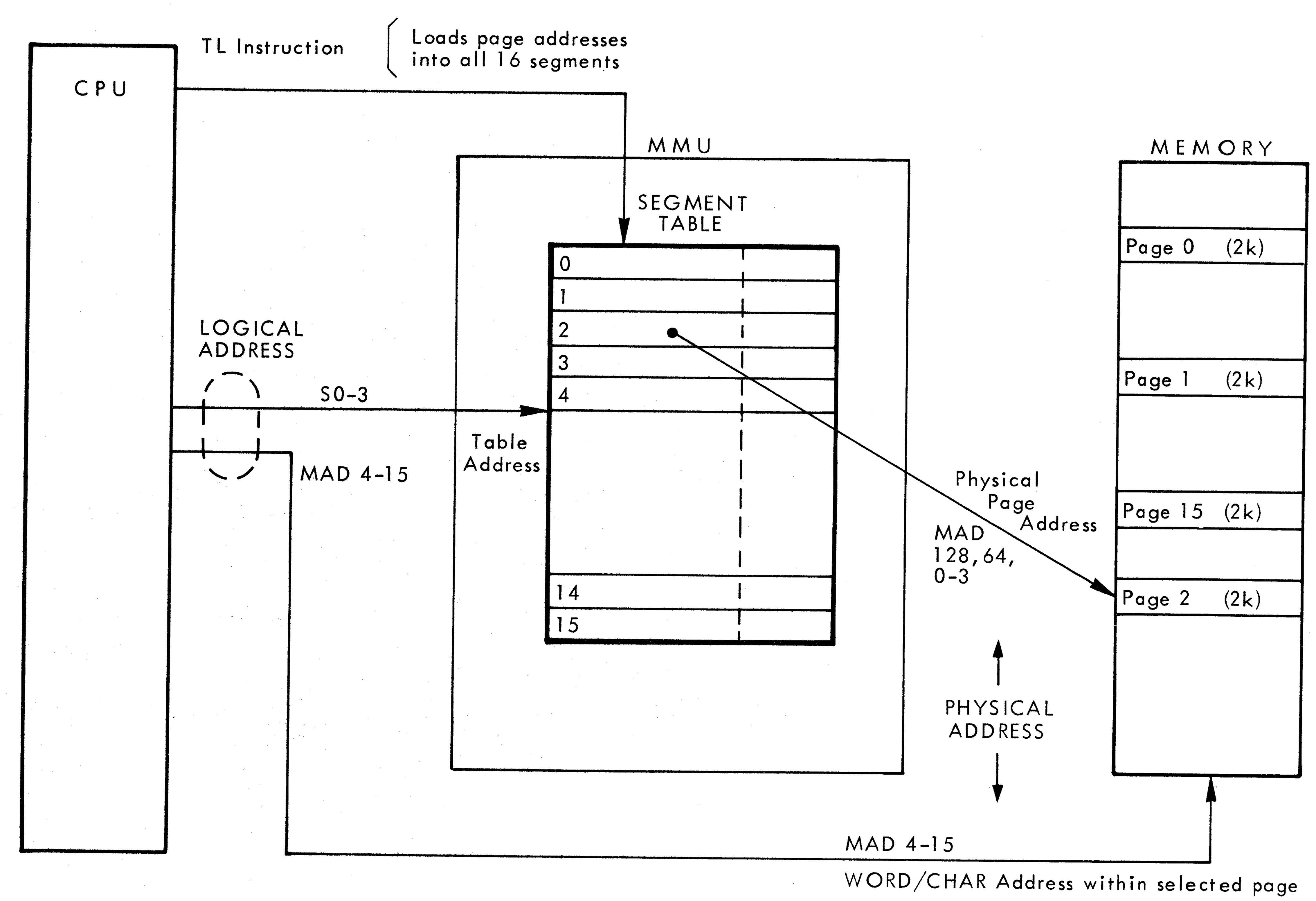
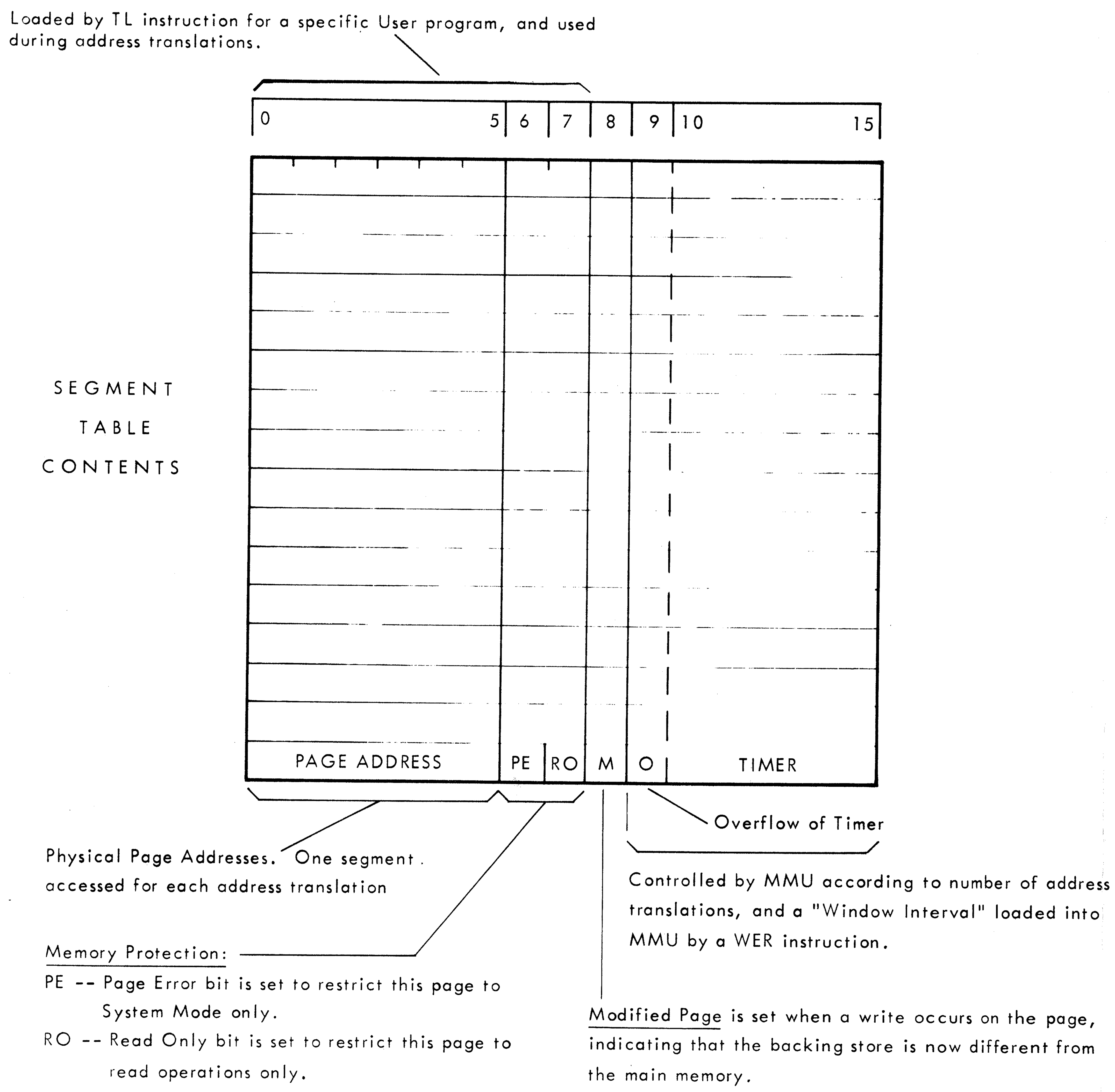


Figure C-1 MMU System Diagram



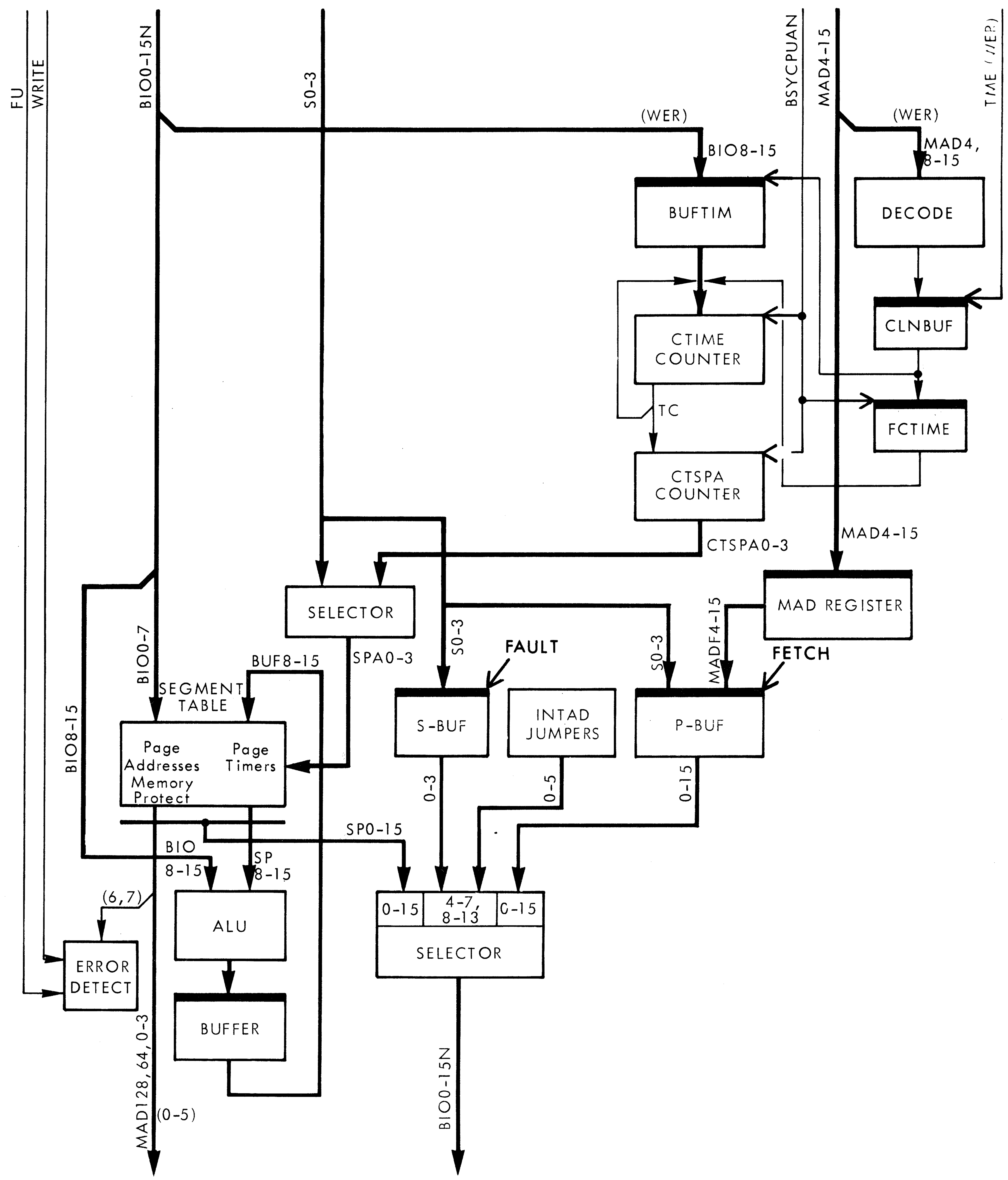


Figure C-2 MMU Segment Table Contents

Figure C-3 MMU Block Diagram

C.14 LOGIC DESCRIPTION

The MMU logic description is organized into the different operating sequences of the MMU. A general block diagram of the MMU is provided in Figure C-3. The detailed logic is shown in Figure C-10 at the end the logic description. The different operating sequences used in the MMU are:

- Table Load
- Table Store
- WER -- Interval Time Loading
- Translation Operation
- Increment Timer
- Page Fault

C.15 Table Load

The CPU starts the Table Load operation (Figure C-4) in the MMU by sending TMMN. The CPU controls the operation with the signal BSYCPUAN which it sends for each of the 16 page addresses to be loaded into the MMU segment table. TRMN is received from the memory each time the data from the memory is valid. The TMMN signal resets counter CTSPA (via TLS and CTSPAZON) so that segment-table position 0 is selected for the first entry. The active signal TLSN enables CTSPA counting and selects the counter input to the SP-Address Selector with a high SPAS.

C.16 When the memory is ready with the data, TRMN goes active and removes the reset clamp STARTZON from the START flip-flop. The following high transition of OSC then sets START. The MMU clock T1, T2, T3, T4 runs when START is set, synchronized on OSC from the CPU. One word with the page address and memory-protection information is gated from BIO into the segment table at time T3. Also at T3, DONEMN is sent to the CPU to request the next word transfer.

C.17 Next Word Transfer. The CPU responds to DONEMN by dropping BSYCPUAN. The MMU increments the CTSPA count at the trailing edge of BSYCPUAN to select the next position in the segment table. START resets on the first rising OSC after BSYCPUAN drops, and is held reset when TRMN goes inactive. When memory is ready with data, TRMN goes active and the MMU

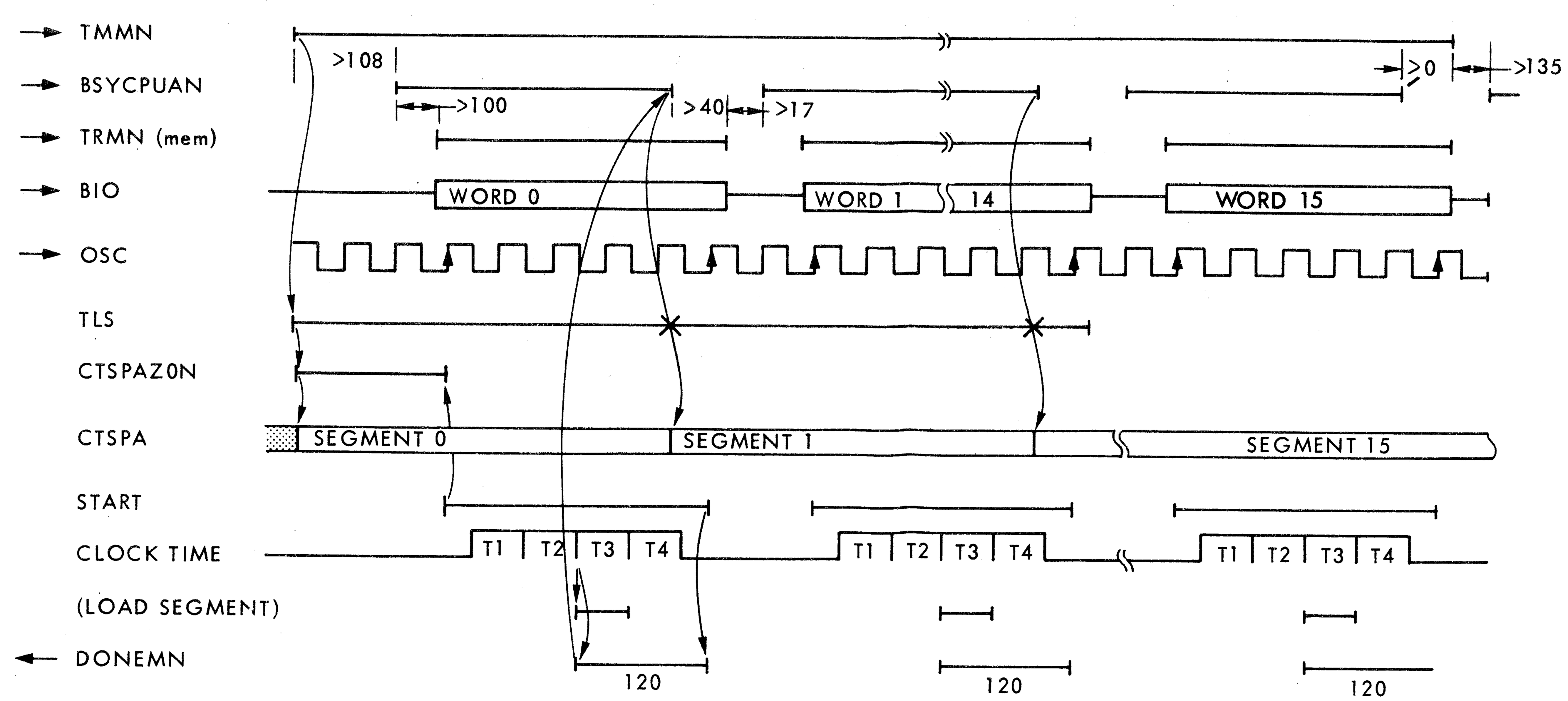


Figure C-4 Table Load Sequence

sequence (START, clock cycle, BIO → segment table, DONEMN) is repeated to load this word.

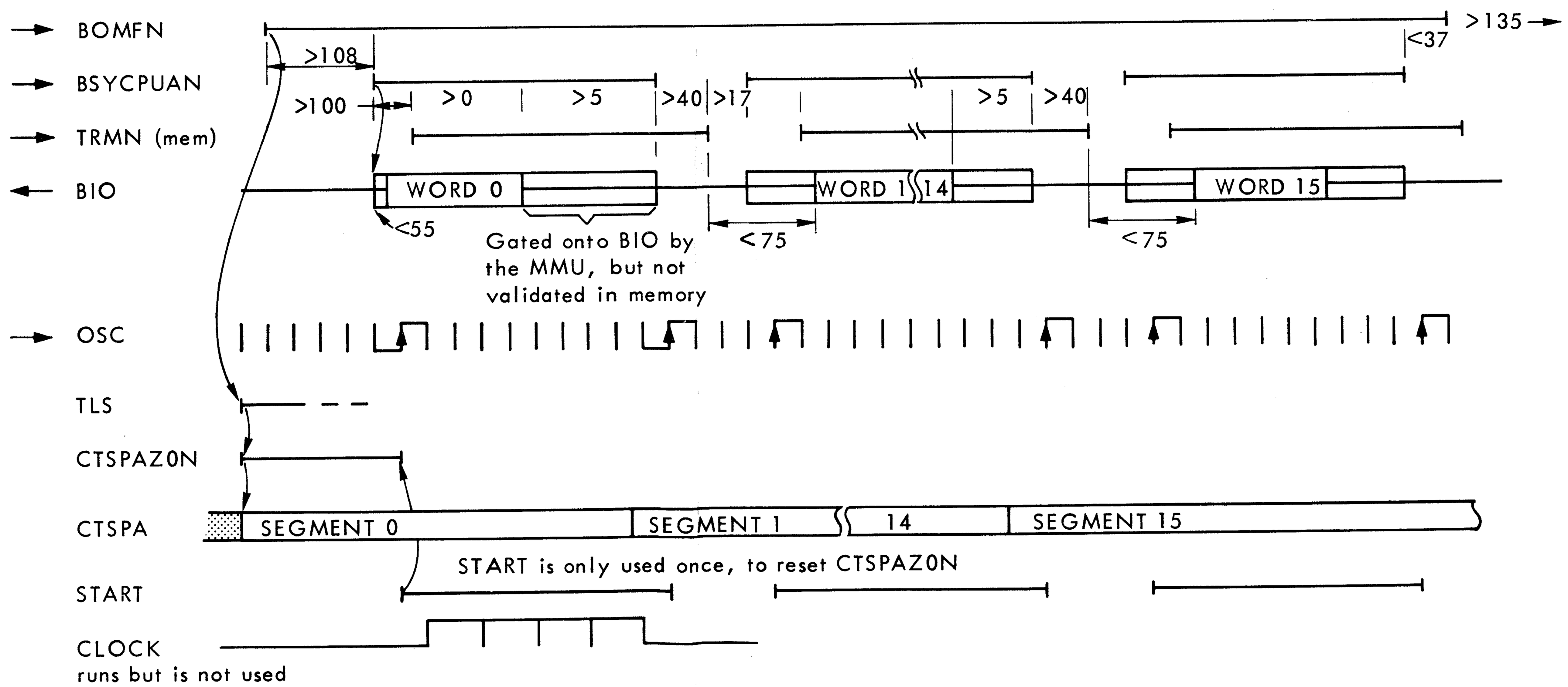
C.18 Last Word Transfer. The CPU drops TMMN after it receives the 16th DONEMN from the MMU. The inactive TMM signal in the MMU blocks further sending of DONEMN to the CPU. In all other respects, the MMU sequence for the last word transfer is identical to the previous transfers.

C.19 Table Store

The CPU starts the Table Store operation (Figure C-5) in the MMU by sending BOMFN. The CPU controls the operation with the signal BSYCPUAN which it sends for each of the 16 segment-table words to be transferred out to memory. TRMN is received from the memory each time the BIO data from the MMU has been accepted. The CPU uses TRMN to know when to initiate each transfer. The BOMFN signal resets counter CTSPA (via TLS and CTSPAZ0N) so that segment table position 0 is selected for the first entry. The active signal TLSN enables CTSPA counting and selects the counter input to the SP-Address Selector with a high SPAS.

C.20 The MMU uses BSYCPUAN and BOMFN to gate the first segment-table word (including page address, memory protection, and page timer information) onto the BIO to the memory. When the memory has accepted the data, TRMN goes active and removes the reset clamp STARTZ0N from the START flip-flop. The following high transition of OSC then sets START. The START signal resets CTSPAZ0N to allow the CTSPA counter to increment at each trailing edge of BSYCPUAN. START and the MMU clock continue to operate with each BSYCPUAN input, but are not used for the rest of the Table Store operation.

C.21 Each successive word is gated from the MMU segment table onto the BIO lines by the BSYCPUAN signal. The trailing edge of BSYCPUAN increments CTSPA to select the words sequentially from the segment table. The CPU terminates the operation after the 16th word transfer by dropping the BOMFN signal.



NOTES:

- Times in ns.
- OSC signal is not complete; shows only the relationship between edges of BSYCPUAN and OSC, for controlling START and clock times.

Figure C-5 Table Store Sequence

C.22 WER -- Interval Time Loading

The CPU loads the window interval software constant into the MMU with a WER instruction (Figure C-6). The CPU starts the operation by activating BSYCPUAN and, at the same time, placing the MMU address on the MAD08-15 lines and placing the window interval software constant on the BIO08-15 lines. When the MMU decoder logic detects its own address on the MAD lines and the WER instruction code (MAD04 = 0), signal MAREN is activated. With decoded-address signal MAREN active, the CPU timing signal TMEN sets flip-flop CLNBUF. The CLNBUF output gates the interval time from the BIO lines into the BUFTIME register and also, in conjunction with TMEN, generates the timing response signal TRMN back to the CPU.

C.23 The CPU responds to TRMN by terminating BSYCPUAN to the MMU. The trailing edge of BSYCPUAN, gated by CLNBUF, sets the FCTIME flip-flop. With FCTIME set and the interval time loaded in the BUFTIME register, the MMU is ready for page-time counting during the address-translation operations. When the CPU drops TMEN, the MMU resets CLNBUF (via CLBUFZ0, CLBUFZ0N) and drops the TRMN signal to the CPU.

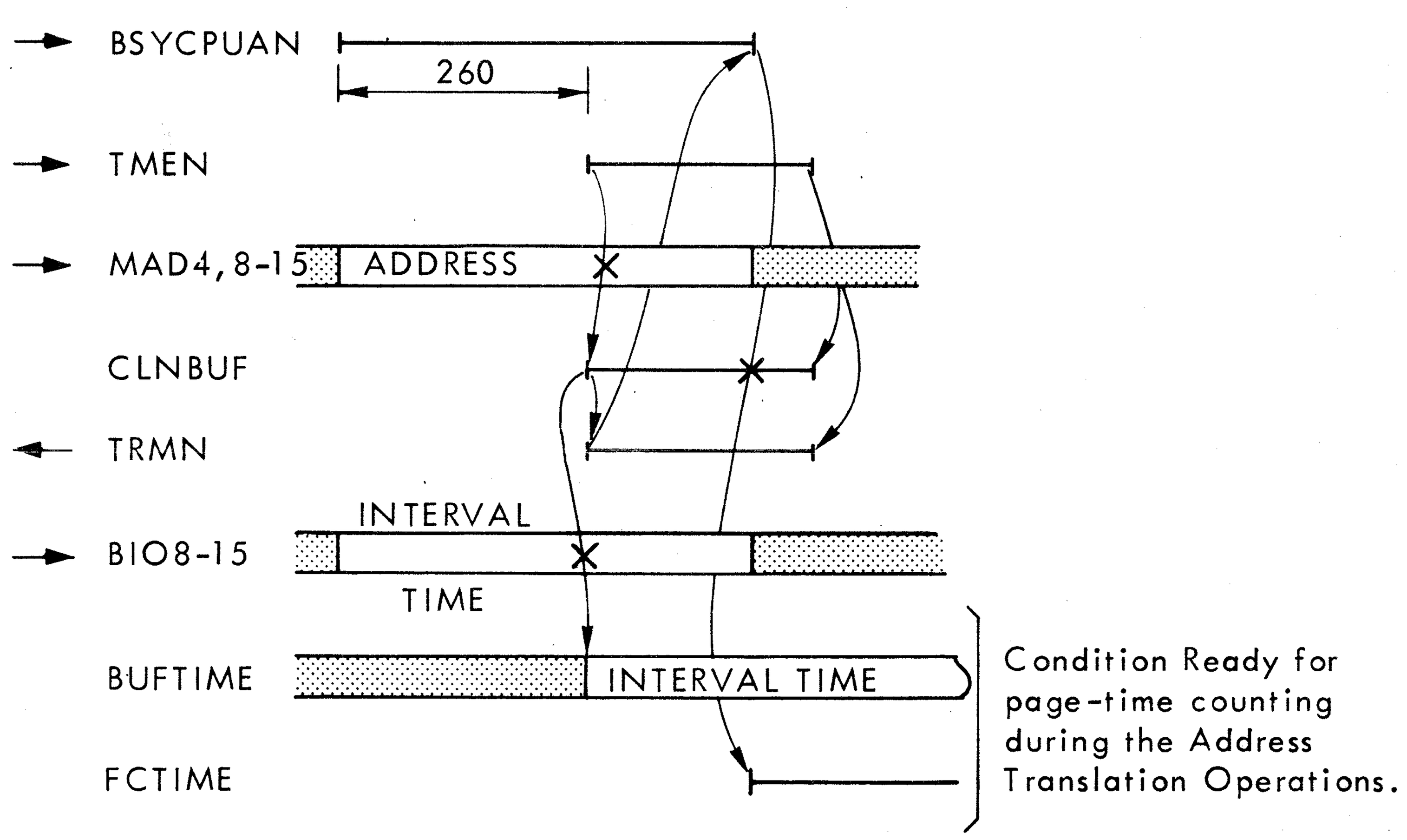


Figure C-6 WER Sequence (Loading Window Interval Software Constant)

C.24 Translation Operation

The CPU activates signal TMMU to indicate a memory transfer requiring MMU address translation (Figure C-7). The MMU scratchpad address selector is switched to the S00-03 input by the inactive SPAS ($\overline{\text{FINCRN}}.\overline{\text{TLSN}}$). TMMU and BSYCPUAN together gate the physical page address (SP00-05) from the location selected by the logical page address (S00-03).

C.25 The START flip-flop is set on the first rising edge of OSC after BSYCPUAN is received. Each trailing edge of OSC then increments the MMU clock. The signals TMMU.START.$\overline{\text{FINCRN}}$ (=TRANSL) set the BUF08D selector to test the WRITE signal from the CPU. TRANSLN, together with $\overline{\text{TMMN}}.\overline{\text{FINCR}}$, sets the ALU to the Logical-Zero mode ($\overline{\text{FINCRN}}$ to the Mode input pin 8 selects Logical operations). The leading edge of T3 gates the addressed-word timer from SP08-15, through the ALU, to the Buffer: this resets the timer (bits 09-15) and sets bit 8 if the operation is a Write operation. The Buffer contents are reloaded into SP08-15 at the trailing edge of T3.

C.26 When the SP word is accessed by S00-03, the bits 6 and 7 are compared with the CPU command bits FU and WRITE. If bit 6 is set and FU is active (System Mode only bit and User Mode), ERRORN is generated. If bit 7 is set and WRITE is active (Read-Only bit and Write command) and FU is set (User Mode), ERRORN is generated and MFAULTN is sent to the CPU. This error condition prevents setting TMRF at T2, and blocks the sending of TMRF to memory, thus blocking the memory transfer.

C.27 If the memory check is correct, the TMRF flip-flop is set at T2 by the rising edge of OSC. TMRF and START together send timing signal TMRN to the memory. When the memory part of the transfer is complete, it sends TRMN to the CPU. The CPU responds by dropping BSYCPUAN to the MMU.

C.28 During the first translation operation following a WER instruction, flip-flop FCTIME is set. When BSYCPUAN drops at the end of the first translation, the complement of the software constant is gated from the BUFTIM

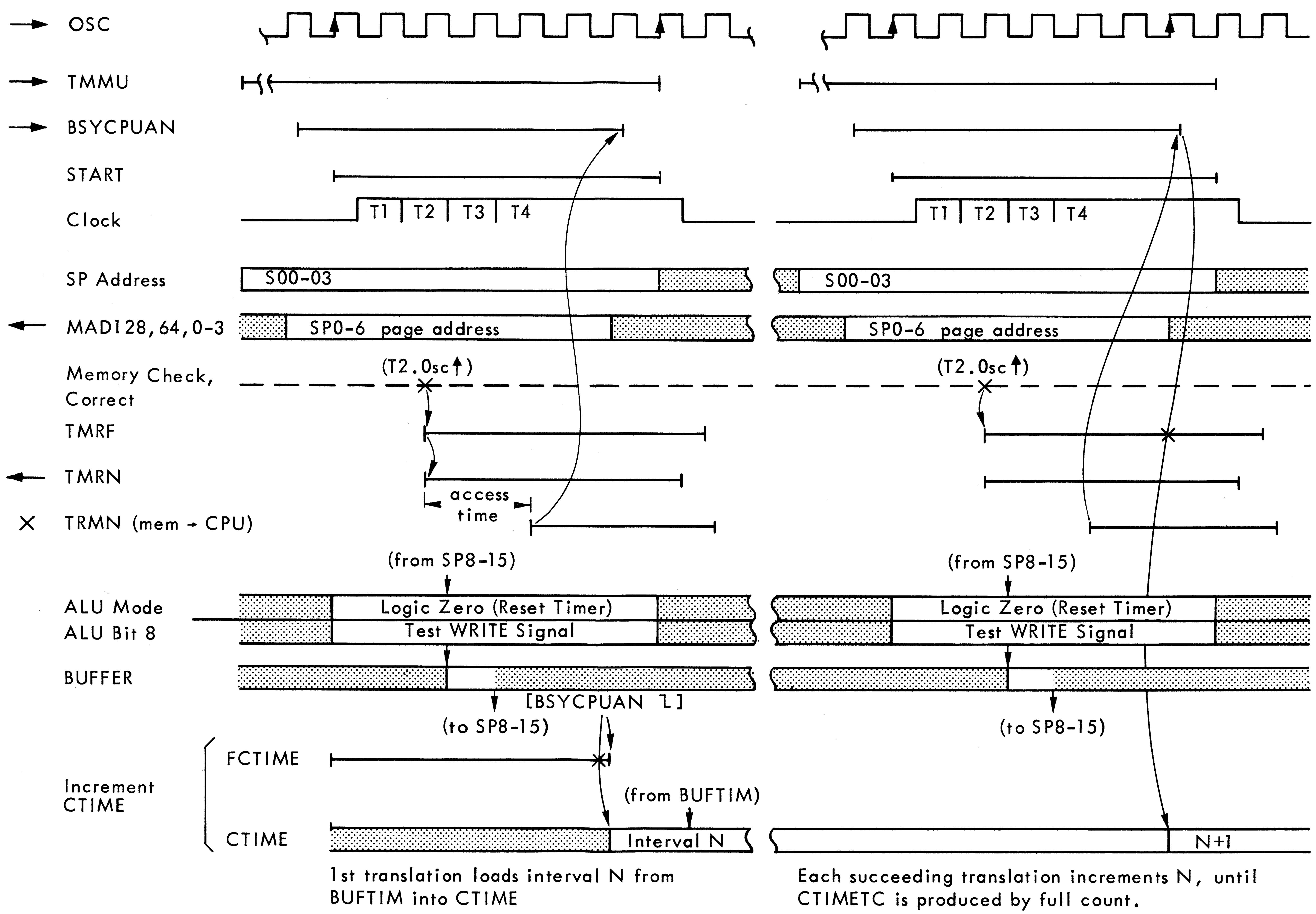


Figure C-7 Translation Sequences

register into the CTIME counter. When BSYCPUAN drops at the end of all succeeding translations, CTIME is incremented.

C.29 Increment Timer

The complement of the software constant is loaded into CTIME from the BUFTIM register (Figure C-8). Each translation increments CTIME so that it will be full when the number of translations equals the software constant. When the trailing edge of BSYCPUAN increments CTIME to full, CTIMETC is produced. The next rising edge of OSC then sets flip-flop FINCR. The same edge of OSC also resets TMRF (normal translation sequence). With TMRF removed from the CET input of CTIME, the CTIMETC output drops.

C.30 The FINCR flip-flop activates SPAS so that the scratchpad address selector selects the CTSPA source. The active FINCR signal also sets the ALU and the BUF08D selector to the arithmetic A+1 mode. At the leading edge of time T3, the scratchpad page timer (bits 08-15) selected by CTSPA is gated through the ALU into the BUFFER. At the trailing edge of T3, the buffer contents are loaded back into the scratchpad, now incremented by 1. FINCR is reset by the T4 clock, and the inactive FINCR signal then resets T4 to end the increment timer sequence.

C.31 TMRF is generated during the succeeding translation sequence and, since CTIME still contains its full count, the CTIME output is reactivated. At the end of the translation then, CTIMETC causes the trailing edge of BSYCPUAN to increment the CTSPA counter (for selecting the next page timer) and to reload the CTIME counter with the complement of the software constant from the BUFTIM register.

C.32 Page Fault

A page fault is detected during a normal Translation sequence if a User Mode operation (FU set) accesses a System-Mode-only page (bit 6 set) or if a Write operation is attempted (WRITE signal) in User mode on a read-only page (bit 7 set). The resultant ERROR signal inhibits the TMRF flip-flop and blocks the sending of the memory timing signal TMRN. The flip-flop FAULT is set at time T2 (Figure C-9). FAULT generates the MFAULTN signal to the CPU, gates the S00-03 bits into the S-Buf register, and sets the BIOS0 flip-flop.

C.33 BIOS0 and BIOS1 are now set (=1,0) to select the P-Buf source via the BIO Select circuit. The CPU responds to MFAULT by sending BOMFN and BSYCPUAN to the MMU. These two signals together gate the P-Buf contents (P00-15) onto the BIO lines to the CPU. At the trailing edge of BOMFN, BIOS0 is reset and BIOS1 is set (=0,1) so they select the S-Buf and INTAD sources via the BIO Select circuit. The next time BSYCPUAN and BOMFN are received together from the CPU, the MMU gates the segment-table address (S00-03) and the MMU interrupt code (INTAD0-5) onto the BIO lines.

C.34 The CPU activates BSYCPUAN once between the two BOMFN signals to transfer its PSW to memory. Since BOMFN is not active, however, there is no action taken in the MMU at that time.

Stack after Page Fault Interrupt

0	4	8	13 15		
0 0 0 0	S0-4	INTAD0-5	0 0	MMU →	Memory
PSW				CPU →	
P0-15				MMU →	

S = page number
P = program counter or aborted instruction

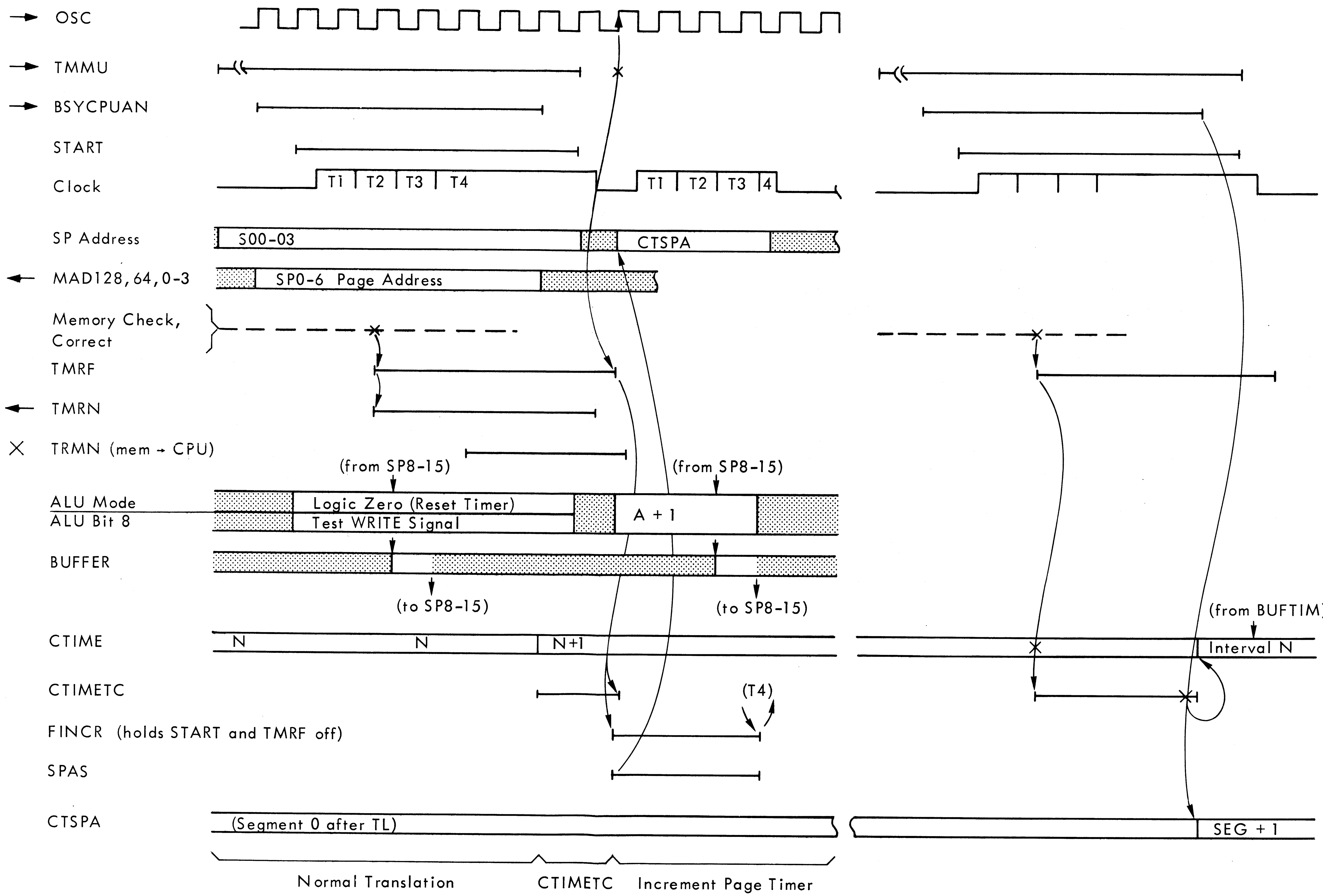


Figure C-8 Increment Timer Sequence

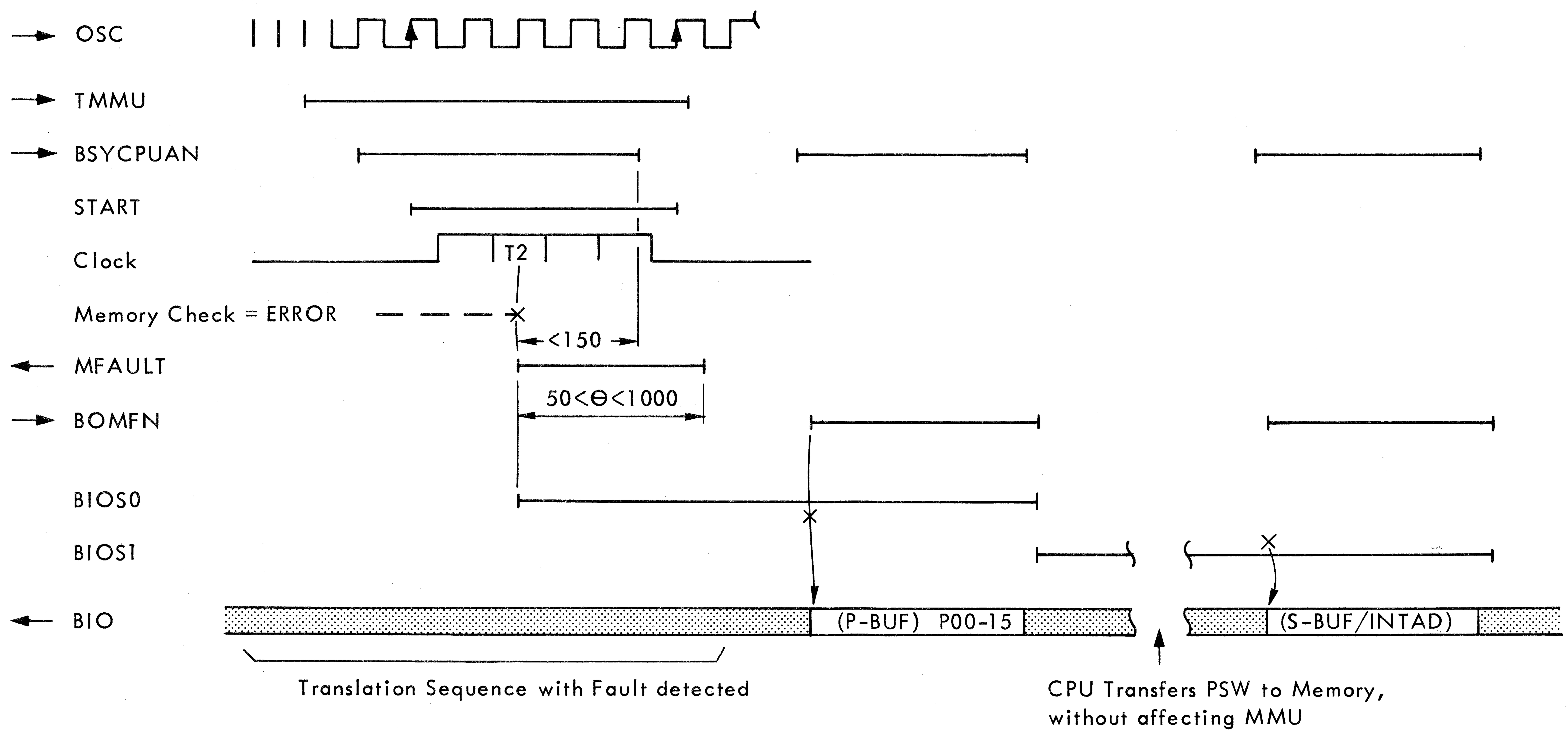


Figure C-9 Page Fault Sequence

C.35 INPUT/OUTPUT SIGNALS

CPU → MMU Signals	
BSYCPUAN	CPU has Bus control. Validates Bus signals and controls all MMU operations.
BOMFN	Table Store control signal. Page-fault stack-loading control signal: 1st -- to read aborted instruction-counter value; 2nd -- to read logic page address which set page fault and the program level of the page-fault job, coded on the MMU card.
FU = 1 = 0	CPU in User Mode. Any memory violation sets page fault. CPU in System Mode.
GFETCH	Fetch cycle is executed by CPU. Instruction counter value is loaded into MMU P-Buf register.
OSCFLO	CPU clock signal, used by MMU for internal timing.
S00-03	Logical page address from CPU P-register.
TMMN	Table Load control signal.
TMMU	Translation control signal; for page-address translation, memory protection check, and memory activation by the MMU.
MMU → CPU Signals	
DONEMN	MMU reply during Table Load as each segment is loaded.
MFAULTN	Page fault is detected during translation.
MMUABS	Held at 0v (inactive) when MMU card is in place.
BUS → MMU Signals	
BIO0N-15N	Data used during Table Load and WER operations.
CLEARN	Clear signal for initializing system.
TMEN	WER instruction timing for loading the software constant into the MMU BUFTIM register.
TRMN	Memory reply: -- During Table Load and Table Store validates data. -- During translation releases memory activation.
MAD04-15	Address lines to: -- select the MMU during WER instruction, and -- store the CPU instruction-counter value during each Fetch instruction cycle.
WRITE	CPU command for store cycles. Used during translation to test for page fault (read-only page and User Mode), and sets segment table Modified-Page bit (8) of accessed page (if no page fault).

MMU → BUS Signals	
BIO0N-15N	Data used during Table Store or Page-Fault stack loading operations.
MAD128,64, 0-3	Memory address lines for physical page addresses from the segment table.
TMRN	Activates memory during page address translation, if there is no page fault.
TRMN	External register reply to CPU when software constant is loaded during WER instruction.

C.36 CIRCUIT BOARD AND COMPONENTS

The Memory Management Unit is implemented on a double-sided printed circuit board. Component locations are shown on Figure C-11. The MMU uses connector 3 for GP Bus connections and power, and connector 5 for special CPU control signals. The MMU parts list is given in Table C-1. A list and guide of the integrated circuits used by the MMU is provided in Figure C-12.

C.37 ADDRESS/INTERRUPT U-LINKS

The U-link connections for both the external-register address of the MMU and the interrupt level address are shown on Figure C-11. For both groups of U-links, the least-significant bits are at the top and the most-significant are at the bottom. The examples shown on Figure C-11 are set up for the standard external-register address and standard interrupt level of the MMU (address /80 and level 47_{10}).

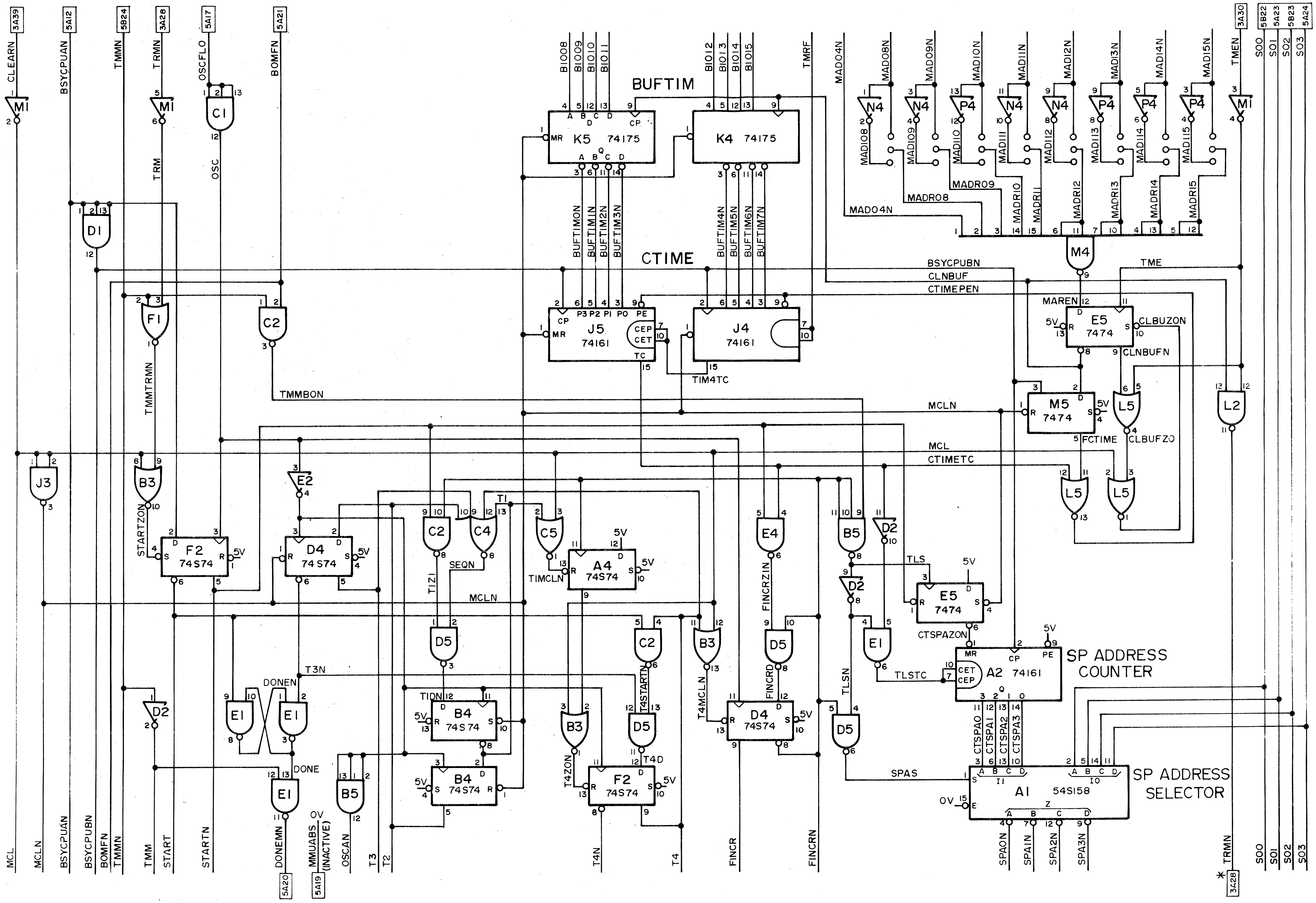


Figure C-10a MMU Logic Diagram

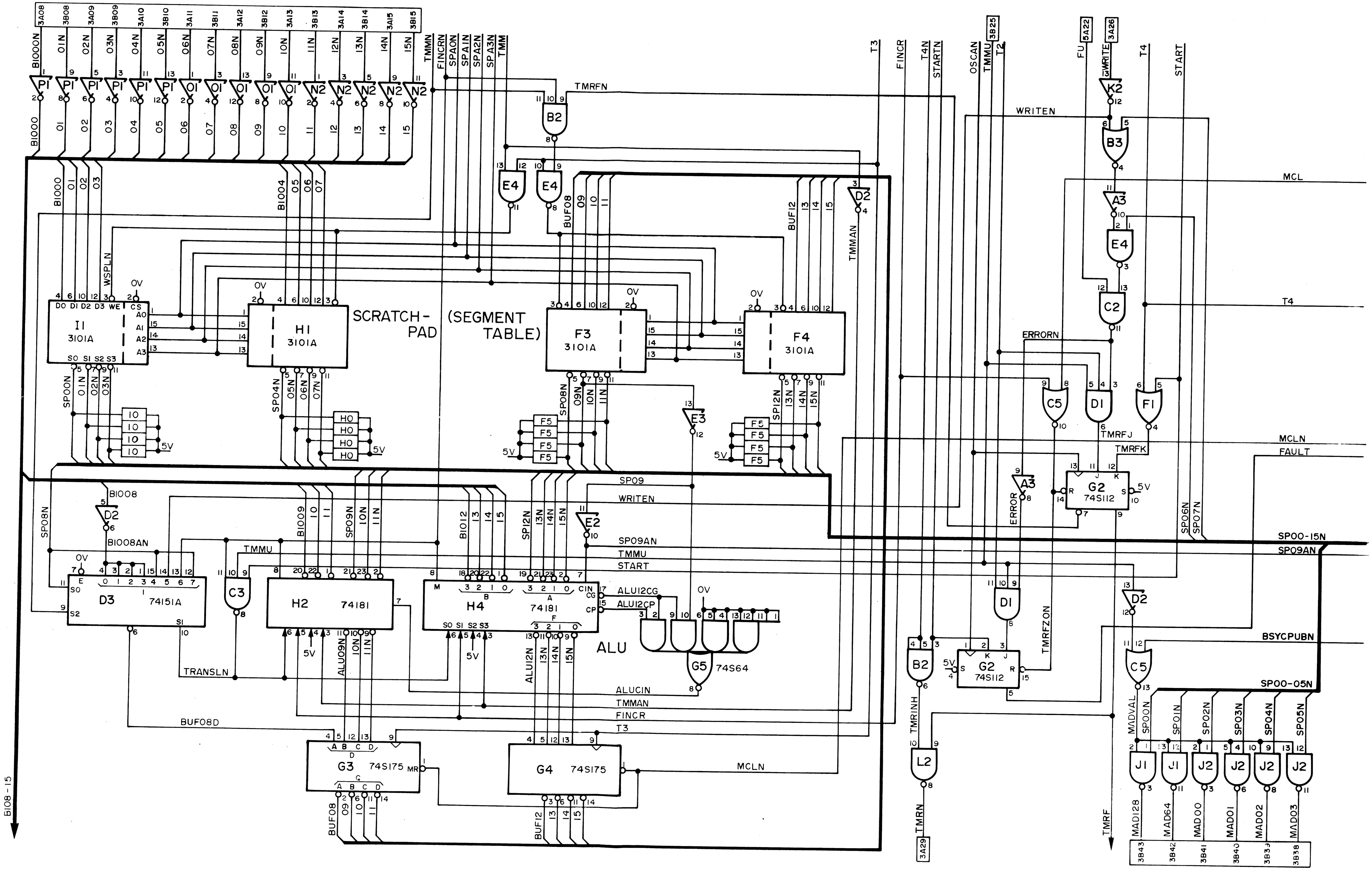


Figure C-10b MMU Logic Diagram

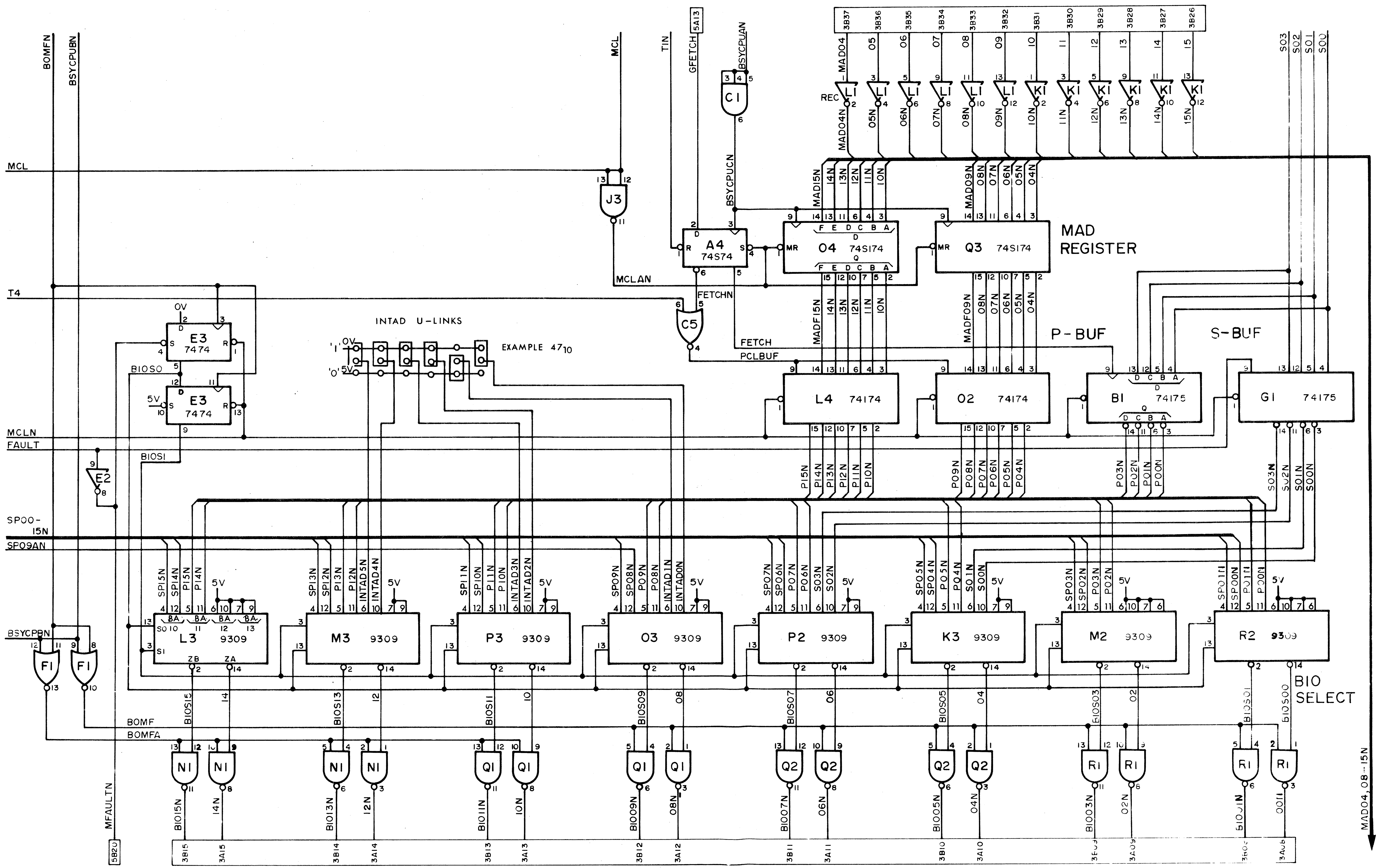


Figure C-10c MMU Logic Diagram

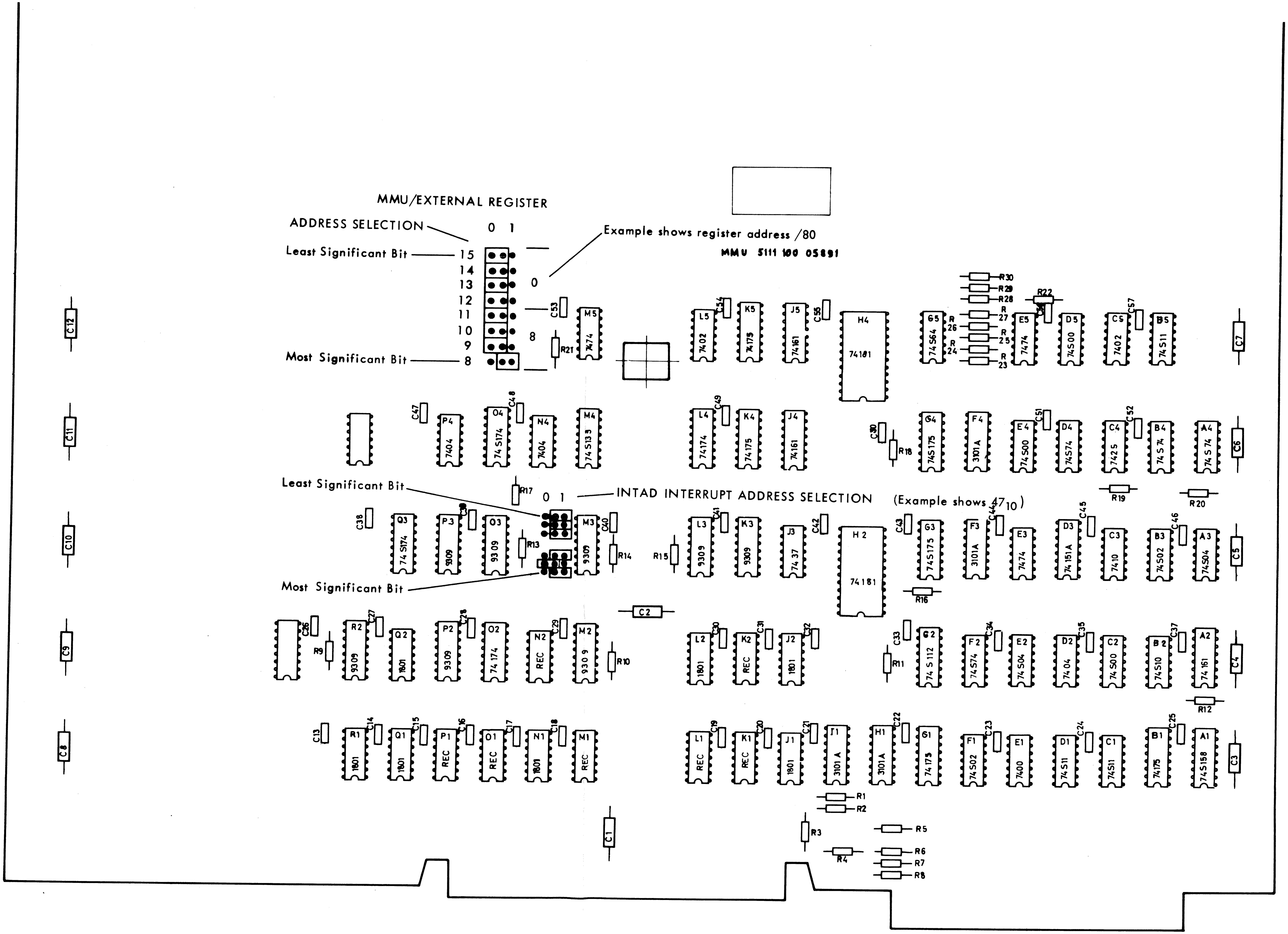


Figure C-11 MMU Card Layout

Table C-1 MMU Parts List

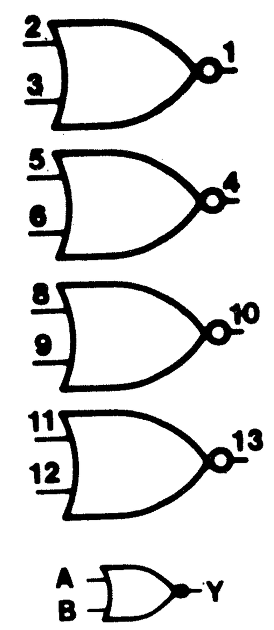
Reference	Description	12NC Code
	Printed circuit	5111 100 05891
A1	Integrated circuit 74S158	
B1 G1 K4 K5.	Integrated circuit 74175	
B5 C1 D1.	Integrated circuit 74S11	
E1.	Integrated circuit 7400	
B3 F1.	Integrated circuit 74S02	
F3 F4 H1 I1.	Integrated circuit 3101A	
J1 J2 L2 N1 Q1 Q2 R1.	Integrated circuit 1801	
K1 K2 L1 M1 N2 O1 P1.	Integrated circuit REC 0613	
A2 J4 J5.	Integrated circuit 74161	
B2.	Integrated circuit 74S10	
C2 D5 E4	Integrated circuit 74S00	
D2 N4 P4.	Integrated circuit 7404	
A3 E2	Integrated circuit 74S04	
A4 B4 D4 F2	Integrated circuit 74S74	
G2	Integrated circuit 74S112	
H2 H4	Integrated circuit 74181	
K3 L3 M2 M3 O3 P2 P3 R2.	Integrated circuit 9309	
L4 O2.	Integrated circuit 74174	
C3.	Integrated circuit 7410	
C4.	Integrated circuit 7425	
C5 L5.	Integrated circuit 7402	
D3.	Integrated circuit 74151A	
E3 E5 M5.	Integrated circuit 7474	
G3 G4.	Integrated circuit 74S175	
G5.	Integrated circuit 74S64	
J3.	Integrated circuit 7437	
M4.	Integrated circuit 74S133	
O4 Q3.	Integrated circuit 74S174	
R9-22.	Resistor 1KΩ, 0.25W, 5%.	
R1-8,23-30.	Resistor 560Ω, 0.25W, 5%.	
C1,2.	Capacitor 47μF, 10V, FITCO.	
C3-12.	Capacitor 10μF, 25V, FITCO.	
C13-35,37-57.	Capacitor 10nF, 20%, ceramic.	
	U Link DCW 06	

1801

(see 7400)

7402

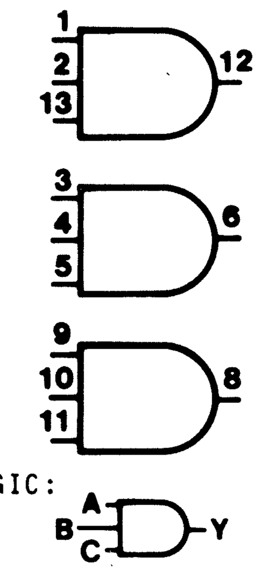
QUADRUPLE 2-INPUT POSITIVE-NOR GATE



Vcc 14
GND 7

7411

7411



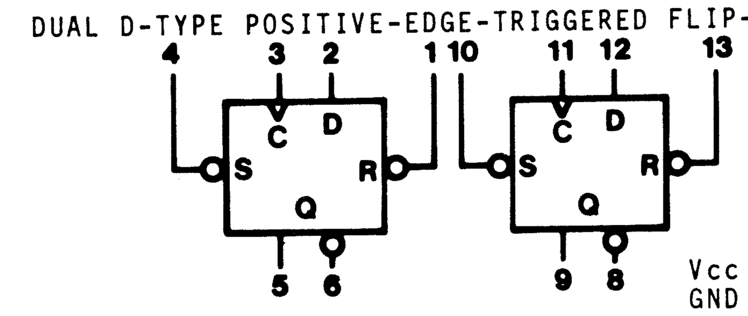
POSITIVE LOGIC:
Y=ABC

Vcc 14
GND 7

7474

7474, H74, S74

DUAL D-TYPE POSITIVE-EDGE-TRIGGERED FLIP-FLOPS

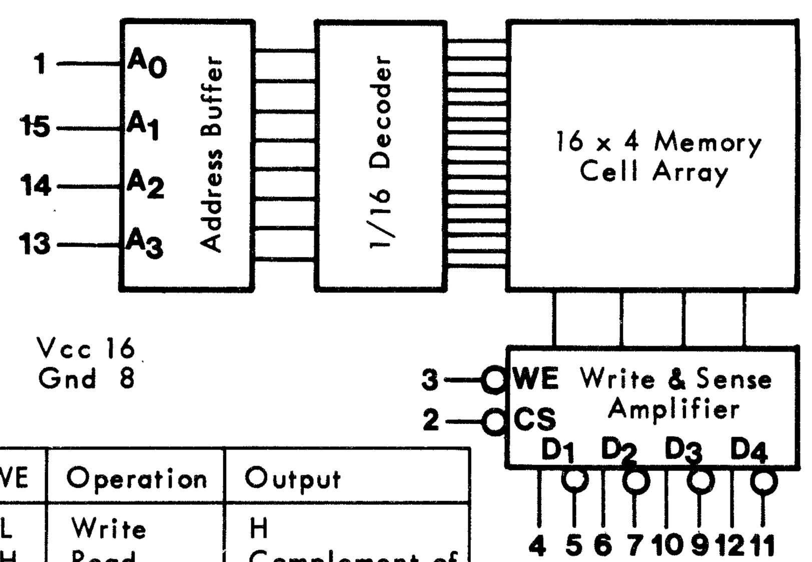


Vcc 14
GND 7

FUNCTION TABLE					
INPUTS				OUTPUTS	
S	R	C	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q_0	Q_0

3101

3101 HIGH SPEED FULLY DECODED 64 BIT MEMORY

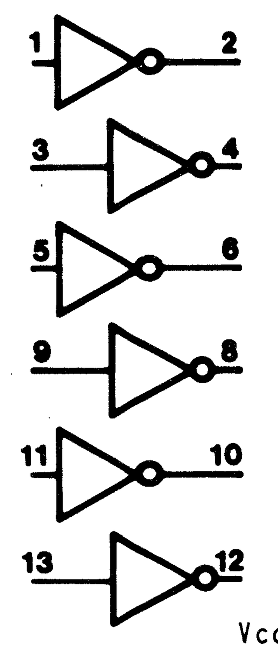


Vcc 16
Gnd 8

CS	WE	Operation	Output
L	L	Write	H
L	H	Read	Complement of Written Data
H	L		H
H	H		H

7404

7404, S04, H04
7406 o.c. 30V
7414 sh Tris
7416 o.c. 15V

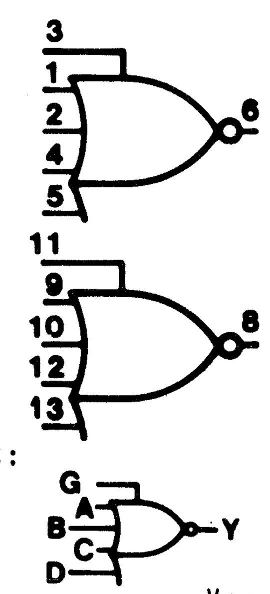


Vcc 14
GND 7

7425

7425

4-INPUT NOR WITH STROBE



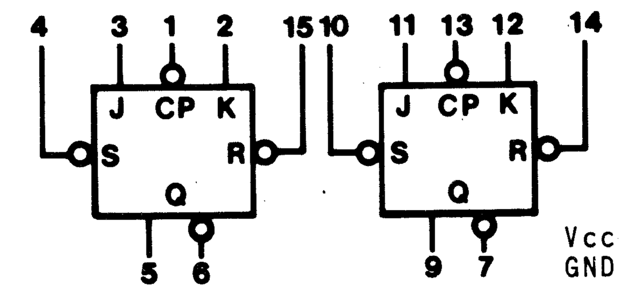
POSITIVE LOGIC:
$Y=\overline{G(A+B+C+D)}$

Vcc 14
GND 7

74112

74S112

J-K NEGATIVE-EDGE-TRIGGERED FLIP-FLOPS



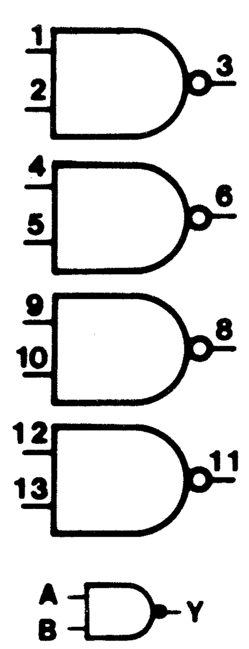
Vcc 16
GND 8

FUNCTION TABLE						
INPUTS					OUTPUTS	
S	R	C	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	H	H
L	L	X	X	X	H	H
H	H	↓	L	L	Qo	Qo
H	H	↓	H	L	H	L
H	H	↓	L	H	L	H
H	H	↓	H	H	TOGGLE	
H	H	H	X	X	Q_0	Q_0

7400

7400
S00
H00
03[1]
37[2]
38
1801[1]

1 open collector
2 buffer

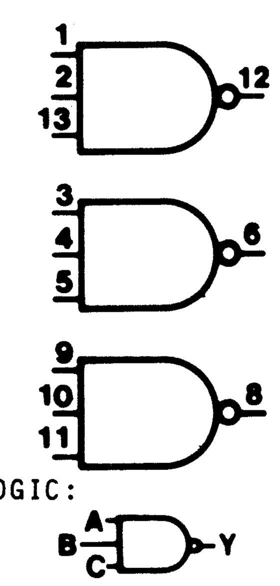


$Y=\overline{AB}$

Vcc 14
GND 7

7410

7410



POSITIVE LOGIC:
$Y=\overline{ABC}$

Vcc 14
GND 7

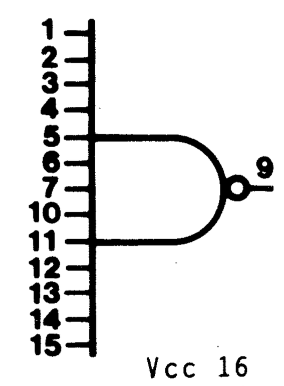
7437

(see 7400)

74133

74S133

13-INPUT NAND GATES

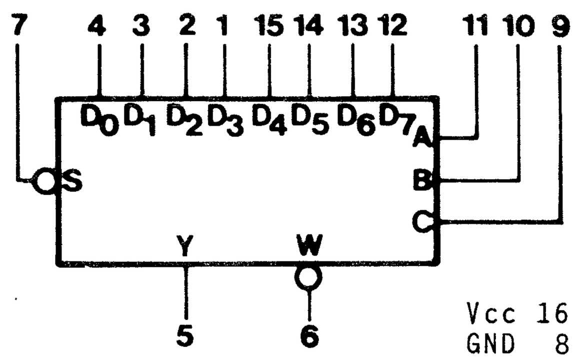


Vcc 16
GND 8

Figure C-12a MMU IC Guide

74151

DATA SELECTOR/MULTIPLEXER



Vcc 16
GND 8

FUNCTION TABLE

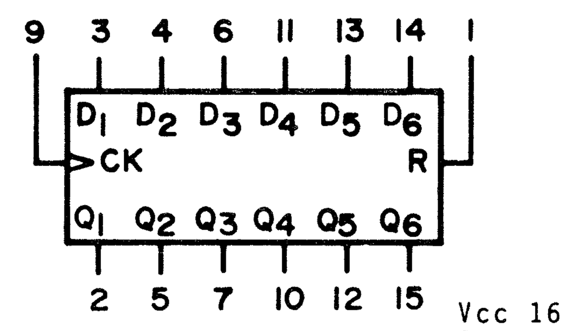
INPUTS		OUTPUTS	
SELECT C B A	STROBE	Y	W
X X X	H	L	H
L L L	L	D0	D0
L L H	L	D1	D1
L H L	L	D2	D2
L H H	L	D3	D3
H L L	L	D4	D4
H L H	L	D5	D5
H H L	L	D6	D6
H H H	L	D7	D7

D0,D1...D7 = the level of the D respective input

74174

74174

HEX D-TYPE FLIP-FLOPS WITH COMMON CLOCK AND RESET

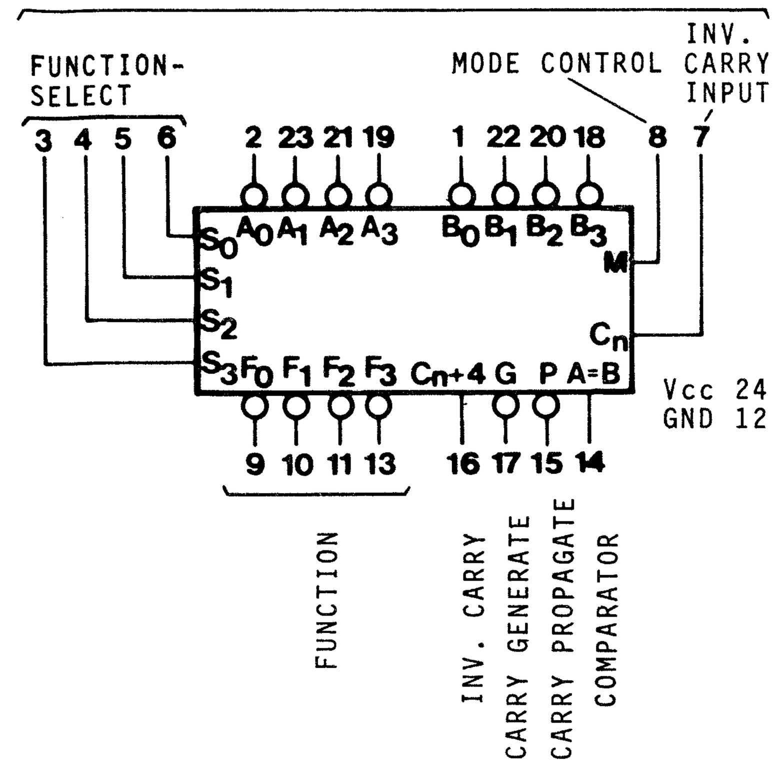


Vcc 16
GND 8

FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\overline{Q}$t
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	$\overline{Q}$o

74181

ARITHMETIC LOGIC UNIT



Vcc 24
GND 12

Table 1

SELECTION	ACTIVE-HIGH DATA		
	M=H	M = L, ARITHMETIC OPERATIONS	
S3 S2 S1 S0	LOGIC FUNCTIONS	Cn = H (no carry)	Cn = L (with carry)
L L L L	$F = \overline{A}$	F = A	F = A plus 1
L L L H	$F = \overline{A + B}$	F = A + B	F = (A + B) plus 1
L L H L	$F = \overline{A}B$	$F = A + \overline{B}$	$F = (A + \overline{B})$ plus 1
L L H H	F = 0	F = minus 1 (2's compl)	F = zero
L H L L	$F = \overline{AB}$	$F = A$ plus $A\overline{B}$	$F = A$ plus $A\overline{B}$ plus 1
L H L H	$F = \overline{B}$	$F = (A + B)$ plus $A\overline{B}$	$F = (A + B)$ plus $A\overline{B}$ plus 1
L H H L	$F = A \oplus B$	F = A minus B minus 1	F = A minus B
L H H H	$F = A\overline{B}$	$F = A\overline{B}$ minus 1	$F = A\overline{B}$
H L L L	$F = \overline{A} + B$	F = A plus AB	F = A plus AB plus 1
H L L H	$F = \overline{A \oplus B}$	F = A plus B	F = A plus B plus 1
H L H L	F = B	$F = (A + \overline{B})$ plus AB	$F = (A + \overline{B})$ plus AB plus 1
H L H H	F = AB	F = AB minus 1	F = AB
H H L L	F = 1	F = A plus A*	F = A plus A plus 1
H H L H	$F = A + \overline{B}$	F = (A + B) plus A	F = (A + B) plus A plus 1
H H H L	F = A + B	$F = (A + \overline{B})$ plus A	$F = (A + \overline{B})$ plus A plus 1
H H H H	F = A	F = A minus 1	F = A

Table 2

SELECTION	ACTIVE-LOW DATA		
	M=H	M = L, ARITHMETIC OPERATIONS	
S3 S2 S1 S0	LOGIC FUNCTIONS	Cn = L (no carry)	Cn = H (with carry)
L L L L	$F = \overline{A}$	F = A minus 1	F = A
L L L H	$F = \overline{AB}$	F = AB minus 1	F = AB
L L H L	$F = \overline{A} + B$	$F = A\overline{B}$ minus 1	$F = A\overline{B}$
L L H H	F = 1	F = minus 1 (2's comp)	F = zero
L H L L	$F = \overline{A + B}$	$F = A$ plus $(A + \overline{B})$	$F = A$ plus $(A + \overline{B})$ plus 1
L H L H	$F = \overline{B}$	$F = AB$ plus $(A + \overline{B})$	$F = AB$ plus $(A + \overline{B})$ plus 1
L H H L	$F = \overline{A \oplus B}$	F = A minus B minus 1	F = A minus B
L H H H	$F = A + \overline{B}$	$F = A + \overline{B}$	$F = (A + \overline{B})$ plus 1
H L L L	$F = \overline{A}B$	F = A plus (A + B)	F = A plus (A + B) plus 1
H L L H	$F = A \oplus B$	F = A plus B	F = A plus B plus 1
H L H L	F = B	$F = A\overline{B}$ plus (A + B)	$F = A\overline{B}$ plus (A + B) plus 1
H L H H	F = A + B	F = A + B	F = (A + B) plus 1
H H L L	F = 0	F = A plus A*	F = A plus A plus 1
H H L H	$F = A\overline{B}$	F = AB plus A	F = AB plus A plus 1
H H H L	F = AB	$F = A\overline{B}$ plus A	$F = A\overline{B}$ plus A plus 1
H H H H	F = A	F = A	F = A plus 1

* Each bit is shifted to the next more significant position.

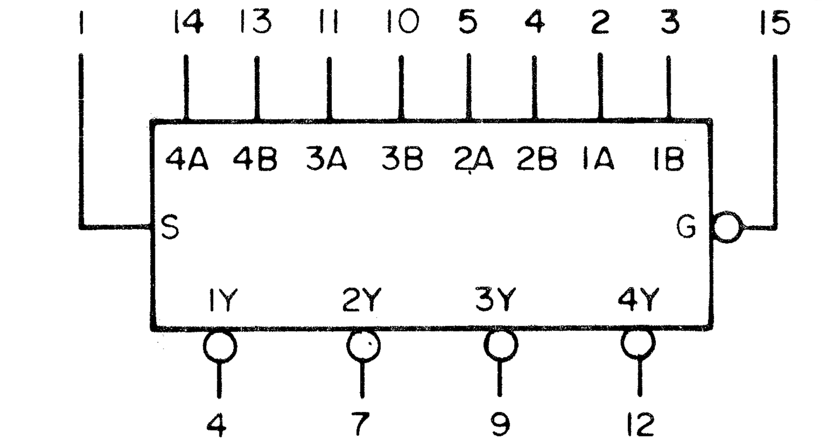
REC 0613

(see 7404)

74158

74158

QUAD 2-to-1-LINE DATA SELECTORS/MULTIPLEXERS



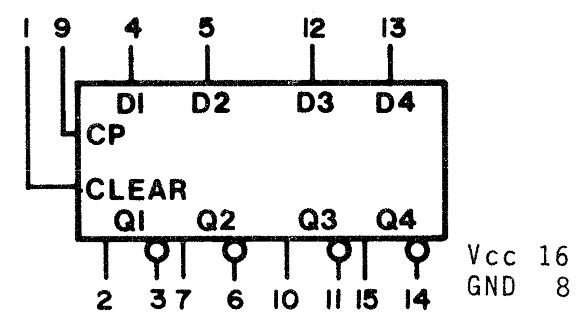
Vcc 16
GND 8

OUTPUTS

74175

74175

QUAD D-TYPE FLIP-FLOPS



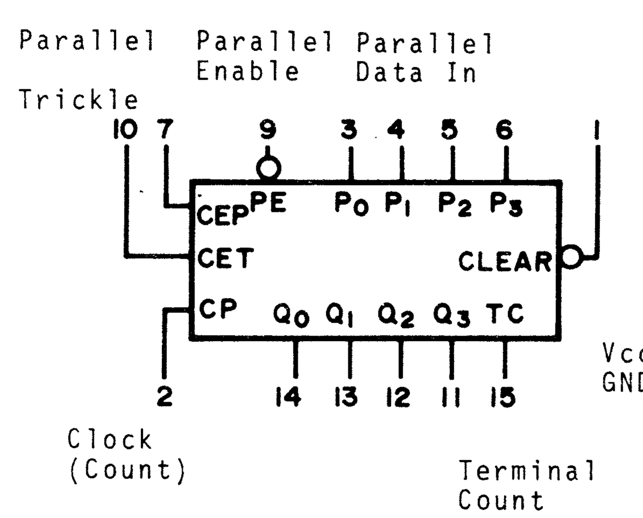
Vcc 16
GND 8

FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\overline{Q}$
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	Qo

74161

9310, 74161

BCD COUNTER
BINARY COUNTER

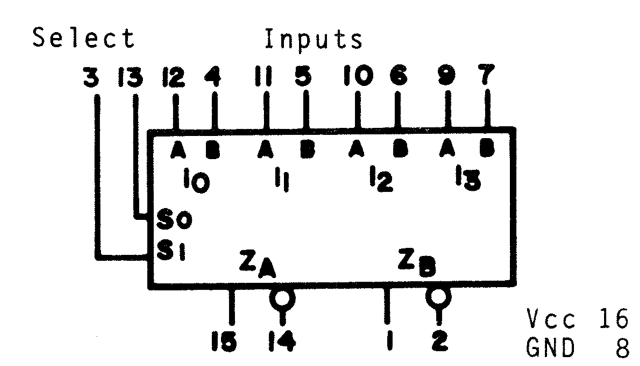


Vcc 16
GND 8

9309

9309

DUAL, 4-INPUT MULTIPLEXER



Vcc 16
GND 8

APPENDIX D

FLOATING POINT PROCESSOR

SERVICE MANUAL

Effective Pages March 1977

ii	through	iv
D.1	through	D-28

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

APPENDIX D

FLOATING POINT PROCESSOR

D.1 GENERAL

The Floating Point Processor (FPP) performs floating-point arithmetic operations by providing an extension to the CPU arithmetic section. The FPP comprises a single card located in a dedicated slot of the main chassis. The FPP operates with the CPU and the memory via the standard GP Bus, however, some dedicated wiring between the FPP and the CPU is used to increase operating speed.

D.2 LOGIC LAYOUT

The floating-point operations are done between the floating-point accumulator (in the FPP) which contains the first operand, and the memory effective address which contain the second operand. The result is stored into the FPP accumulator or into the memory address, depending on the instruction Store indicator. Floating-point conversions are done between the FPP accumulator and the CPU internal registers A1, A2.

D.3 The FPP logic (Figure D-1) performs complete parallel operation on all data. Data are transferred in and out of the FPP one word (16 bits) at a time. The single-word exponent is loaded into the Exponent-Logic section and the double-word mantissa (or double-precision integer) is loaded into the Mantissa-Logic section. Processing is then done on the 48 bits in parallel. The FPP logic is completely microprogram controlled. The Command/Timing logic section contains the FPP sequensor, instruction decode and control logic, status register, and most FPP input/output control signals.

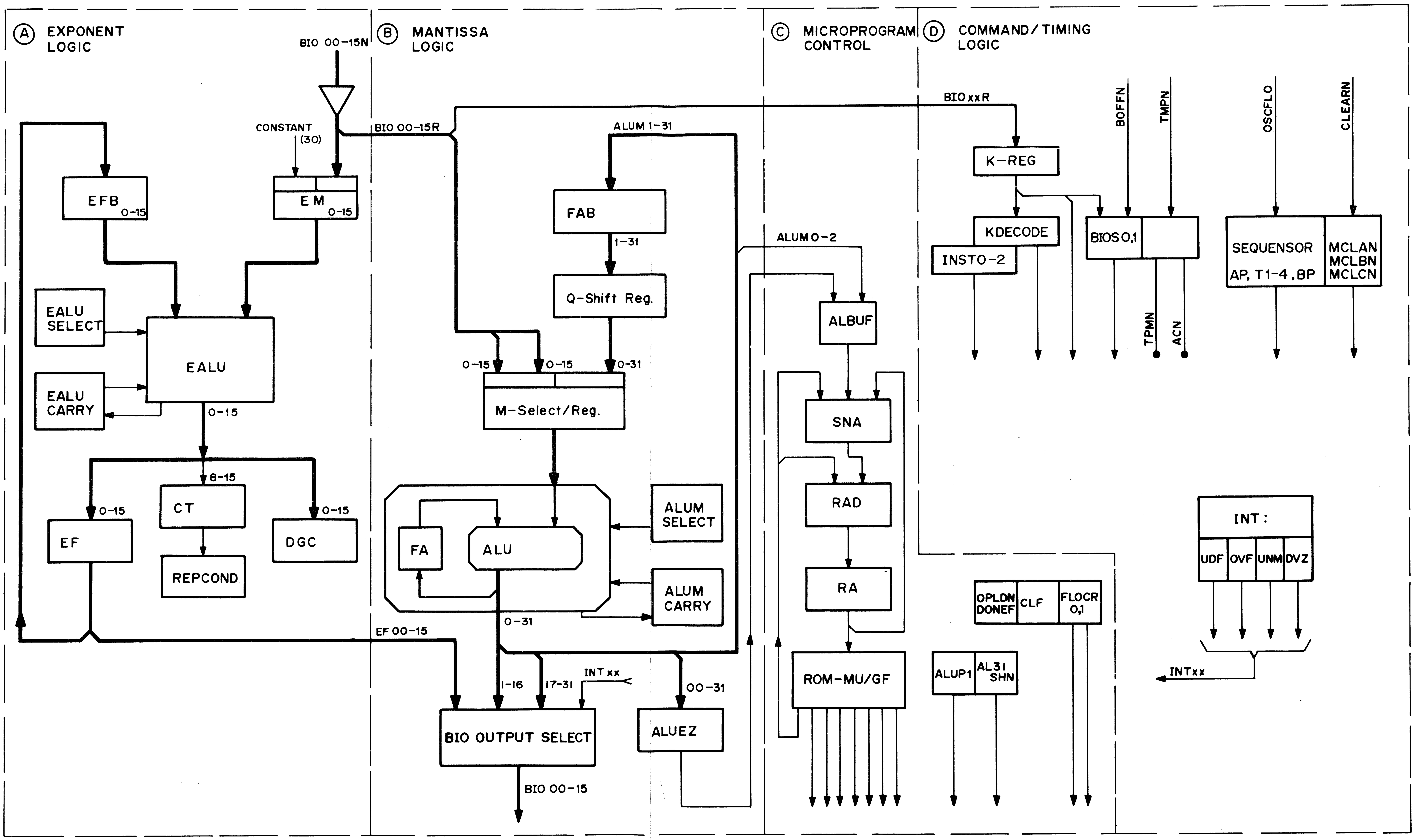


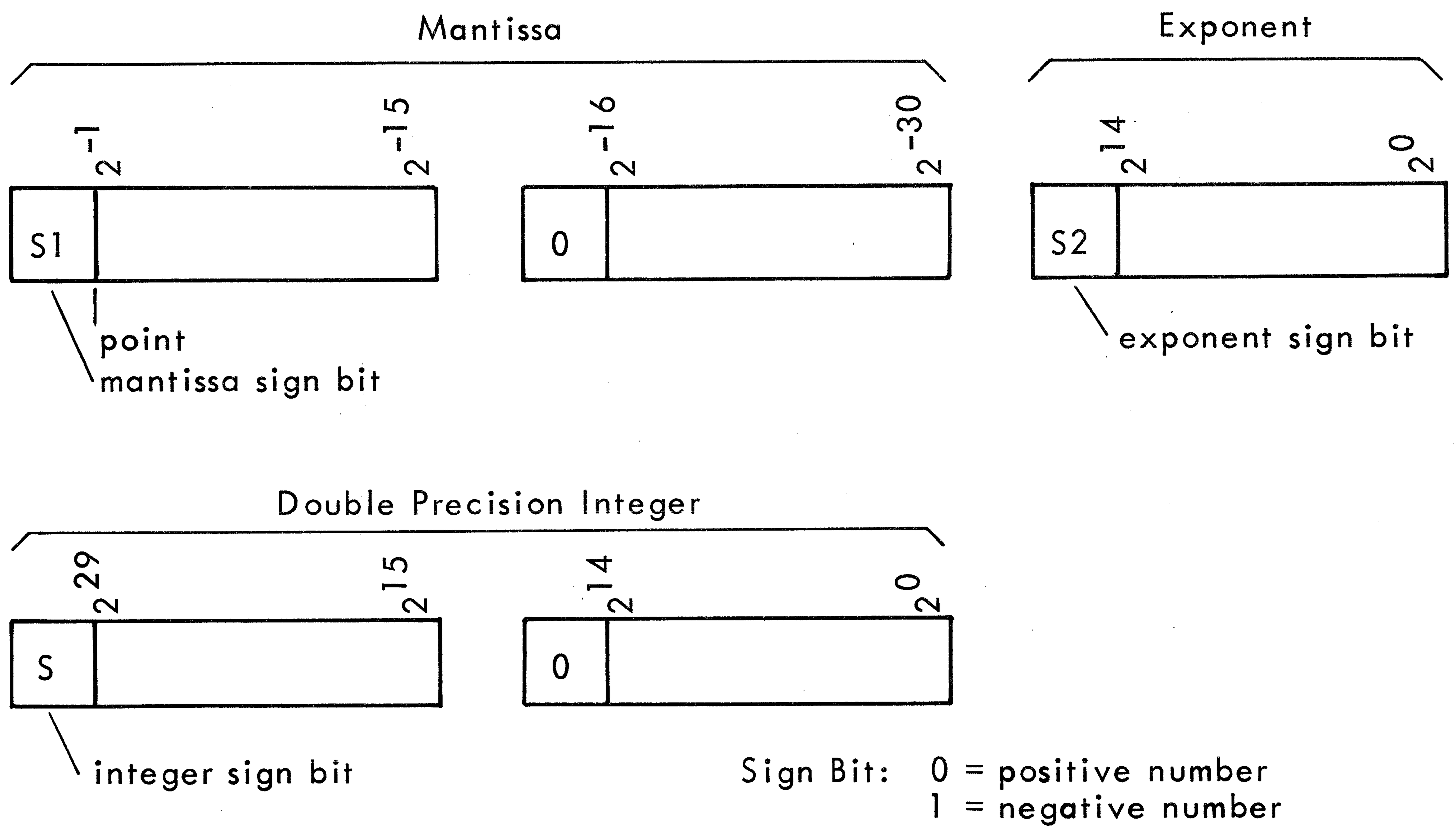
Figure D-1 FPP Block Diagram

D.4 INSTRUCTIONS

The FPP is operated by special floating-point instructions during the normal CPU programmed operation. The floating-point instruction is controlled by the CPU which is the Master of the dialogue and allows the FPP to execute the arithmetic operations. If the FPP is not installed in the system, an attempted floating-point instruction is trapped and a software subroutine may be initiated (if the subroutine is not directly called by a Call Function instruction). The command codes and functions of all FPP instructions are shown in Figure D-2.

D.5 DATA FORMAT

The two types of data handled by the FPP are floating-point data and double-precision integer data (following diagram). The FPP performs arithmetic operations on floating-point data only, and performs conversions between floating-point and integer data.



D.6 Floating-Point Data

Floating-point data are real numbers contained in three 16-bit words: a double-word mantissa and a single-word exponent.

D.7 The mantissa is a fraction (<1), with the point considered to be at the extreme left of the data quantity bits. The mantissa must be left normalized, thus, the left data bit is a 1 for positive numbers (sign bit 0) and a 0 for negative numbers (sign bit 1). Input data are checked for normalization (by comparing the left-most data bit with the sign bit) and the result of any arithmetic operation is normalized. The numerical range of the mantissa is given in Figure D-3.

D.8 The exponent is a single-precision integer with a range of -2^{15} to $+2^{15}-1$. The range of the exponent and the range of the complete floating-point number (mantissa with exponent) is shown on Figure D-3.

D.9 A floating-point data Zero is represented by three null words (mantissa and exponent). The FPP recognizes as Zero any data with a null mantissa, regardless of the value of the exponent. The FPP produces a null result under the following conditions:

- during FAD/FSU if both received and stored operands are null.
- during FMU if either received or stored operand is null.
- during FLD if the received operand is null.
- during FDV if the stored operand (dividend) is null.

If the received operand (divisor) for an FDV command is null, the FPP sets Divide-by-Zero error status and does not change the stored operand.

D.10 The mantissa of all floating-point operands, other than null, must be normalized. If any unnormalized mantissa is received, the FPP sets Unnormalized-Operand error status and does not change the stored operand. All FPP operation results other than null are normalized.

D.11 Integer Data

The integer (Figure D-3) is a double-precision whole number with a sign bit and 30 quantity bits. The range is from -2^{30} to $+2^{30}-1$.

Instruction (Addressing Mode)	Format	Function	Explanation
FFL - Integer to F.P. (T1)	0 4 5 9 11 15 1 1 0 0 | 1 0 0 1 | 0 0 0 | 0 0 0 0 | 0	(A1),(A2) → FPA	(CPU) → FPP 30 Double-precision integer in CPU A1,A2 sent to FPP, converted to floating-point data.
FFX - F.P. to Integer (T1)	1 1 0 0 | 1 0 0 1 | 0 0 0 | 0 0 0 0 | 1	(FPA) → A1,A2	(FPP) → CPU e Floating-point data in FPP accumulator converted to double-precision integer, sent to CPU A1,A2. If exponent > 30 conversion not done (Overflow)
FLD,R - F.P. Load (T4-T7,T3)	1 1 0 0 | 0 0 0 1 0 | MD | R2 | 0 M (except T3)	(EA) → FPA	(Memory) (FPP) e e Floating-point operand from memory loaded into FPP accumulator.
FST,R - F.P. Store (T4-T7,T3)	1 1 0 0 | 0 0 0 1 0 | MD | R2 | 1 M (except T3)	(FPA) → EA	(FPP) (Memory) e e Floating-point data from FPP accumulator stored in memory.
FAD,R - F.P. Add FSU,R - F.P. Subtract FMU,R - F.P. Multiply FDV,R - F.P. Divide (T4-T7,T3)	0 4 5 9 11 15 1 1 0 0 | 1 | R1 | MD | R2 | S M (except T3) R1: 1 0 0 0 / 1 0 1 0 / 1 1 0 0 / 1 1 1 0	S = 0: (EA)*(FPA)→FPA S = 1: (EA)*(FPA)→EA + - × ÷	e e (Memory) (FPP) + - × ÷ S = 1 S = 0 Result Arithmetic operation performed on two floating-point operands: one from memory and one from FPP accumulator; result stored in memory or FPP (S = 1 or 0).

FPA: Floating Point Accumulator in FPP
EA : Memory Effective Address — points to first of three consecutive words in memory.
F.P.: Floating Point

Shaded part ▒ not sent to FPP

Type	MD	R2	Effective Address
T3	0 1	≠0	in CPU register R2
T4	1 0	=0	in next word
T5	1 0	≠0	indexed
T6	1 1	=0	indirect
T7	1 1	≠0	indirect indexed

CPU Condition Register	
CR0:	result is zero
CR1:	result is positive
CR2:	result is negative
CR3:	abnormal condition detected.

Figure D-2 FPP Instructions

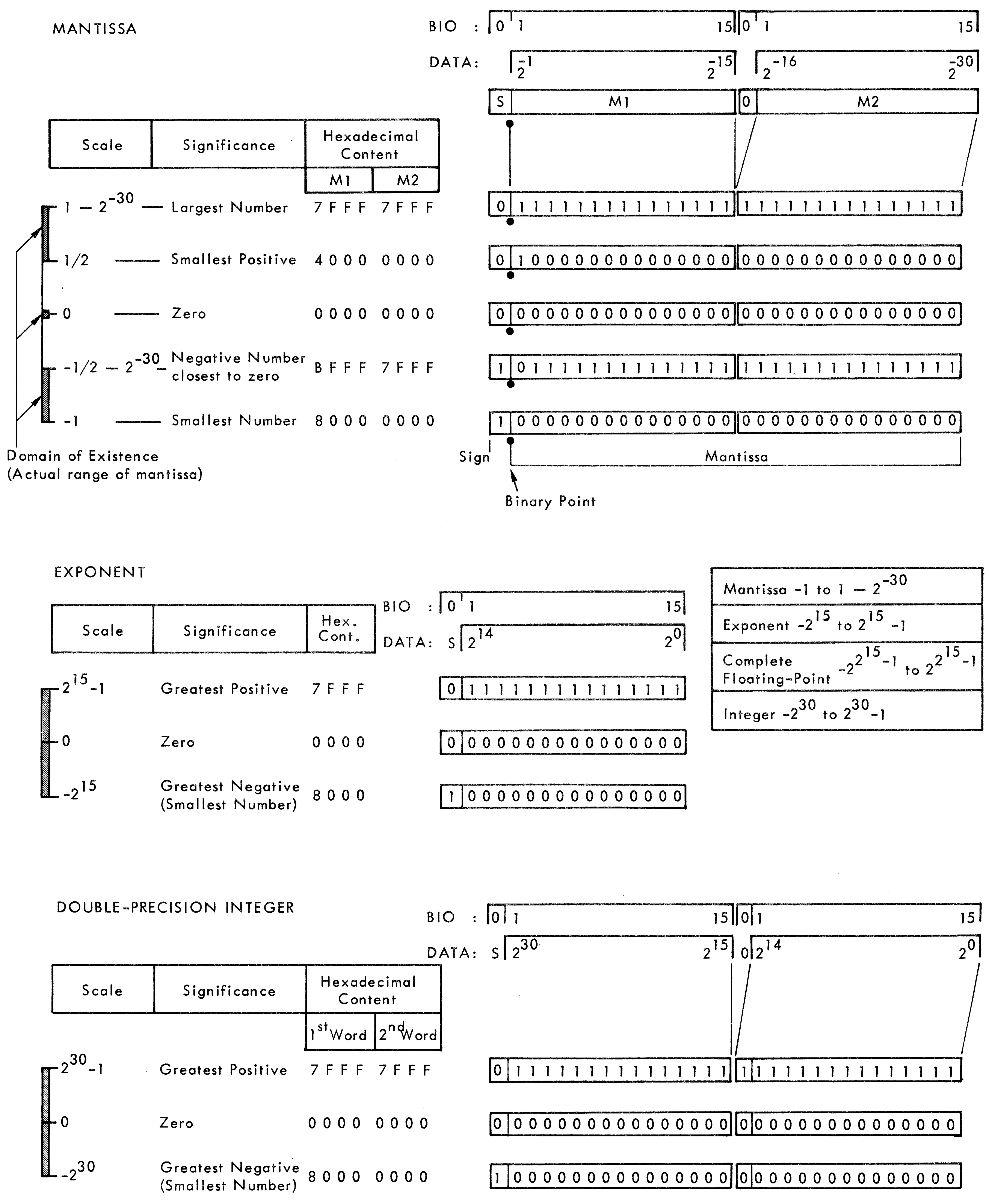


Figure D-3 Format and Range of Numbers

D.12 OPERATIONAL FLOW

The FPP logic is microprogram controlled. Figure D-4 is a flow diagram of all FPP operations. Each microinstruction is represented by a single block, with the microprogram address shown (in hexadecimal) in the upper-left corner of each block.

D.13 The paths from one microinstruction to another are controlled by the microprogram address control (paragraph D.25). Each microinstruction specifies the address of the next microinstruction, either directly (explicit address) or according to certain branch conditions: IDLE, TEFL, OPER, FIX, RESULT. These branch conditions are shown in Table D-1. The contents of the Address ROM are listed in Table D-2.

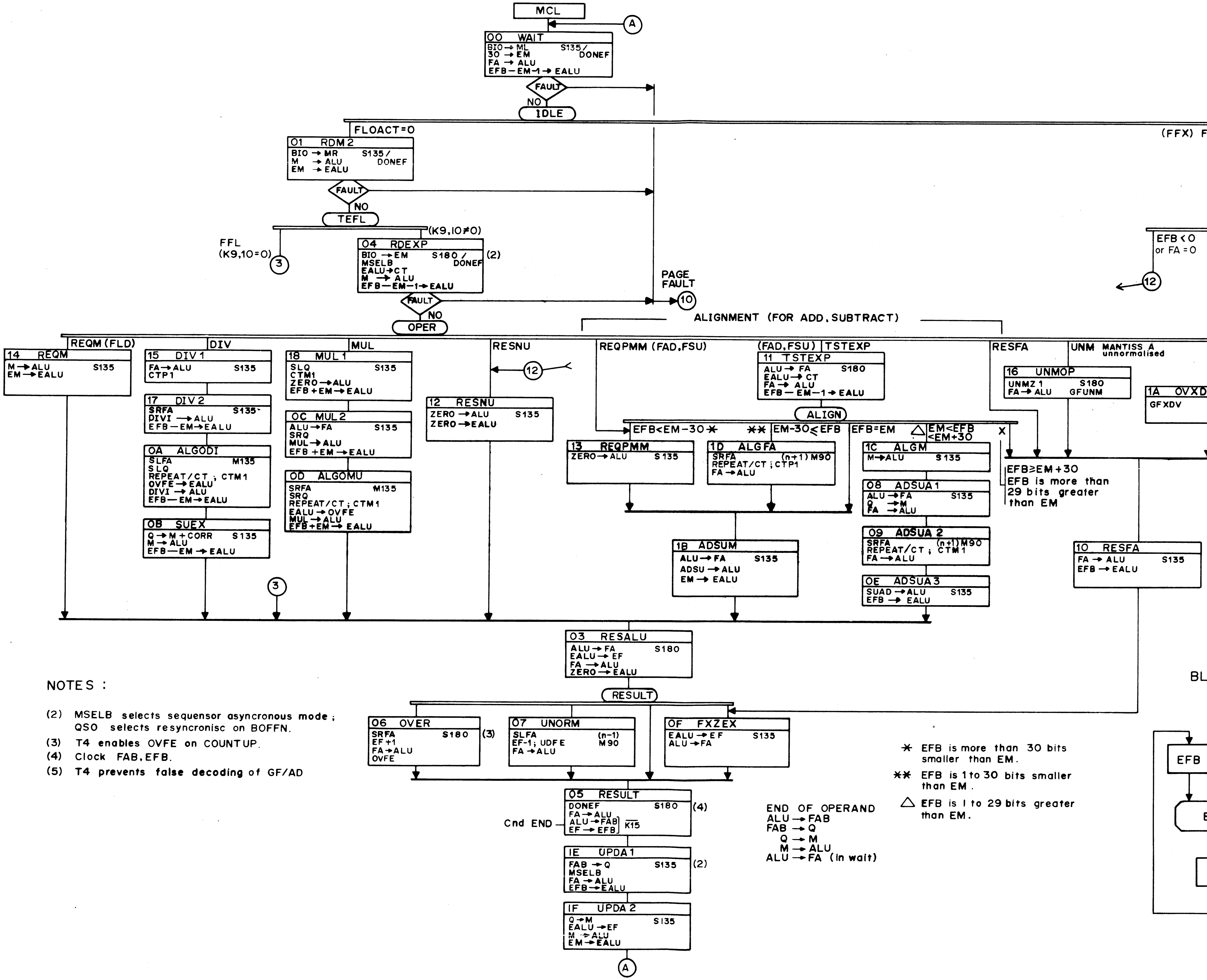
D.14 Start of Commands

The FPP goes to the Wait (/00) microinstruction upon the completion of any previous command or when the system Master Clear (MCL) is pressed. The FPP stays at Wait until a command is received from the CPU.

D.15 FFL Command

For the FFL Command, the first part of the mantissa is loaded into the M-register left half during microinstruction Wait (/00). A constant of 30 is loaded into the exponent register EM, and then cycled through EALU to EFB.

D.16 The flow branches via IDLE to microinstruction RDM2 (/01) where the second half of the mantissa is loaded into M-register right half. The complete M register is then loaded into the ALU. The flow now branches via TEFL to microinstruction RESALU (/03). The complete mantissa is loaded from the ALU into floating-point accumulator FA. The exponent of 30 is loaded into register EF, while EALU is reset to zero. The flow branches via RESULT to analyze the result of the operation.



NOTES :

(2) MSELB selects sequensor asyncronous mode ; QSO selects resyncronisc on BOFFN.

(3) T4 enables OVFE on COUNTUP.

(4) Clock FAB, EFB.

(5) T4 prevents false decoding of GF/AD

* EFB is more than 30 bits smaller than EM.

** EFB is 1 to 30 bits smaller than EM.

△ EFB is 1 to 29 bits greater than EM.

BLOCK DIAGRAM

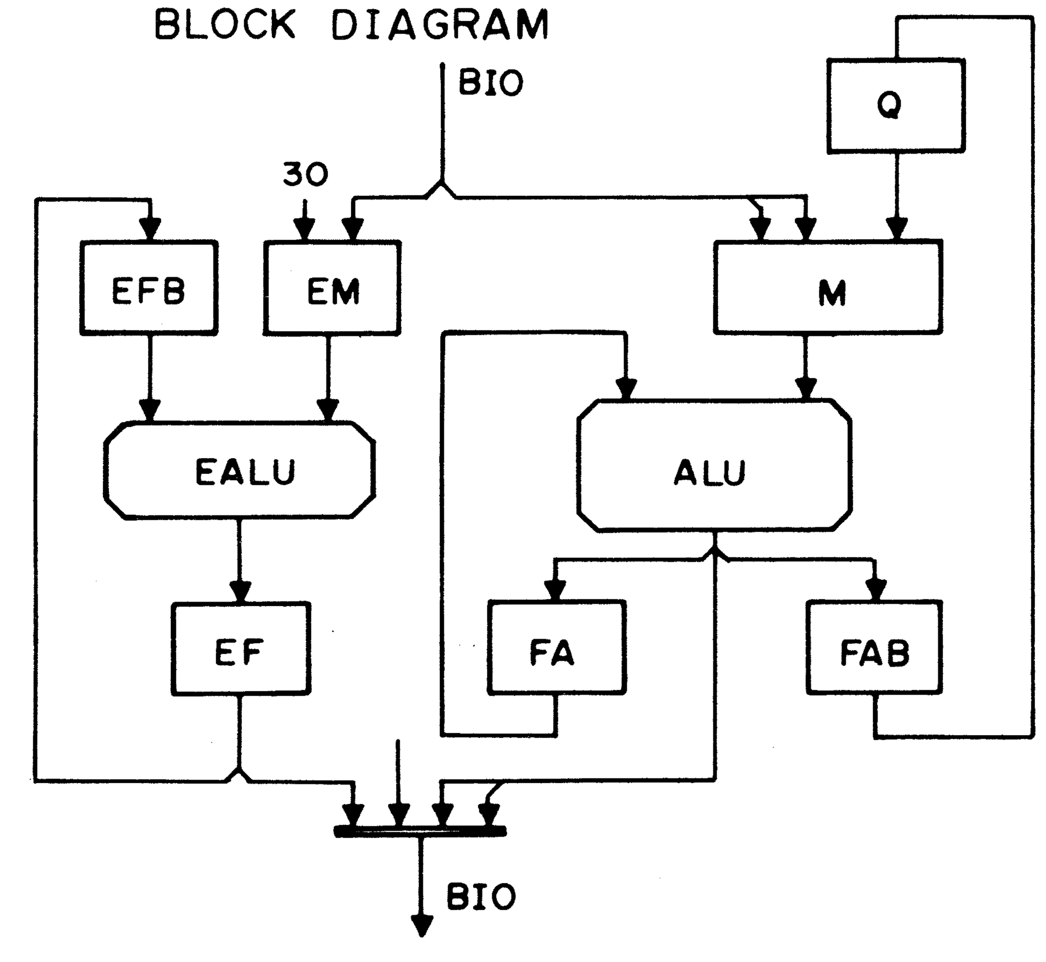


Figure D-4 FPP Flow Diagram

Table D-1 Next Address Branch Conditions (A)

NEXT ADDRESS GENERATION (NAGN) CONTROL BITS

Type		SEL Code	Address Source (active low)					T4
		μSNA0-1-2	SNA3	SNA4	SNA5	SNA6	SNA7	SEQ
EXPLICIT 0		0 0 0	RA0	RA1	RA2	RA3	RA4	-
EXPLICIT 1		1 0 0						
BRANCH	IDLE	0 0 1	0	0	0	0	FLOACT	-
	TEFL	0 1 0	0	0	0	K09	K10	-
	RESULT	0 1 1	0	ALBUFZ	ALBUF0	ALBUF1	ALBUF2	*
	FIX	1 0 1	EFB00	ALBUFZ	EALU00	EQM1	0	-
	ALIGN	1 1 0	EFB00	EM00	EALU00	EQM1	DGC	*
	OPER	1 1 1	Q31D	ALBUFZ	KDVFLD	K06	ALBUF2Z	*

FLOACT = 1 : FFX instruction
ALBUFZ = 1 : (M) = 0; received mantissa is zero
ALBUF0 : overflow bit
ALBUF1 : mantissa sign bit
ALBUF2 : first data bit (2^{-1}) verifies if mantissa is normalized
ALBUF2Z = 1: (FA) = 0; stored mantissa is zero
Q31D = 1 : sign-bit and 1st-bit un-alike; operand is normalized
K06 = 0 : Add/Subtract; K06 = 1: multiply/divide
DGC = 1 : Data Greater than Constant (30)
EQM1 = 1 : Both operands have the same exponent

IDLE

Control	Next Address
FLOACT (Bus Timing from CPU)	
0	01 - RDM2
1	02 - FFX

TEFL

Control	Next Address
K09, 10	
0 0	03-RESALU (if FFL)
0 1	04-RDEXP
1 0	
1 1	

Table D-1 Next Address Branch Conditions (B)

OPER

Control					Operation		Next Address
(M0xM1) Q31D	ALBUFZ (mant. A=0)	Command Decode KDVFLD	K06	ALBUF2Z (mant. B=0)			
0	0	X	X	X	Mantissa unnormalized ERROR Q31D=M0xM1=0		16-UNMOP
0	1	0	0	0	FADSU	(Operand Null)	10-RESFA
0	1	0	0	1			12-RESNU
0	1	0	1	0	FMU		
0	1	0	1	1			
0	1	1	0	0	FLD		
0	1	1	0	1			
0	1	1	1	0	FDV		1A-OVXDV
0	1	1	1	1			
1	0	0	0	0	FADSU		11-TSTEXP
1	0	0	0	1			13-REQPMM
1	0	0	1	0	FMU		18-MUL1
1	0	0	1	1			12-RESNU
1	0	1	0	0	FLD		14-REQM
1	0	1	0	1			
1	0	1	1	0	FDV		15-DIV1
1	0	1	1	1			12-RESNU
1	1	X	X	X			

Q31D = 1 : sign-bit and 1st-bit un-alike; operand is normalized.
ALBUFZ = 1 : (M) = 0; received mantissa-A is zero.
ALBUF2Z = 1: (FA) = 0; stored mantissa-B is zero.
K06 = 0 : Add/Subtract; K06 = 1: Multiply/Divide

FIX

Control					Operation	Next Address
EFB0	ALBUFZ	EALU00	EAEB (EQM1)	SNA7		
0	0	0	0	0	EFB > 30	1A-OVXDV (EFB-30-1>0)
0	0	0	1	0	EAEB=1 ⇒ EALU0≠1	
0	0	1	0	0	0≤EFB < 30	19-DENORM
0	0	1	1	0	EFB=30	10-RESFA
0	1	X	0	0	Mantissa = 0	12-RESNU
0	1	0	1	0		
0	1	1	1	0	EFB=30	12-RESNU
X	X	X	X	1		
1	X	X	X	X		12-RESNU

Exponent < 0 (negative); number < 1 absolute value; may not be changed to fixed-point number.

Table D-1 Nest Address Branch Conditions (C)

ALIGN

Control					Compare Exponent	Next Address
EFB0	EM00	EALU00	EAEB	DGC		
0	0	0	0	0	EM < EFB < EM+30	1C-ALGM
X	1	0	0	0		
0	X	0	0	1	EM+30 ≤ EFB	10-RESFA
0	1	1	0	X	EM ≤ EFB-2^{15}	
1	1	0	0	1	EM+30 ≤ EFB < 0	
0	0	0	1	X		if EALU00 = 0 ⇒ EAEB = 0
0	0	1	0	0	EFB < EM-30	13-REQPMM
1	0	1	0	0		
1	1	1	0	0		
1	0	0	0	X	EFB < EM-2^{15}	
0	0	1	0	1	EM-30 ≤ EFB < EM	1D-ALGFA
1	0	1	0	1		
1	1	1	0	1		
0	0	1	1	0		if EAEB = 1 ⇒ DGC = 0
0	0	1	1	1	0 ≤ EFB = EM	1B-ADSUM
1	1	1	1	1	EFB = EM < 0	
0	1	1	1	1		EF00 ≠ EM00 ⇒ EAEB = 0
X	1	0	1	X		
X	1	1	1	0		
1	0	X	1	X		

DGC = 1 : Data Greater than Constant (30).
EAEB = 1 : Exponents are the same.
EALU00 = 1: Output of EALU ≤ Constant (30).

RESULT

Control		Operation	Next Address
ALBUFZ	ALBUF0,1,2		
0	0 0 0		07-UNNORM
0	1 1 1		
0	0 0 1		05-RESULT
0	1 1 0		
0	0 1 X	Overflow	06-OVF
0	1 0 X		
1	0 0 0	Result = zero	OF-FXZEX
1	(≠0)		

Overflow occurs when the two most-significant bits are different after the 32 bits are adjusted right, for operands between -2^{30} and $+2^{30}-1$ inclusive (limit of fixed-point numbers).

ALBUFZ = 1: mantissa is zero
ALBUF0 : overflow bit
ALBUF1 : mantissa sign bit
ALBUF2 : first data bit (2^{-1}) verifies if mantissa is normalized

Table D-2 Address ROM (RAD) Listing

FLOATING POINT ADDRESS STORE

FPAS FORM 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 16 4-BIT WORDS

FAPIP DATA /00FF LEADING FF

FNAXOL	FPAS	0 0 0 8 3 0 0 C A 0 0 3 0 3 0 F	EXPLICIT ZERO
FNAXOH	FPAS	5 3 3 D 3 B E 9 5 5 0 0 0 0 0 0	
FNAIDL	FPAS	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	IDLE
FNAIDH	FPAS	0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 1	
FNARML	FPAS	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	RDM2
FNARMH	FPAS	0 0 0 0 0 0 0 0 0 0 0 0 4 4 4 3	
FNAREL	FPAS	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	RESULT
FNAREH	FPAS	0 0 0 0 0 0 0 F 7 5 6 6 6 6 5 7	
FNAXIL	FPAS	0 F B 0 0 0 0 0 0 0 7 0 B 0 0 0	EXPLICIT ONE
FNAXIH	FPAS	0 0 0 0 0 0 0 0 0 0 E 0 0 0 0 0	
FNAFXL	FPAS	0 2 0 2 0 2 0 2 0 0 0 2 0 0 0 2	FIX
FNAFXH	FPAS	0 2 0 2 0 2 0 2 0 0 0 9 0 0 0 A	
FNAAGL	FPAS	B 0 D 3 0 0 0 C 0 0 D 3 0 0 3 3	ALIGNMENT
FNAAGH	FPAS	0 0 0 0 0 0 0 C B 0 D 3 0 0 0 C	
FNAOPL	FPAS	0 0 0 0 0 0 0 0 2 5 4 4 2 8 3 1	OPERATION
FNAOPH	FPAS	A A 2 2 2 2 2 0 6 6 6 6 6 6 6 6	
		0 1 2 3 4 5 6 7 8 9 A B C D E F	

D.17 FFX Command

For the FFX command, no data is loaded during the Wait microinstruction (/00). A constant of 30 is loaded into exponent register EM. This constant, plus one, is then subtracted from the exponent in EFB and the difference is placed in EALU. The CPU command signal FLOACT is received as part of the FFX command, and the flow branches via IDLE to microinstruction FFX (/02). The exponent difference from 31 is loaded into the CT counter, and the mantissa is loaded into accumulator FA.

D.18 The branch condition FIX tests the exponent to determine if and how the mantissa must be modified for conversion to a double-precision integer:

- If either the mantissa (FA) is zero or the exponent (EFB) is negative, the flow branches to RESNU (/12) where both registers are reset to zero. This Reset-Null operation is completed in the RESALU (/03) microinstruction as the flow moves to the RESULT branch.
- If the exponent (EFB) is precisely 30, the mantissa can be used directly, without any change. The flow branches directly to microinstruction RESFA (/10).
- If the exponent (EFB) is greater than 30, the conversion to an integer is not done. The flow branches to microinstruction RESFA (/10), via microinstruction OVXDV (/1A) where the Overflow interrupt is set.
- If the exponent (EFB) is between 1 and 29 inclusive, the DENORM (/19) microinstruction is performed to convert the mantissa to an integer with the number of significant digits specified by the exponent. The mantissa is shifted right, end-off; the number of shifts is controlled by the CT counter which is loaded by the Wait microinstruction at the start of the FFX command.

The two FFX operations, other than the Null and the Overflow, pass via microinstruction RESFA (/10) to the FXZEX (/0F) branch of RESULT. The sequensor is resynchronized during the cycles UPDA by signals BOFFN and BSYCPUAN/BN to the logic for T3D.

D.19 FLD Command

The FLD command is complete when the two-word mantissa and the exponent have been loaded into the FPP from memory. The mantissa is loaded in microinstructions WAIT (/00) and RDM2 (/01). The exponent is loaded in microinstruction RDEXP (/04). The flow then branches via OPER to microinstruction REQM (/14). Between REQM (/14) and the following RESALU (/03), the loaded data are cycled through the ALU/EALU to the floating-point accumulator (FA/EF). If the mantissa received from memory equals Zero, the OPER branch is via RESNU (/12) to load both the mantissa and exponent registers with zero.

D.20 FST Command

The FST command is controlled by the CPU; no FPP microinstructions are used and the FST is not shown on the flow diagram. CPU Bus-control timing signals BSYCPUAN and BOFF gate the FPP accumulator out onto the BIO lines (logic b). The BSYCPUAN signal also operates the Bus Selection Counter (logic d) to switch the three 16-bit parts of the floating-point data onto the BIO lines.

D.21 FAD, FSU Commands

The double-word mantissa and the exponent are loaded into M/EM by microinstructions WAIT (/00), RDM2 (/01), and RDEXP (/04). This operand is added to, or subtracted from, the operand in FA/EF. The difference between the add and subtract operations is in the logic control of the ALU during microinstruction ADSUM (/1B) or ADSUA3 (/0E). The branch at OPER depends on the conditions of the mantissa (Table D-1):

- RESNU (/12) if both mantissa are zero.
- RESFA (/10) if only the new mantissa (M) is zero. The result is thus already in the accumulator (FA).
- REQPMM (/13) if the mantissa in FA only is zero. The received mantissa is added to zero by microinstructions REQPMM (/13) and ADSUM (/1B).
- TSTEXP (/11) to test and align the exponents if both mantissa are other than zero.

D.22 FMU Command

A double-word mantissa and an exponent are loaded into M/EM by microinstructions WAIT (/00), RDM2 (/01), and RDEXP (/04). The flow then branches via OPER to microinstruction MUL1 (/18). The mantissa of the first operand, already loaded in Q, is shifted left one bit (MUL1, /18) and then right one bit (MUL2, /0C); this operation removes the unused bit-0 of the right word. The actual multiplication is performed during repeated sequensor cycles during microinstruction ALGOMU (0D).

D.23 FDV Command

The divisor operand is loaded into M/EM by microinstructions WAIT (/00), RDM2 (/01), and RDEXP (/04). The flow then branches via OPER to microinstruction DIV1 (/15). The dividend in FA is shifted right (SRFA in DIV2, /17) to ensure that the dividend is smaller than the divisor. The division is performed during repeated sequensor cycles during the microinstruction ALGODI (/0A).

D.24 LOGIC AND TIMING

The timings for the different FPP operations are provided together on Figure D-5. The control codes for the different logic operations are given on Figure D-6. Detailed logic diagrams are provided by Figure D-7 at the end of this section; specific sheets (a to d) of the logic are referenced by the letter suffix only, in the following manner: Figure D-7, sheet b is referenced as "logic b".

D.25 Microprogram Control (logic c)

The FPP operations are controlled by a selection of microinstruction control words stored in a read only memory (Control ROM). At time AP of a microinstruction cycle, a ROM address is selected and the ROM contents of that address are available to control the FPP logic for that cycle at time BP. Each microinstruction is shown as a separate block on the flow diagram, Figure D-4, and the hexadecimal ROM address is given in each block. The complete contents of the microprogram control ROM are shown in the ROM listing, Table D-3.

D.26 The microcommand bits from the ROM are grouped into functional fields that provide control for the different logic sections. The fields are listed and explained at the beginning of Table D-3.

D.27 Microinstruction Addressing is provided by the Next Address Selection field of each microinstruction word; this field sets up the address for the following microinstruction. The three control bits from the ROM (μSNA0-2) control the SNA multiplexer (logic c) for selecting an explicit address or one of the addresses selected by the operation Branch Conditions (Table D-1). The address-control conditions selected via SNA, and ROM bits μSNA0-2, select an address code from the ROM Address Logic (RAD). At the completion of the microinstruction, the rising-edge of AP which starts the next microinstruction clocks the new address into register RA, and the contents of another ROM word are present at the ROM output. The Next-Address control code is shown in Figure D-6.

```
*       DEFINITION OF THE MICROINSTRUCTION FORMAT

*       FIELD LENGTH DEFINITION
R       FORM    3,1 = 0,3,1,4,2,2,3,3,2,2,2,3,1 = 0
*       FIELD   0         1 2 3 4 5 6 7 8 9 10 11
*       FIELD   0       NEXT ADDRESS SELECTION

EXPO    EQU     0       EXPLICIT WITH ADDRESS < /10
IDLE    EQU     1       IDLE TEST
TEFL    EQU     2       READ EXPONENT OR FFL TEST
REST    EQU     3       RESULT TEST
EXPI    EQU     4       EXPLICIT WITH ADDRESS > /0F
IFIX    EQU     5       FIX TEST
ALGN    EQU     6       ALIGNMENT TEST
OPER    EQU     7       OPERATION TEST

*       FIELD   1       SEQUENSOR

S180    EQU     0       SINGLE BP; AP IN 180 NS
M090    EQU     1       MULTIPLE BP EVERY 90 NS
S135    EQU     2       SINGLE BP; AP IN 135 NS
M135    EQU     5       MULTIPLE BP EVERY 135 NS

*       FIELD   2       END

S       EQU     1       END OF PROCESS

*       FIELD   3       MANTISSA ARITHMETIC UNIT SELECTION

ALFA    EQU     0       ALUM = FA
ALUZ    EQU     1       ALUM = ZERO
ALFM    EQU     2       ALUM = M
MULT    EQU     4       ALUM = FA (+OR-) M ACCORDING MULTI ALGORITHM
ADSU    EQU     8       ALUM = FA (+OR-) M ACCORDING K07
SUAD    EQU     10      ALUM = M (+OR-) FA ACCORDING K07
DIVI    EQU     12      ALUM = FA (+OR-) M ACCORDING DIVI ALGORITHM

*       FIELD   4       MANTISSA ACCUMULATOR CONTROL

NFA     EQU     0       FA UNCHANGED
RFA     EQU     1       FA = ALUM RIGHT SHIFTED
LFA     EQU     2       FA = ALUM LEFT SHIFTED
AFA     EQU     3       FA = ALUM

*       FIELD   5       Q REGISTER CONTROL

NCQ     EQU     0       Q UNCHANGED
SLQ     EQU     1       Q SHIFTED LEFT
SRQ     EQU     2       Q SHIFTED RIGHT
LDQ     EQU     3       Q = FAR

*       FIELD   6       M REGISTER CONTROL

MNC     EQU     0       M UNCHANGED
MYQ     EQU     3       M = 0
MSB     EQU     4       M SELECT BIO BUT NO CLOCK (LOADING EXPON.)
MBR     EQU     5       M RIGHT = BIO (LOADING 2ND M.)
MBL     EQU     6       M LEFT = BIO (LOADING 1ST M.)

*       FIELD   7       EXPONENT ARITHMETIC UNIT SELECTION

EAMB    EQU     0       EALU = EFB-EM
EACB    EQU     1       EALU = EFB-EM-1
EQEM    EQU     2       EALU = EM
EALZ    EQU     4       EALU = ZERO
EQFB    EQU     6       EALU = EFB
EAPB    EQU     7       EALU = EFB+EM

*       FIELD   8       EM REGISTER CONTROL

EM30    EQU     1       EM = 30
EMNC    EQU     2       EM UNCHANGED
EMLB    EQU     3       EM = BIO

*       FIELD   9       EF REGISTER CONTROL

EFM1    EQU     0       EF = EF-1
EFP1    EQU     1       EF = EF+1
EFNC    EQU     2       EF UNCHANGED
EFLD    EQU     3       EF = EALU

*       FIELD   10      CT COUNTER CONTROL

CTM1    EQU     0       CT = CT-1
CTP1    EQU     1       CT = CT+1
CTNC    EQU     2       CT UNCHANGED
CTLD    EQU     3       CT = EALU

*       FIELD   11      MISCELLANEOUS COMMAND

GFUNM   EQU     1       SET UNNORMALISED FLAG
GFXDV   EQU     2       SET DVZO OR OVF FIX
GFEVF   EQU     3       SET OVERFLOW ON EXPONENT ARITHMETIC
GFEVP   EQU     4       SET OVERFLOW ON EXPONENT COUNT UP
GFEVN   EQU     5       SET UNDERFLOW ON EXPONENT COUNT DOWN
GFFIX   EQU     6       VALID. OF NEG. FFX CORRECTION
A       EQU     /A
B       EQU     /B
C       EQU     /C
D       EQU     /D
E       EQU     /E
F       EQU     /F
```

Table D-3 Microprogram (ROM) Listing (B)

	NAME		SNA	SEQ	E	ALU	FA	Q	M	EALU	EM	EF	CT	MISC.
00	WAIT	R	IDLE,	S135,	0,	ALFA,	AFA,	NCQ,	MBI,	EACB,	EM30,	EFNC,	CTNC,	0
01	RDM2	R	TEFL,	S135,	0,	ALEM,	NFA,	NCQ,	MBR,	EQEM,	EMNC,	EFNC,	CTNC,	0
02	FFX	R	IFIX,	S135,	0,	ALFA,	NFA,	NCQ,	MNC,	EACB,	EMNC,	EFNC,	CTLD,	0
03	RESALU	R	REST,	S180,	0,	ALFA,	AFA,	NCQ,	MNC,	EALZ,	EMNC,	EFLD,	CTNC,	0
04	RDEXP	R	OPER,	S180,	0,	ALEM,	NFA,	NCQ,	MSB,	EACB,	EMLB,	EFNC,	CTLD,	0
05	RESULT	R	EXP1,	S180,	S,	ALFA,	NFA,	NCQ,	MNC,	EQEM,	EMNC,	EFNC,	CTNC,	0
06	OVER	R	EXP0,	S180,	0,	ALFA,	RFA,	NCQ,	MNC,	EQEM,	EMNC,	EFP1,	CTNC,	GFEVP
07	UNORM	R	EXP0,	M090,	0,	ALFA,	LFA,	NCQ,	MNC,	EQEM,	EMNC,	EFM1,	CTNC,	GFEVN
08	ADSUA1	R	EXP0,	S135,	0,	ALFA,	AFA,	NCQ,	MYQ,	EQEM,	EMNC,	EFNC,	CTNC,	0
09	ADSUA2	R	EXP0,	M090,	0,	ALFA,	RFA,	NCQ,	MNC,	EQEM,	EMNC,	EFNC,	CTM1,	0
0A	ALGODI	R	EXP0,	M135,	0,	DIVI,	LFA,	SLQ,	MNC,	EAMB,	EMNC,	EFNC,	CTM1,	GFEVF
0B	SUEX	R	EXP0,	S135,	0,	ALEM,	NFA,	NCQ,	MYQ,	EAMB,	EMNC,	EFNC,	CTNC,	0
0C	MUL2	R	EXP0,	S135,	0,	MULT,	AFA,	SRQ,	MNC,	EAPB,	EMNC,	EFNC,	CTNC,	0
0D	ALGOMU	R	EXP0,	M135,	0,	MULT,	RFA,	SRQ,	MNC,	EAPB,	EMNC,	EFNC,	CTM1,	GFEVF
0E	ADSUA3	R	EXP0,	S135,	0,	SUAD,	NFA,	NCQ,	MNC,	EQFB,	EMNC,	EFNC,	CTNC,	0
0F	FXZEX	R	EXP0,	S135,	0,	ALFA,	AFA,	NCQ,	MNC,	EQEM,	EMNC,	EFLD,	CTNC,	0

AD	NAME		SNA	SEQ	E	ALU	FA	Q	M	EALU	EM	EF	CT	MISC.
10	RESFA	R	EXP0	S135	0	ALFA	NFA	NCQ	MNC	EQFB	EMNC	EFNC	CTNC	0
11	TSTEXP	R	ALGN	S180	0	ALFA	NFA	NCQ	MNC	EACB	EMNC	EFNC	CTLD	0
12	RESNU	R	EXP0	S135	0	ALUZ	NFA	NCQ	MNC	EALZ	EMNC	EFNC	CTNC	0
13	REQPMM	R	EXP1	S135	0	ALUZ	NFA	NCQ	MNC	EQEM	EMNC	EFNC	CTNC	0
14	REQM	R	EXP0	S135	0	ALEM	NFA	NCQ	MNC	EQEM	EMNC	EFNC	CTNC	0
15	DIV1	R	EXP1	S135	0	ALFA	NFA	NCQ	MNC	EQEM	EMNC	EFNC	CTP1	0
16	UNMOP	R	EXP1	S180	0	ALFA	NFA	NCQ	MNC	EQEM	EMNC	EFNC	CTNC	GFUNM
17	DIV2	R	EXP0	S135	0	DIVI	RFA	NCQ	MNC	EAMB	EMNC	EFNC	CTNC	0
18	MUL1	R	EXP0	S135	0	ALUZ	NFA	SLQ	MNC	EAPB	EMNC	EFNC	CTM1	0
19	DENORM	R	EXP1	M090	0	ALFA	RFA	SRQ	MNC	EQEM	EMNC	EFNC	CTP1	GFFIX
1A	OVXDV	R	EXP1	S180	0	ALFA	NFA	NCQ	MNC	EQEM	EMNC	EFNC	CTNC	GFXDV
1B	ADSUM	R	EXP0	S135	0	ADSU	AFA	NCQ	MNC	EQEM	EMNC	EFNC	CTNC	0
1C	ALGM	R	EXP0	S135	0	ALEM	NFA	NCQ	MNC	EQEM	EMNC	EFNC	CTNC	0
1D	ALGFA	R	EXP1	M090	0	ALFA	RFA	NCQ	MNC	EQEM	EMNC	EFNC	CTP1	0
1E	UPDA1	R	EXP1	S135	0	ALFA	NFA	LDQ	MSB	EQFB	EMNC	EFNC	CTNC	0
1F	UPDA2	R	EXP0	S135	0	ALEM	NFA	NCQ	MYQ	EQEM	EMNC	EFLD	CTNC	0

Table D-3 Microprogram (ROM) Listing (C)

ADDRESS	FPCS0				FPCS1			FPCS2			FPCS3			
	0 - SNA		1 - SEQ	2 - END	3 - ALU	4 - FA	5 - Q	6 - M	7 - EALU	8 - EM	9 - EF	10 - CT	11 - GF	
0	0 0 1	0	0 1 0	0	0 0 0 0	1 1	0 0	1 1 0	0 0 1	0 1	1 0	1 0	0 0 0	0
1	0 1 0	0	0 1 0	0	0 0 1 0	0 0	0 0	1 0 1	0 1 0	1 0	1 0	1 0	0 0 0	0
2	1 0 1	0	0 1 0	0	0 0 0 0	0 0	0 0	0 0 0	0 0 1	1 0	1 0	1 1	0 0 0	0
3	0 1 1	0	0 0 0	0	0 0 0 0	1 1	0 0	0 0 0	1 0 0	1 0	1 1	1 0	0 0 0	0
4	1 1 1	0	0 0 0	0	0 0 1 0	0 0	0 0	1 0 0	0 0 1	1 1	1 0	1 1	0 0 0	0
5	1 0 0	0	0 0 0	1	0 0 0 0	0 0	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 0 0	0
6	0 0 0	0	0 0 0	0	0 0 0 0	0 1	0 0	0 0 0	0 1 0	1 0	0 1	1 0	1 0 0	0
7	0 0 0	0	0 0 1	0	0 0 0 0	1 0	0 0	0 0 0	0 1 0	1 0	0 0	1 0	1 0 1	0
8	0 0 0	0	0 1 0	0	0 0 0 0	1 1	0 0	0 1 1	0 1 0	1 0	1 0	1 0	0 0 0	0
9	0 0 0	0	0 0 1	0	0 0 0 0	0 1	0 0	0 0 0	0 1 0	1 0	1 0	0 0	0 0 0	0
0A	0 0 0	0	1 0 1	0	1 1 0 0	1 0	0 1	0 0 0	0 0 0	1 0	1 0	0 0	0 1 1	0
0B	0 0 0	0	0 1 0	0	0 0 1 0	0 0	0 0	0 1 1	0 0 0	1 0	1 0	1 0	0 0 0	0
0C	0 0 0	0	0 1 0	0	0 1 0 0	1 1	1 0	0 0 0	1 1 1	1 0	1 0	1 0	0 0 0	0
0D	0 0 0	0	1 0 1	0	0 1 0 0	0 1	1 0	0 0 0	1 1 1	1 0	1 0	0 0	0 1 1	0
0E	0 0 0	0	0 1 0	0	1 0 1 0	0 0	0 0	0 0 0	1 1 0	1 0	1 0	1 0	0 0 0	0
0F	0 0 0	0	0 1 0	0	0 0 0 0	1 1	0 0	0 0 0	0 1 0	1 0	1 1	1 0	0 0 0	0
10	0 0 0	0	0 1 0	0	0 0 0 0	0 0	0 0	0 0 0	1 1 0	1 0	1 0	1 0	0 0 0	0
11	1 1 0	0	0 0 0	0	0 0 0 0	0 0	0 0	0 0 0	0 0 1	1 0	1 0	1 1	0 0 0	0
12	0 0 0	0	0 1 0	0	0 0 0 1	0 0	0 0	0 0 0	1 0 0	1 0	1 0	1 0	0 0 0	0
13	1 0 0	0	0 1 0	0	0 0 0 1	0 0	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 0 0	0
14	0 0 0	0	0 1 0	0	0 0 1 0	0 0	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 0 0	0
15	1 0 0	0	0 1 0	0	0 0 0 0	0 0	0 0	0 0 0	0 1 0	1 0	1 0	0 1	0 0 0	0
16	1 0 0	0	0 0 0	0	0 0 0 0	0 0	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 0 1	0
17	0 0 0	0	0 1 0	0	1 1 0 0	0 1	0 0	0 0 0	0 0 0	1 0	1 0	1 0	0 0 0	0
18	0 0 0	0	0 1 0	0	0 0 0 1	0 0	0 1	0 0 0	1 1 1	1 0	1 0	0 0	0 0 0	0
19	1 0 0	0	0 0 1	0	0 0 0 0	0 1	1 0	0 0 0	0 1 0	1 0	1 0	0 1	1 1 0	0
1A	1 0 0	0	0 0 0	0	0 0 0 0	0 0	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 1 0	0
1B	0 0 0	0	0 1 0	0	1 0 0 0	1 1	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 0 0	0
1C	0 0 0	0	0 1 0	0	0 0 1 0	0 0	0 0	0 0 0	0 1 0	1 0	1 0	1 0	0 0 0	0
1D	1 0 0	0	0 0 1	0	0 0 0 0	0 1	0 0	0 0 0	0 1 0	1 0	1 0	0 1	0 0 0	0
1E	1 0 0	0	0 1 0	0	0 0 0 0	0 0	1 1	1 0 0	1 1 0	1 0	1 0	1 0	0 0 0	0
1F	0 0 0	0	0 1 0	0	0 0 1 0	0 0	0 0	0 1 1	0 1 0	1 0	1 1	1 0	0 0 0	0

D.28 Sequensor (logic d)
The sequensor logic is driven by the OSCFLO timing signal from the CPU and the control commands. OSCFLO is derived directly from the CPU sequensor and has the same 22.5 MHz frequency, with a duration of 45 nsec. One FPP sequensor cycle is produced for each microinstruction. Each cycle begins with an AP pulse and ends when the next AP pulse starts the next cycle. At each AP time, a new microprogram address is loaded into register RA (logic c) to select the next microprogram. At time BP, the conditions specified by the microinstruction are executed, such as data transfer from one register to another, arithmetic operation, etc. The timing signals T1, T2, T3, T4 are used within the sequensor logic only.

D.29 The sequensor is controlled by the microprogram to run in one of four modes: S135, S180, M090, M135. In either of the two single-cycle modes (S135, S180), the sequensor produces a cycle of a fixed length (either 135 or 180 nsec) and with a single BP pulse. In either of the two multiple-cycle modes (M090, M135), the sequensor produces a variable-length cycle where the BP pulse is repeated (every 90 or 135 nsec) until the repeat condition (REPCOND) is disabled. The Sequensor Cycles and the Repeat Condition code are shown on Figure D-6.

D.30 Instruction Loading (logic d)
An FPP instruction is loaded into the K-register under CPU Bus control during a CPU Fetch microprogram, while the FPP idles in the WAIT microinstruction (/00). The FPP instruction code (Figure D-6) is loaded into the K-register on the trailing edge of BSYCPUAN/BN. The load-instruction timing is shown in Figure D-5.

D.31 Operand Loading
Operands are loaded into the FPP registers M and EM during instruction FFL (to M only), FLD, and the four arithmetic instructions. The operand source is the CPU for the FFL instruction and the memory for the other instructions (Figure D-2). Part or all of the three-word operand (double-word mantissa and one-word exponent) is loaded during microinstructions WAIT (/00), RDM2 (/01), and RDEXP (/04), according to the specific instruction.

D.32 Operand loading is performed under CPU Bus control. The Bus timing is shown on Figure D-4. The CPU control signals BSYCPUAN and TMFN together enable the sequensor T3 count (logic d). Gating the operand word from the BIO lines into the M or EM register is controlled by the microprogram control bits (Figure D-6); in both cases, the data are gated on the leading edge of BP.

D.33 Store Operand
A Store-Operand operation is performed under CPU Bus control to transfer an integer operand to the CPU (FFX instruction) or to transfer a floating-point operand to the memory (FST instruction). The timing is shown on Figure D-5. The CPU control signals BOFFN and BSYCPUAN/BN (logic d) are used to enable the sequencing of Bus-selection bits BIOS0/S1. The same two control signals gate the selected data onto the BIO lines (logic b).

D.34 FPP Operation Control
The FPP operation processing is controlled by CPU signal FLOACT (Figure D-5; logic d) which is derived from the CPU microprogram bit GFLOT. FLOACT is received at the beginning of an FFX instruction. FLOACT controls the IDLE branch of the microprogram control to start the FFX operation. For the FLD and the four arithmetic instructions, FLOACT is received near the end of the CPU instruction to time the actual processing and to synchronise the end of the operation. FLOACT is a condition for loading the CR in the CPU.

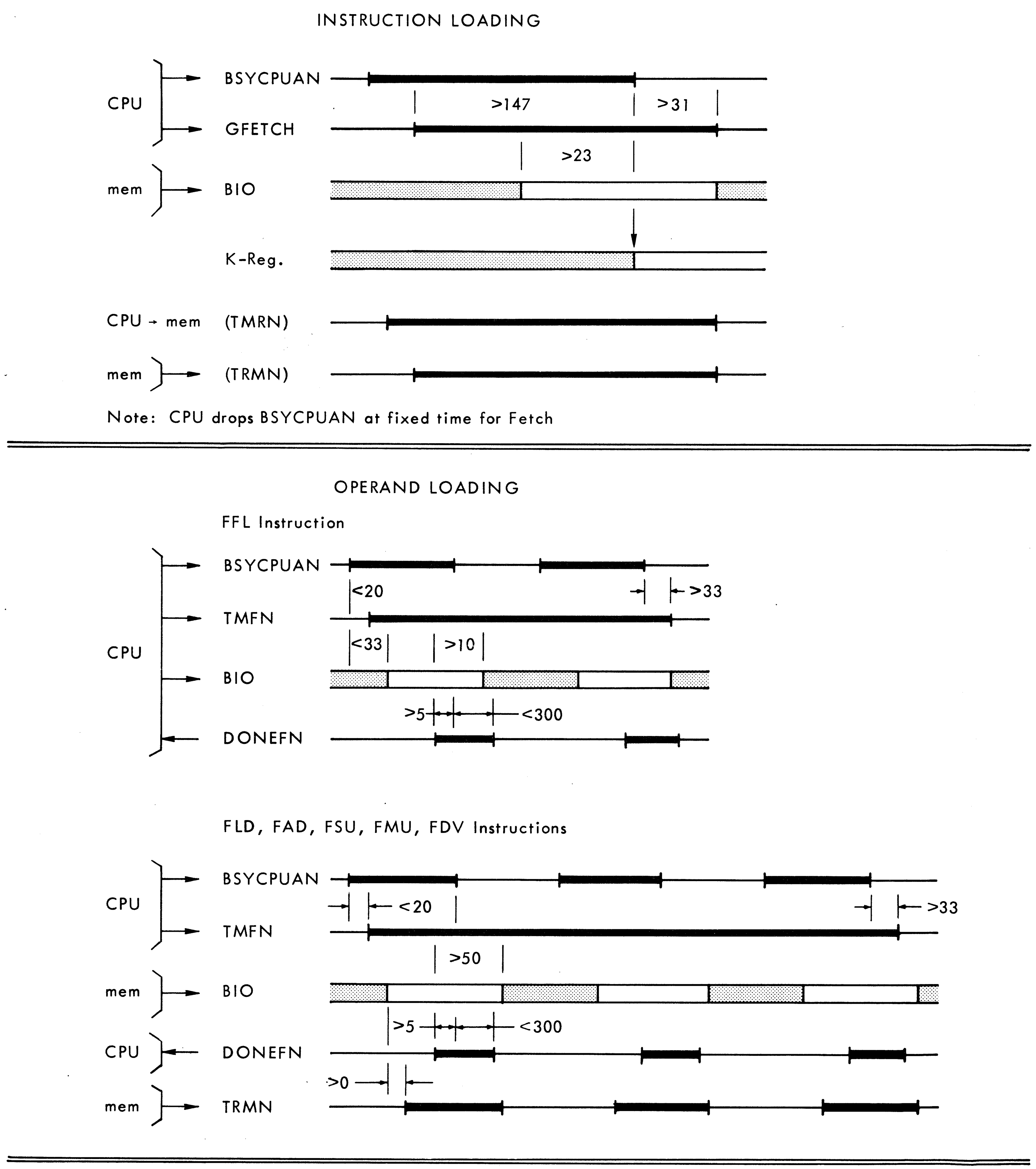


Figure D-5 FPP Timing (A)

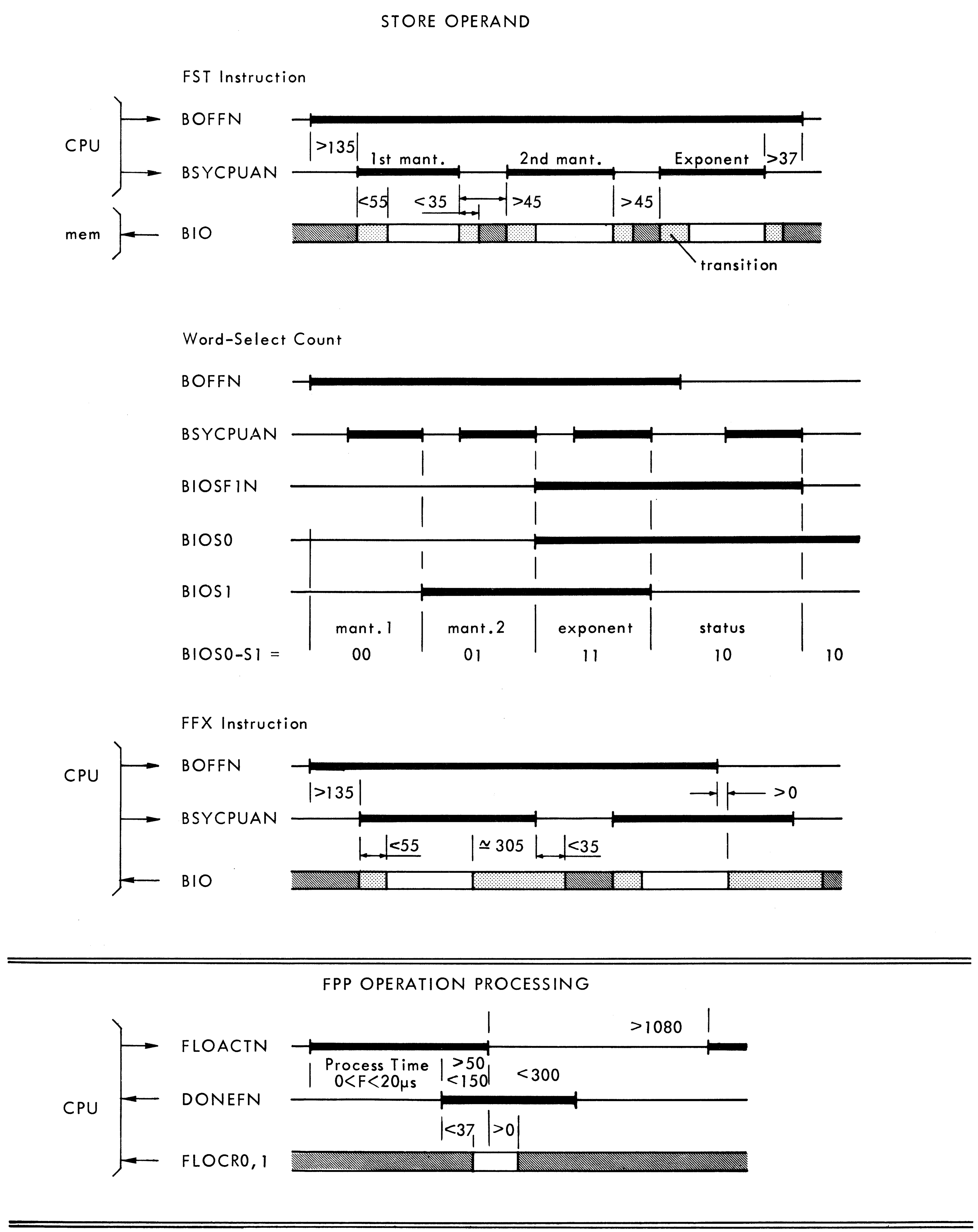
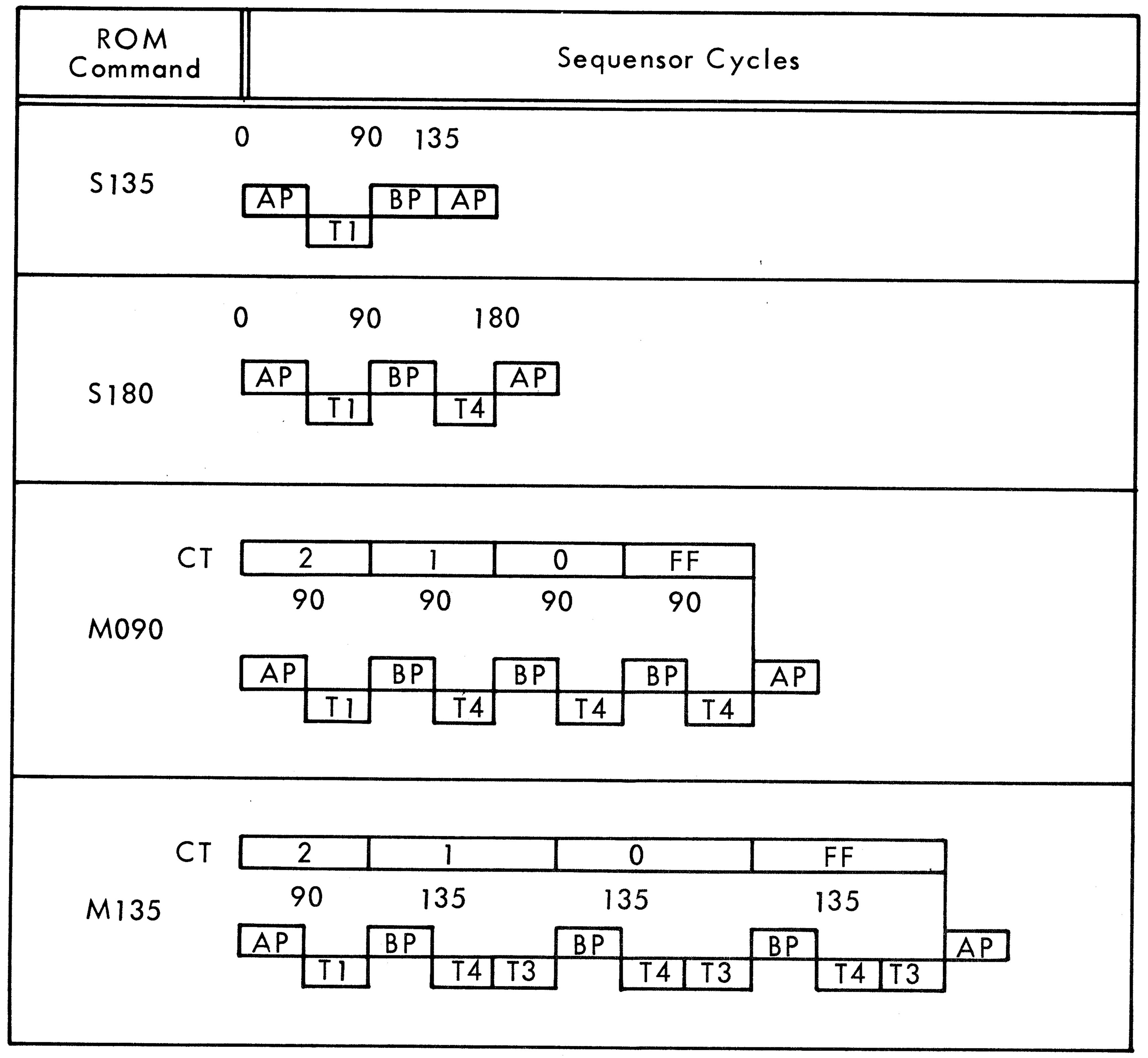


Figure D-5 FPP Timing (B)

Figure D-5 FPP Timing (A)

Figure D-5 FPP Timing (B)



S = Single Cycle
M = Multiple Cycle

REPEAT CONDITION

MUEFON	ALBUF2	ALBUF3	REPCOND
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	$\overline{\text{REPTCN}}$
1	0	1	
1	1	0	
1	1	1	

$\overline{\text{REPTCN}}$ = inactive
Repeat-Terminated Count
(Not CT0-7 = 1 = /FF)

Figure D-6 Logic Control Codes (A)

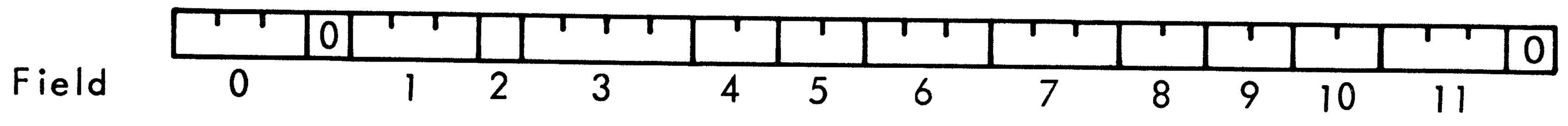
INSTRUCTION CODES

Inst.	BIO Lines, and K-Reg. Bits								K DECODE						KDVFLD	FLFX
									SST (K05)			INST				
	4	5	6	7	8	9	10	15	0	1	2	0	1	2		
FFL	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1
FFX	1	0	0	1	0	0	0	1	0	1	1	0	1	1	0	1
FLD	0	0	0	1	0	-	-	0	0	0	0	0	0	0	1	0
FST	0	0	0	1	0	-	-	1	-	-	-	-	-	-	-	-
FAD	1	1	0	0	0	-	-	-	1	0	0	1	0	0	0	0
FSU	1	1	0	1	0	-	-	-	1	0	1	1	0	1	0	0
FMU	1	1	1	0	0	-	-	-	1	1	0	1	1	0	0	0
FDV	1	1	1	1	0	-	-	-	1	1	1	1	1	1	1	0

SST	BIO Lines											
	4	5	6	7	8	9	10	11	12	13	14	15
							Device Address					
	1	-	-	-	1	1	0	0	0	0	0	0
								KOR2D				

Figure D-6 Logic Control Codes (B)

Microinstruction Word Format



Field 0 — Next Address Selection

		µSNA0	µSNA1	µSNA2	(see Table D-1)
0	EXP0	0	0	0	Explicit, address </10
1	IDLE	0	0	1	Idle test
2	TEFL	0	1	0	Read exponent or FFL test
3	REST	0	1	1	Result test
4	EXP1	1	0	0	Explicit, address >/0F
5	IFIX	1	0	1	Fix test
6	ALIGN	1	1	0	Alignment test
7	OPER	1	1	1	Operation test

Field 1 — Sequensor

		µT3	µT4N	µREP	
0	S180	0	0	0	Single BP, AP in 180ns
1	M090	0	0	1	Multiple BP every 90ns
2	S135	0	1	0	Single BP, AP in 135ns
5	M135	1	0	1	Multiple BP every 135ns

Field 2 — End

		µEND	Microinstruction /05 only
1	S	1	End of Process

Field 3 — Mantissa ALU Selection

		µALU0	µALU1	µALU2	µALUCE	
0	ALFA	0	0	0	0	(FA) → ALUM
1	ALUZ	0	0	0	1	Zero → ALUM
2	ALEM	0	0	1	0	(M) → ALUM
4	MULT	0	1	0	0	(FA) + or - (M) → ALUM*MULTI Algorithm
8	ADSU	1	0	0	0	(FA) + or - (M) → ALUM*K07
10	SUAD	1	0	1	0	(M) + or - (FA) → ALUM*K07
12	DIVI	1	1	0	0	(FA) + or - (M) → ALUM*DIVI Algorithm

* = according to

Figure D-6 Logic Control Codes (C)

Field 4 — Mantissa Accumulator Control

		µFAS1N	µFAS0N	
0	NFA	0	0	FA unchanged
1	RFA	0	1	(ALUM) Right-shifted → FA
2	LFA	1	0	(ALUM) Left-shifted → FA
3	AFA	1	1	(ALUM) → FA

Field 5 — Q-Register Control

		µQS0	µQS1	
0	NCQ	0	0	Q unchanged
1	SLQ	0	1	Shift-Left Q
2	SRQ	1	0	Shift-Right Q
3	LDQ	1	1	Load (FAB) → Q

Field 6 — M-Register Control

		µMSELN	µLDML	µLDMR		
0	MNC	0	0	0	M unchanged	
3	MYQ	0	1	1	(Q) → M	
4	MSB	1	0	0	BIO selected, but no clock	(loading exponent)
5	MBR	1	0	1	BIO → M Right	(loading 2nd. mant.)
6	MBL	1	1	0	BIO → M Left	(loading 1st. mant.)

Field 7 — Exponent ALU Selection

		µEALU0	µEALU1	µEALU2	
0	EAMB	0	0	0	(EFB) - (EM) → EALU
1	EACB	0	0	1	(EFB) - (EM) - 1 → EALU
2	EQEM	0	1	0	(EM) → EALU
4	EALZ	1	0	0	Zero → EALU
6	EQFB	1	1	0	(EFB) → EALU
7	EAPB	1	1	1	(EFB) + (EM) → EALU

Figure D-6 Logic Control Codes (D)

Figure D-6 Logic Control Codes (C)
Figure D-6 Logic Control Codes (D)

Field 8 — EM-Register Control

		µEMS	µEMLD	
1	EM30	0	1	30 → EM
2	EMNC	1	0	EM unchanged
3	EMLB	1	1	BIO → EM

Field 9 — EF-Register Control

		µEF0N	µEF1	(EFLOADN)	
0	EFM1	L	L	H	(EF) -1 → EF [decrement]
1	EFP1	L	H	H	(EF) +1 → EF [increment]
2	EFNC	H	L	H	EF unchanged
3	EFLD	H	H	L	(EALU) → EF [load]

Field 10 — CT-Counter Control

		µCT0N	µCT1	
0	CTM1	0	0	(CT) -1 → CT [decrement]
1	CTP1	0	1	(CT) +1 → CT [increment]
2	CTNC	1	0	CT unchanged
3	CTLD	1	1	(EALU) → CT [load]

Field 11 — Miscellaneous Commands

		µGP0	µGP1	µGP2	
1	GFUNM	0	0	1	Set unnormalized flag
2	GFXDV	0	1	0	Set DVZO or OVF FIX
3	GFEVF	0	1	1	Set overflow on exponent arithmetic
4	GFEVP	1	0	0	Set overflow on exponent count-up
5	GFEVN	1	0	1	Set underflow on exponent count-down
6	GFFIX	1	1	0	Validation of negative FFX correction

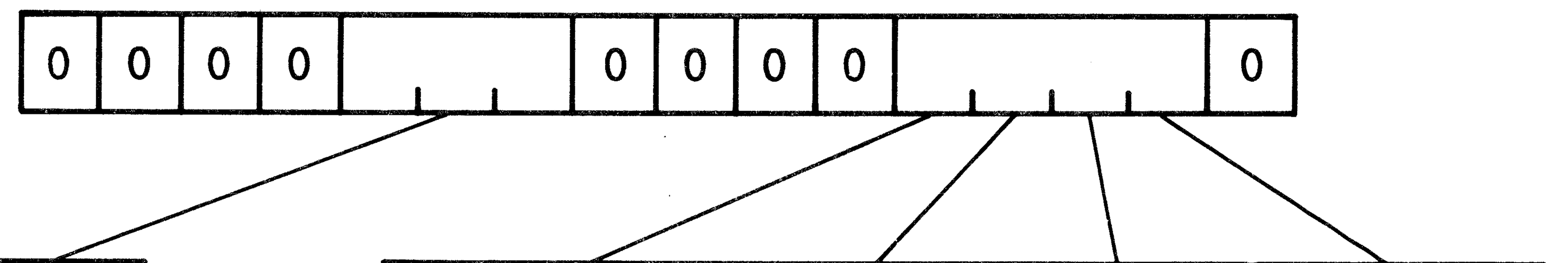
Figure D-6 Logic Control Codes (E)

D.35 STATUS AND INTERRUPTS

The floating-point instruction executions are not interruptable; if a Memory Management Unit detects a Page Fault, the floating-point instruction is stopped like any other instruction. An abnormal condition detected during a floating-point instruction execution sets the corresponding flag in the status register, sets the condition register to 3, and generates a Floating-Point interrupt. The Floating-Point interrupt can be connected only to one of the eight internal interrupts (refer to CPU Section I, Interrupt System).

D.36 At the moment any error condition is detected, the activating status-register bit loads the instruction code from the D-register into the INST register. The INST code is then included with the status word to specify which instruction type caused the error. The system should respond to the FPP interrupt by sending an SST instruction with device address equal to zero. The floating-point status is then transferred to the CPU, as follows:

STATUS



INST 0 1 2		Unnormalized*	Division by Zero	Overflow	Underflow
0 0 0	FLD	Yes			
0 0 1	not used				
0 1 0	FFL				
0 1 1	FFX			exponent > 30	
1 0 0	FAD	Yes		result exponent ≥ 2^{15}	result exponent < -2^{15}
1 0 1	FSU				
1 1 0	FMU				
1 1 1	FDV		divisor is zero		

* If Unnormalized is set, the operation is aborted, no other flag is set, and the FPP accumulator remains unchanged.

D.37 The interrupt and the error bit (right byte) of the status word are reset at the end of an SST instruction. If another floating-point instruction is performed before the SST instruction is received, the error bits of the status word indicate the accumulated status (logical Or) of both instructions. The INST code in the left byte, however, retains the code for the first instruction where error status was set.

D.38 SIGNAL LIST

A complete list of all FPP input and output signals is given in Table D-4.

D.39 CARD LAYOUT

The layout of the FPP card is provided in Figure D-8. There are no U-links or other operator controls located on the FPP card.

D.40 PARTS LIST

A list of FPP components is provided in Table D-5.

Table D-4 FPP Signal List

Signal	Conn.	Logic Sheet	Remarks
INPUT TO FPP			
BIO00-15N	3---	D	
BOFFN	5B13	D	BIO contents must be defined by the FPP
BSYCPUAN	5A12	D	CPU is master of the Bus
CLEARN	3A39	D	
FLOACT	5A11	D	CPU activation signal for FFX and CR
GFETCH	5A13	D	CPU fetch cycle
MFAULTN	5B20	D	MMU detects a Page Fault
OSCFLO	5A17	D	CPU clock signal
TMFN	5B12	D	CPU microcommand bit
TMPN	3A31	D	Bus timing from CPU for SST dialogue
TRMN	3A28	D	Bus timing from memory
OUTPUT FROM FPP			
ACN	3A34	D	Accept for SST command
BIO00-15N	3---	B	
DONEFN	5A14	C	FPP operation was done
FLOCR0	5B14	C	FPP condition register bits to CPU
FLOCR1	5A15	C	
FPPABS	5B15	C	Held inactive when FPP board is inserted
INTFPPN	3A16	D	FPP interrupt
TPMN	3A32	D	Bus timing to CPU for SST command

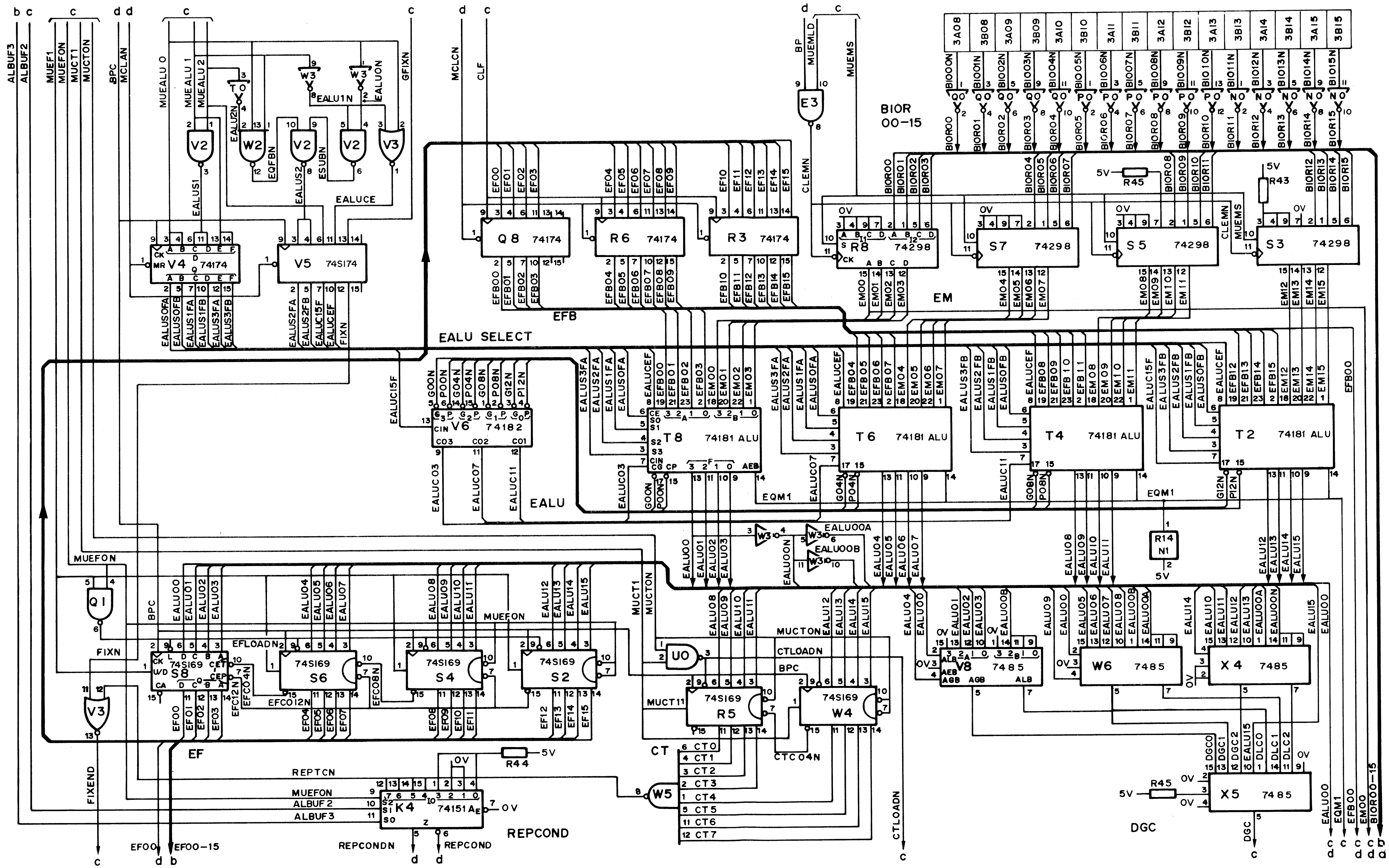


Figure D-7a FPP Logic Diagram

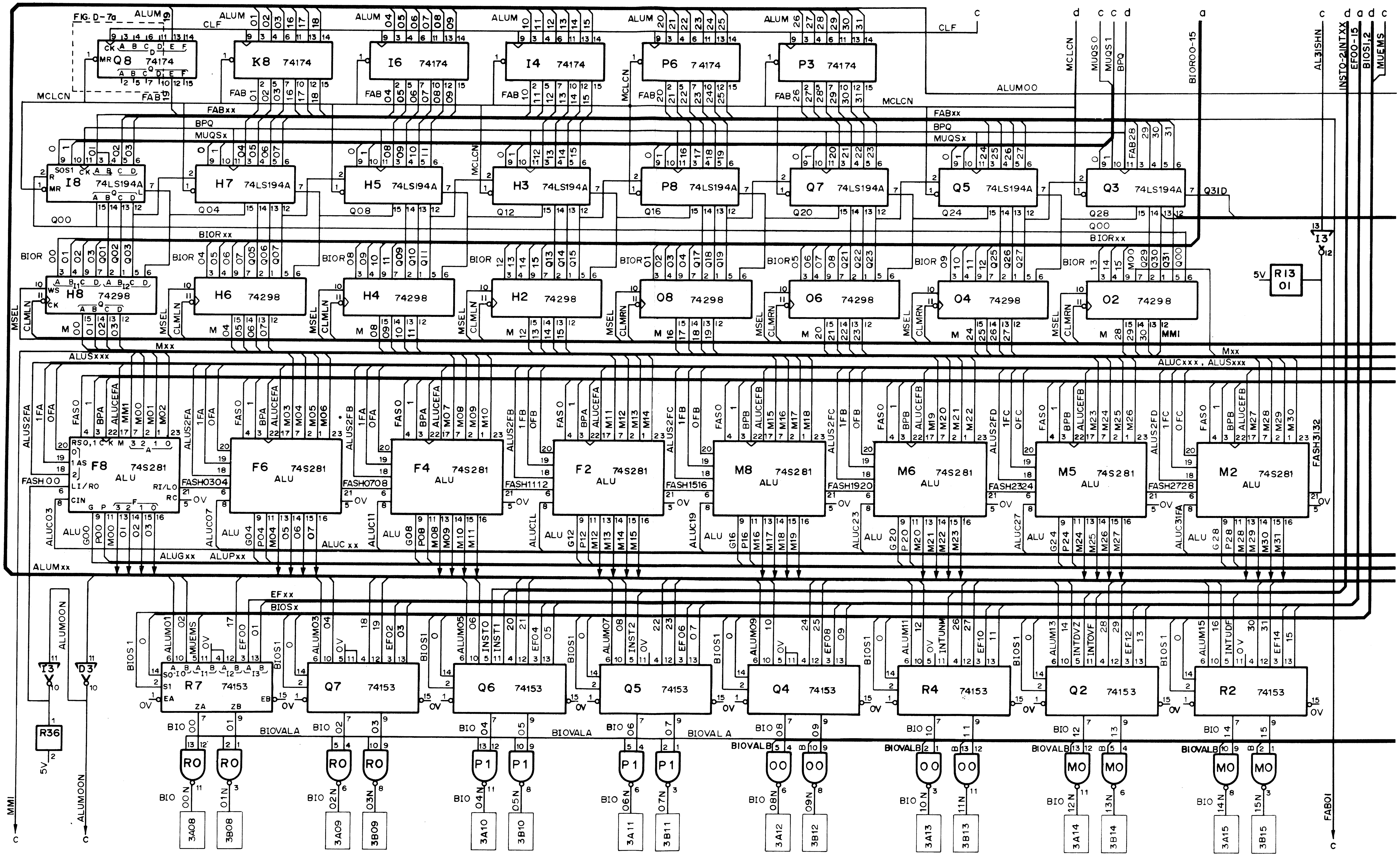


Figure D-7b FPP Logic Diagram

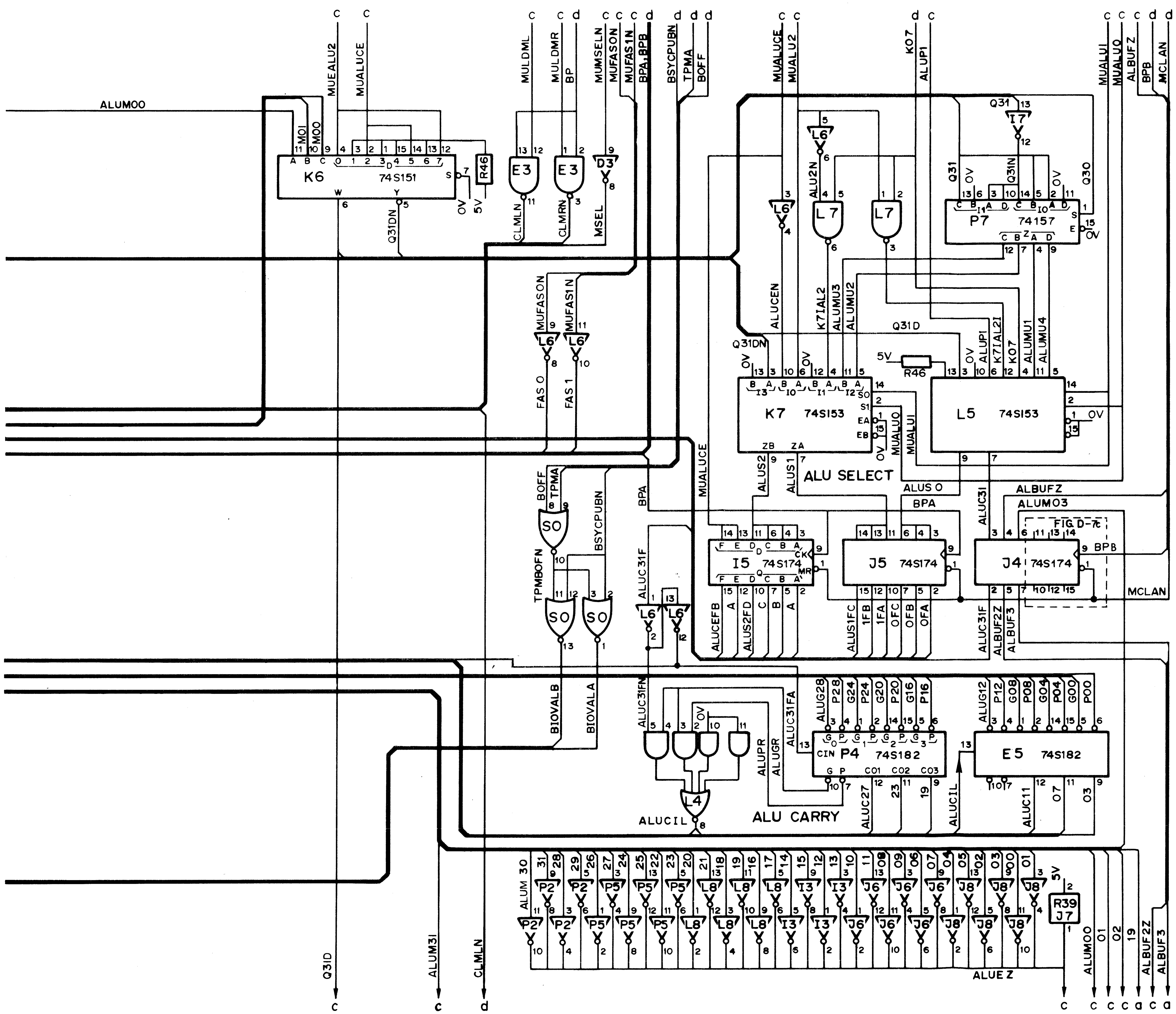


Figure D-7b FPP Logic Diagram

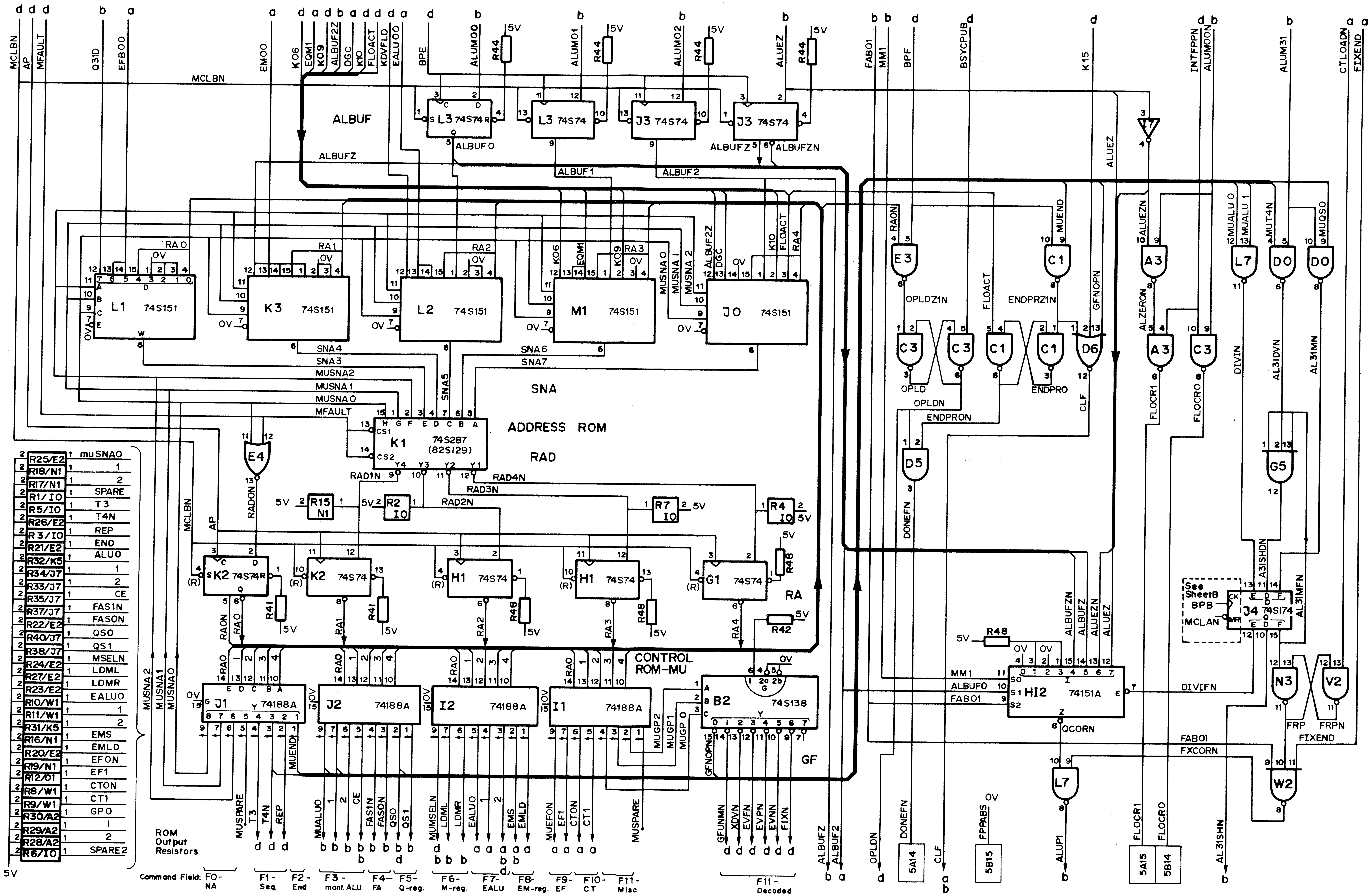


Figure D-7c FPP Logic Diagram

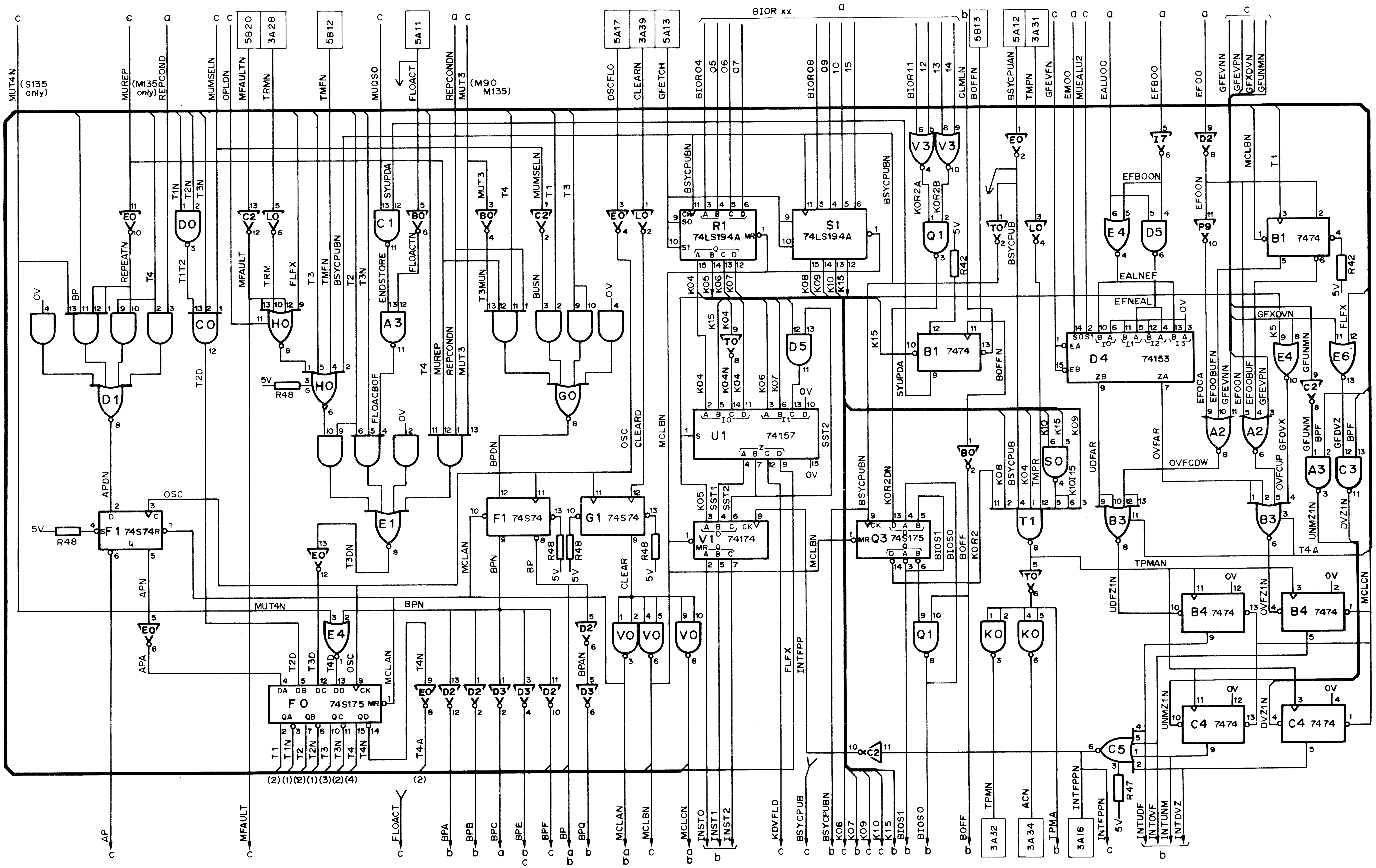


Figure D-7d FPP Logic Diagram

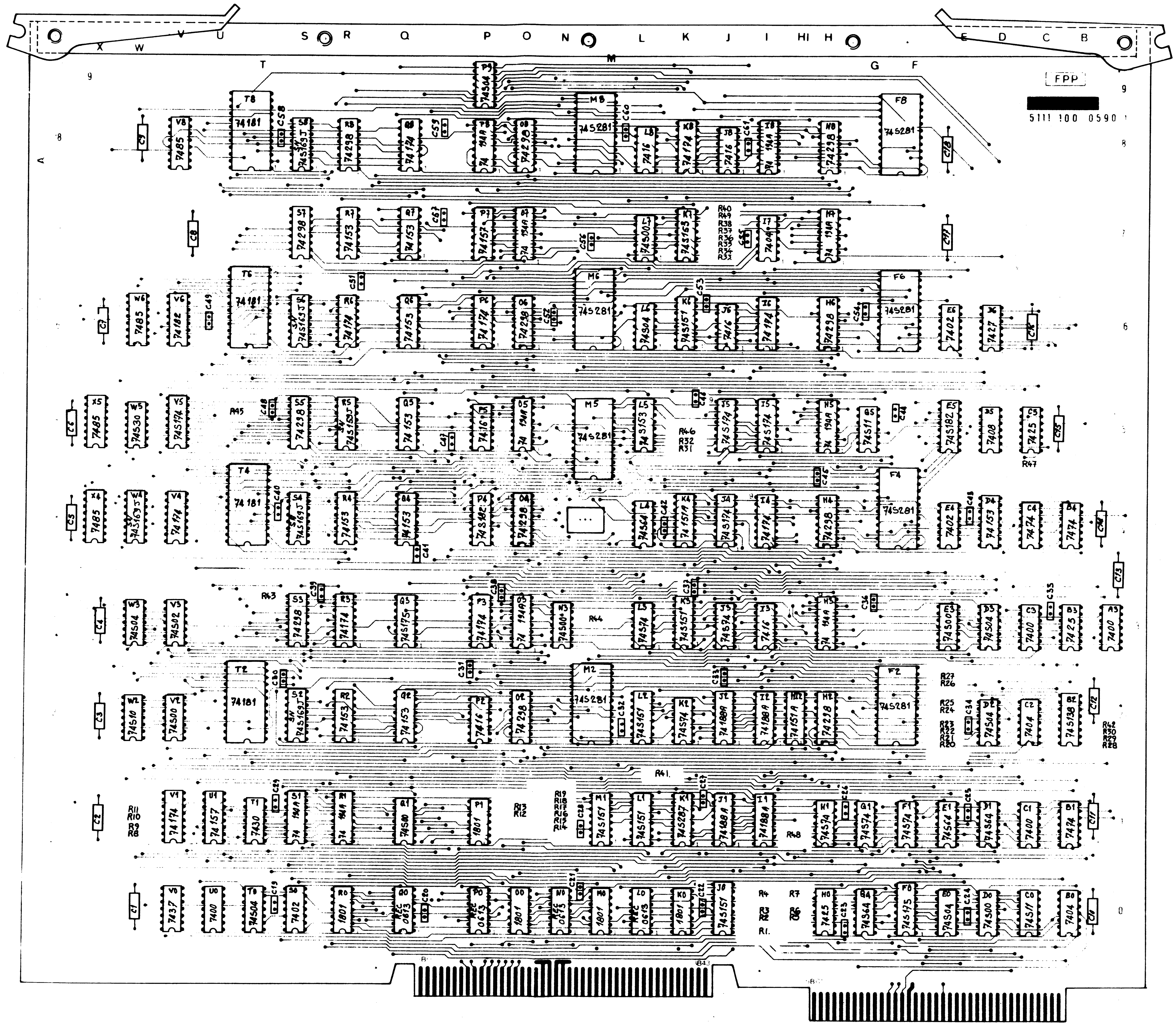


Figure D-8 FPP Card Layout

Table D-5 FPP Parts List

Reference	Description	12NC Code
	Printed circuit	5111 100 05901
A3, C1, C3, U0.	Integrated circuit 7400	
E4,E6,S0.	Integrated circuit 7402	
B0,C2,I7.	Integrated circuit 7404	
D5.	Integrated circuit 7408	
I3, J6, J8, L8, P2, P5.	Integrated circuit 7416	
B3,C5,H0.	Integrated circuit 7425	
D6.	Integrated circuit 7427	
T1.	Integrated circuit 7430	
V0.	Integrated circuit 7437	
B1,B4,C4.	Integrated circuit 7474	
V8,W6,X4,X5.	Integrated circuit 7485	
B2.	Integrated circuit 74S138	
H12,K4.	Integrated circuit 74151A	
D4,Q2,Q4-7,R2,R4,R7.	Integrated circuit 74153	
P7, U1.	Integrated circuit 74157	
I4,I6,K8,P3,P6,Q8,R3,R6,V1,V4.	Integrated circuit 74174	
T2,T4,T6,T8.	Integrated circuit 74181	
V6.	Integrated circuit 74182	
C0,G5.	Integrated circuit 74S11	
K1.	Integrated circuit 2721 (82S129)	
H2,H4,H6,H8,O2,O4,O6,O8,R8,S3,S5, S7.	Integrated circuit 74298	
D0,E3,L7,N3,Q1,V2.	Integrated circuit 74S00	
V3.	Integrated circuit 74S02	
D2,D3,E0,L6,P9,T0,W3.	Integrated circuit 74S04	
W2.	Integrated circuit 74S10	
W5.	Integrated circuit 74S30	
D1,E1,G0,L4.	Integrated circuit 74S64	
F1,G1,H1,J3,K2,L3.	Integrated circuit 74S74	
J0,K3, K6,L1,L2,M1.	Integrated circuit 74S151	
K7, L5.	Integrated circuit 74S153	
I5,J4,J5,V5.	Integrated circuit 74S174	
E5,P4.	Integrated circuit 74S182	
F2,F4,F6,F8,M2,M5,M6,M8.	Integrated circuit 74S281	
H3,H5,H7, I8,O3,O5,O7,P8,R1,S1.	Integrated circuit 74LS194A	
F0,Q3.	Integrated circuit 74S175	
K0,M0,O0,P1,R0.	Integrated circuit 1801	
L0,N0,P0,Q0.	Integrated circuit REC0613	
R5,S2,S4,S6,S8,W4.	Integrated circuit SN74S169J	
J1.	Integrated circuit 2681 (74188A)	
J2.	Integrated circuit 2691 (74188A)	
I2.	Integrated circuit 2701 (74188A)	
I1.	Integrated circuit 2711 (74188A)	

Table D-5 FPP Parts List contd.

Reference	Description	12NC Code
C1-18.	Capacitor 10µF, 25V FITCO.	
C19-49,51-61.	Capacitor 10µF.	
R1,3,5,6,8,9,19-24,26-30,32-34,36,37, 38,40-48.	Resistor 1KΩ, 1/8W, 5%.	
R39.	Resistor 220Ω, 1/8W, 5%.	
R2,4,7,14,15.	Resistor 470Ω, 1/8W, 5%.	
R10-12,16-18,35.	Resistor 4.7KΩ, 1/8W, 5%.	
R25,31.	Resistor 10KΩ, 1/8W, 5%.	

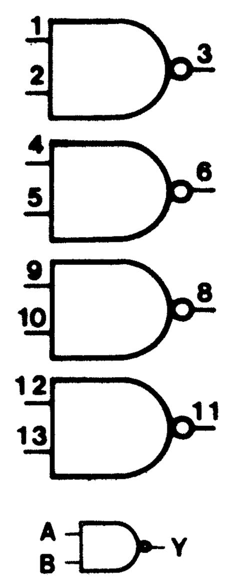
1801

(see 7400)

7400

7400
S00
H00
03[1]
37[2]
38
1801[1]

1 open collector
2 buffer

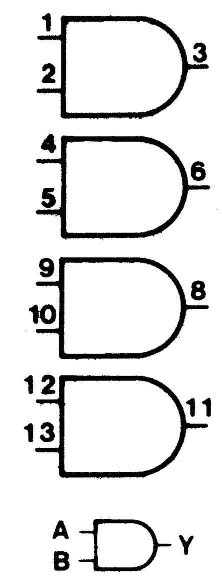


Y=$\overline{AB}$ Vcc 14 GND 7

7408

7408
09[1]

1 open collector



Y=$\overline{AB}$ Vcc 14 GND 7

7416

(see 7400)

2681

2691

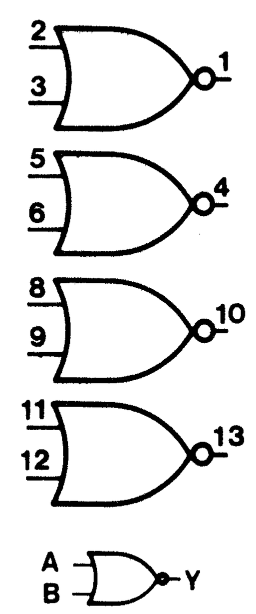
2701

2711

(see 74188)

7402

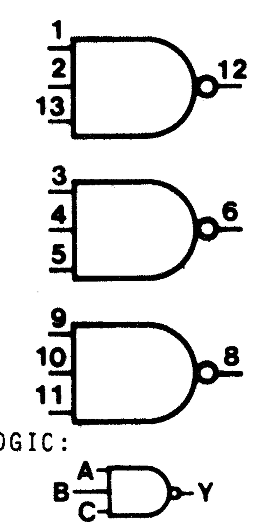
QUADRUPLE 2-INPUT POSITIVE-NOR GATE



Vcc 14 GND 7

7410

7410



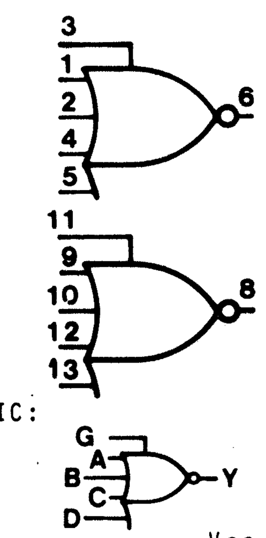
POSITIVE LOGIC:
Y=$\overline{ABC}$

Vcc 14 GND 7

7425

7425

4-INPUT NOR WITH STROBE



POSITIVE LOGIC:
Y=$\overline{G(A+B+C+D)}$

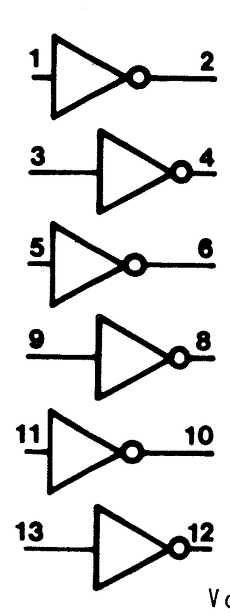
Vcc 14 GND 7

2721

(see 82S129)

7404

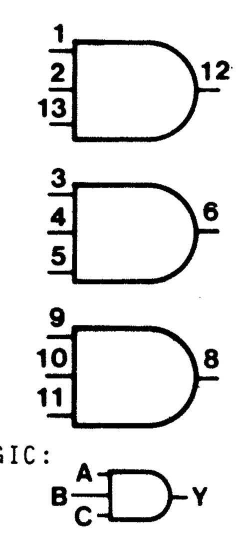
7404, S04, H04
7406 o.c. 30V
7414 sh Tris
7416 o.c. 15V



Vcc 14 GND 7

7411

7411

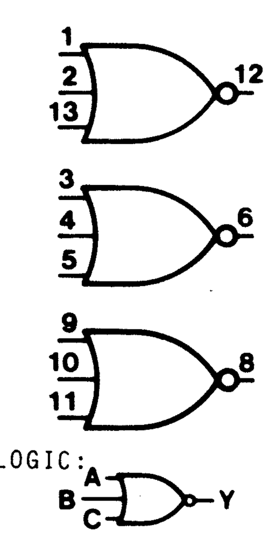


POSITIVE LOGIC:
Y=ABC

Vcc 14 GND 7

7427

7427



POSITIVE LOGIC:
Y=$\overline{A+B+C}$

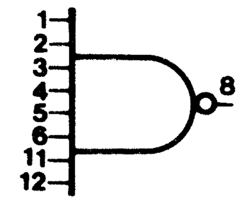
Vcc 14 GND 7

Figure D-9a FPP IC Guide

7430

7430

8-INPUT NAND GATE



Vcc 14
GND 7

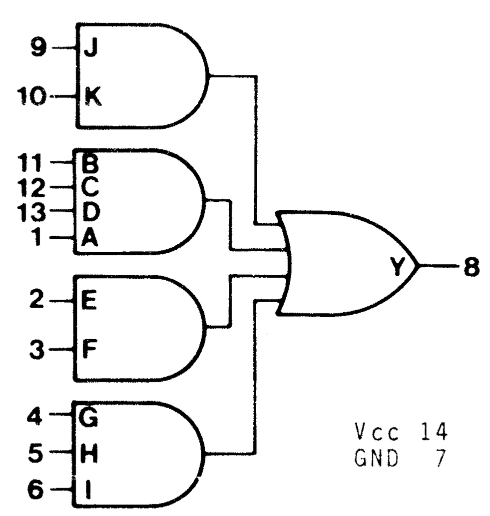
N.C. 9,10,13

7437

(see 7400)

7464

4-2-3-2-INPUT AND-OR-INVERT GATES



Vcc 14
GND 7

FUNCTION TABLE

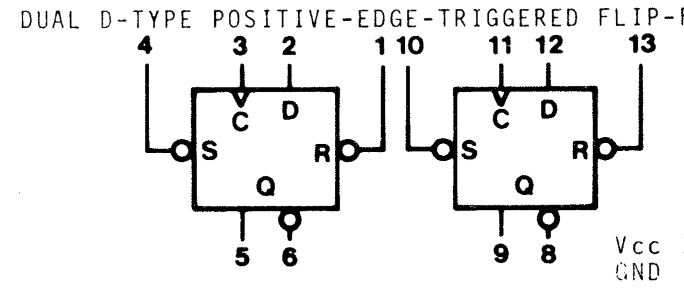
INPUTS					OUTPUTS	
PRESET	CLEAR	CLOCK	J	K	Q	Q
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H	H
H	H	↑	L	L	Qo	Qo
H	H	↑	H	L	H	L
H	H	↑	L	H	L	H
H	H	↑	H	H	TOGGLE	
H	H	L	X	X	Qo	Qo

Positive Logic: Y=ABCD+EF+GHI+JK

7474

7474, H74, S74

DUAL D-TYPE POSITIVE-EDGE-TRIGGERED FLIP-FLOPS

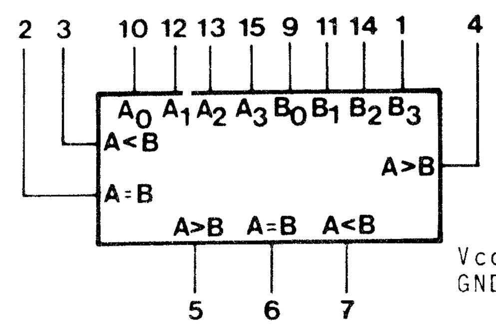


Vcc 14
GND 7

FUNCTION TABLE					
INPUTS				OUTPUTS	
S	R	C	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q_o	Q_o

7485

7485 4-BIT MAGNITUDE COMPARATOR



Vcc 16
GND 8

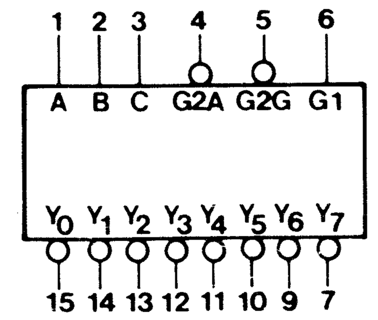
FUNCTION TABLES

COMPARING INPUTS				CASCADING INPUTS			OUTPUTS		
A3,B3	A2,B2	A1,B1	A0,B0	A>B	A<B	A=B	A>B	A<B	A=B
A3>B3	X	X	X	X	X	X	H	L	L
A3<B3	X	X	X	X	X	X	L	H	L
A3=B3	A2>B2	X	X	X	X	X	H	L	L
A3=B3	A2<B2	X	X	X	X	X	L	H	L
A3=B3	A2=B2	A1>B1	X	X	X	X	H	L	L
A3=B3	A2=B2	A1<B1	X	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0>B0	X	X	X	H	L	L
A3=B3	A2=B2	A1=B1	A0<B0	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	H	L	L	H	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	H	L	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	H	L	L	H
'85,'S85									
A3=B3	A2=B2	A1=B1	A0=B0	X	X	H	L	L	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	L	L	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	L	H	H	L
'L85									
A3=B3	A2=B2	A1=B1	A0=B0	L	H	H	L	H	H
A3=B3	A2=B2	A1=B1	A0=B0	H	L	H	H	L	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	H	H	H	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	L	H	H	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	L	L	L	L

H=high level; L=low level; X=irrelevant

74138

DECODER



Vcc 16
GND 8

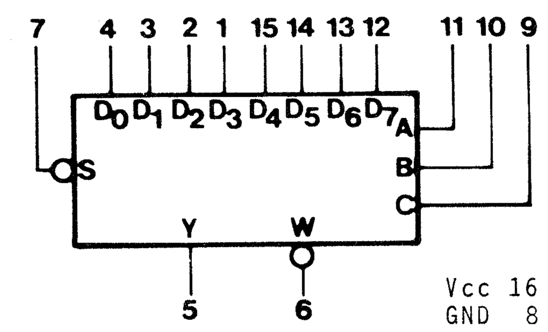
FUNCTION TABLE

INPUTS					OUTPUTS							
ENABLE		SELECT										
G1	G2	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

'G2 = G2A+G2B

74151

DATA SELECTOR/MULTIPLEXER



Vcc 16
GND 8

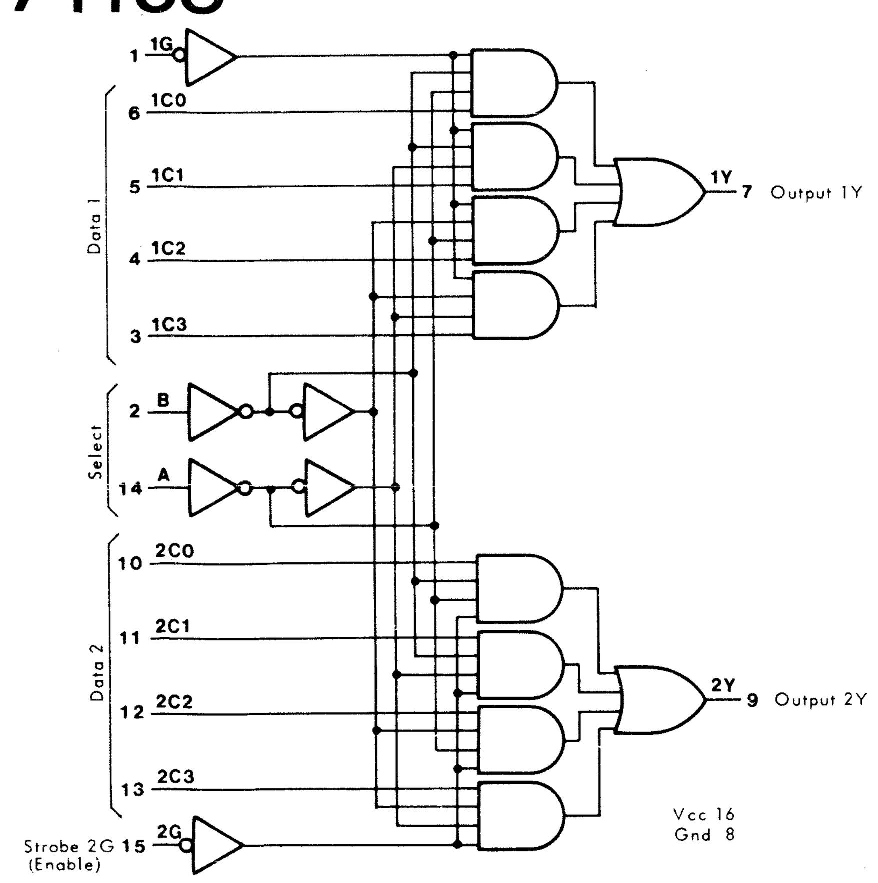
FUNCTION TABLE

INPUTS				OUTPUTS	
SELECT			STROBE		
C	B	A		Y	W
X	X	X	H	L	H
L	L	L	L	D0	D0
L	L	H	L	D1	D1
L	H	L	L	D2	D2
L	H	H	L	D3	D3
H	L	L	L	D4	D4
H	L	H	L	D5	D5
H	H	L	L	D6	D6
H	H	H	L	D7	D7

D0,D1...D7 = the level of the D respective input

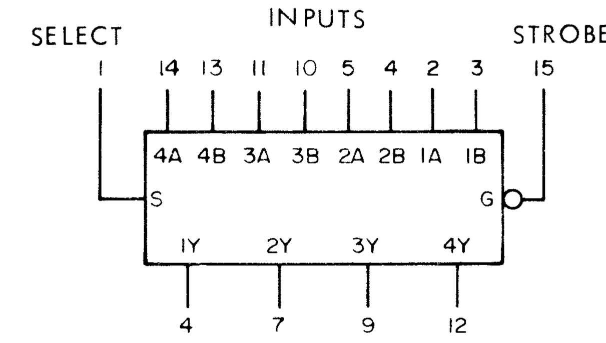
74153

DUAL 4-LINE-TO-1-LINE DATA SELECTOR/MULTIPLEXER



74157

QUAD 2-to-1-LINE DATA SELECTORS/MULTIPLEXERS

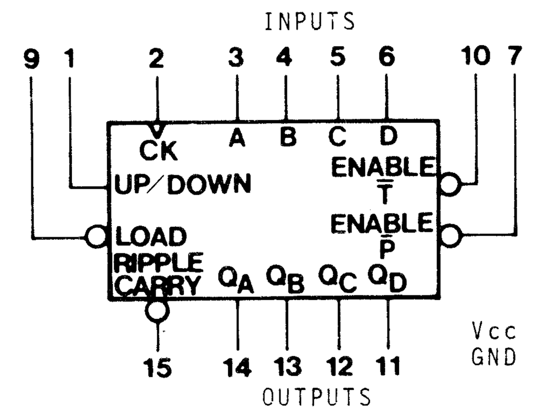


Vcc 16
GND 8

74169

74169

SYNCHRONOUS 4-BIT UP/DOWN COUNTERS

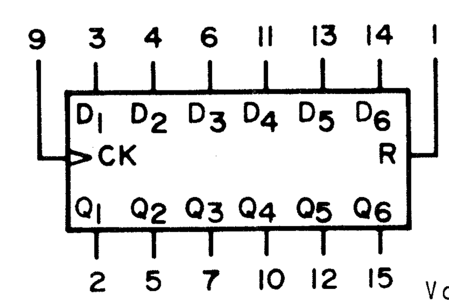


Vcc 16
GND 8

74174

74174

HEX D-TYPE FLIP-FLOPS WITH COMMON CLOCK AND RESET



Vcc 16
GND 8

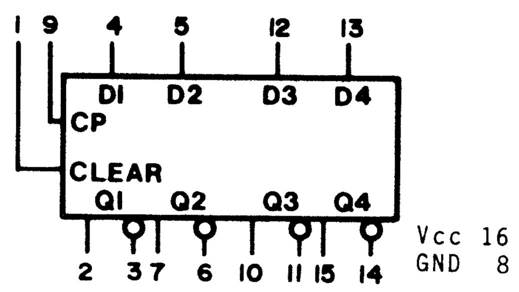
FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\overline{Q}$t
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	$\overline{Q}$o

Figure D-9b FPP IC Guide

74175

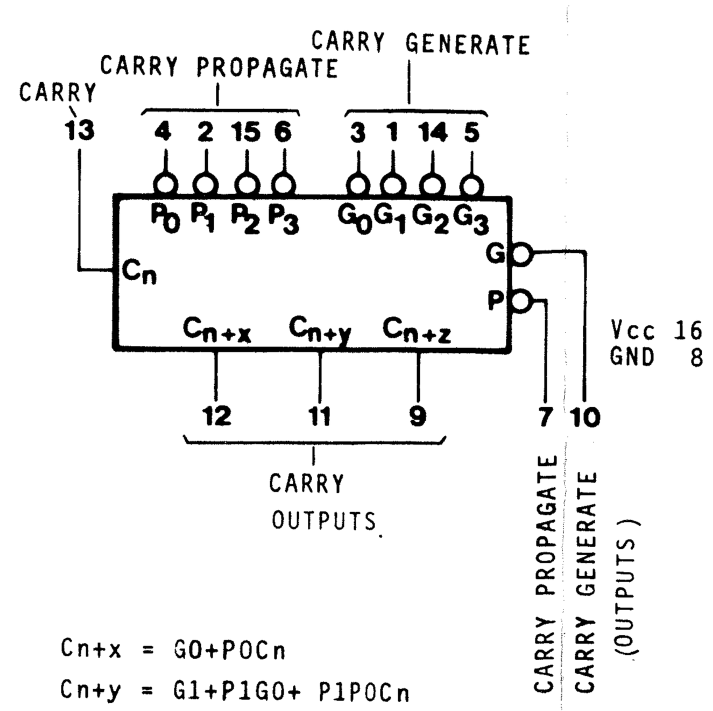
74175

QUAD D-TYPE FLIP-FLOPS



FUNCTION TABLE				
INPUTS			OUTPUTS	
R	C	D	Q	$\bar{Q}$
L	X	X	L	H
H	↑	H	H	L
H	↑	L	L	H
H	L	X	Qo	$\bar{Q}$o

74182 LOOK-AHEAD CARRY GENERATOR



Cn+x = G0+P0Cn

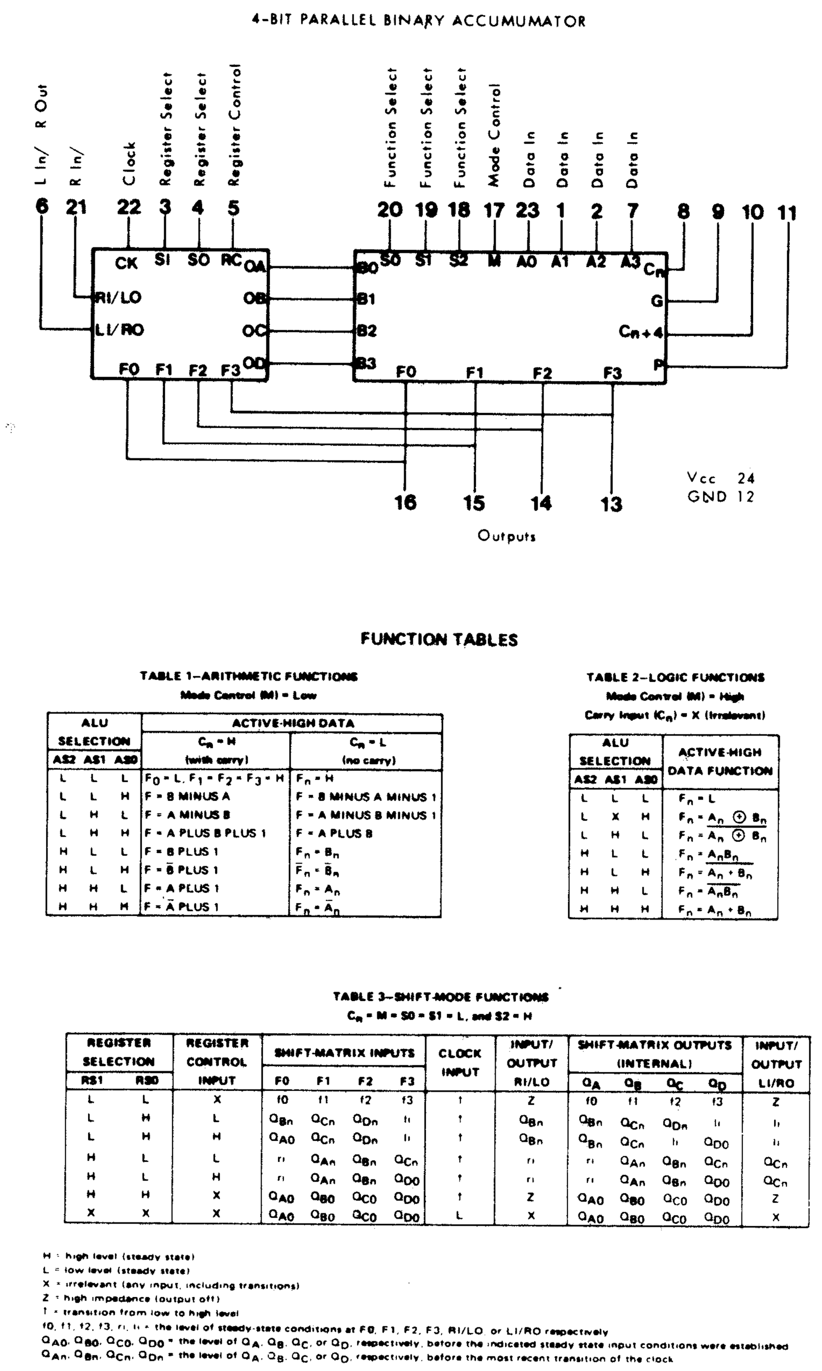
Cn+y = G1+P1G0+ P1P0Cn

Cn+z = G2+P2G1+P2P1G0+P2P1P0Cn

G = G3(P3+G2) (P3+P2+G1) (P3+P2+P1+G0)

P = P3P2P1P0

74281

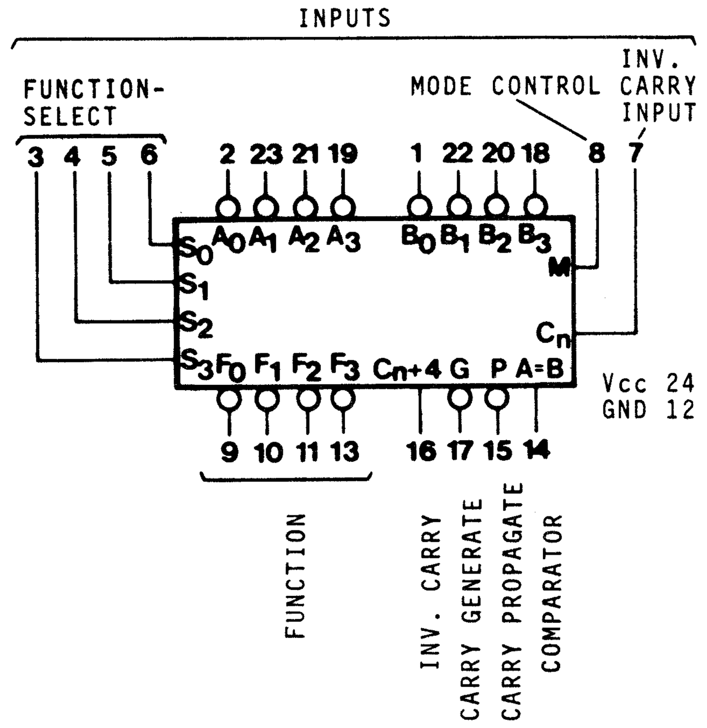


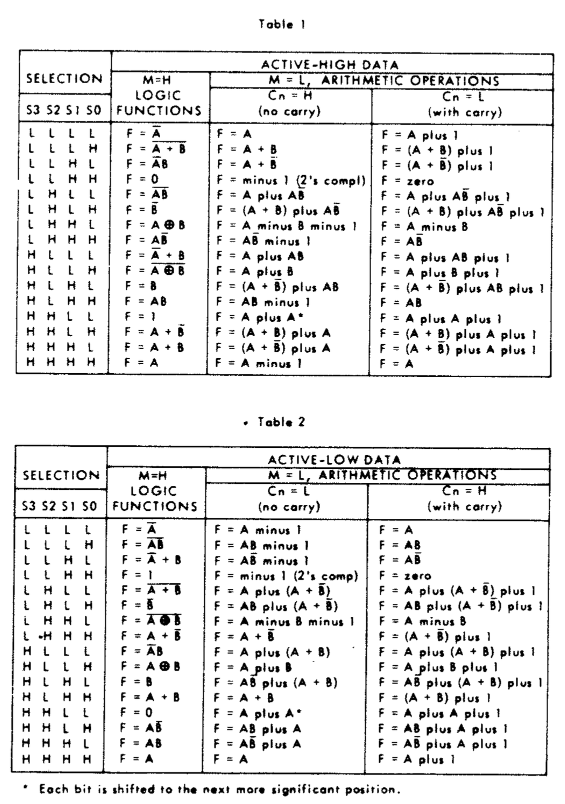
REC 0613

(see 7404)

74181

ARITHMETIC LOGIC UNIT

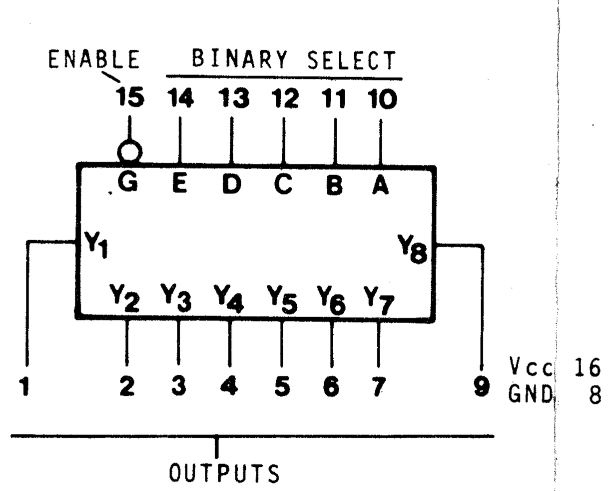




74188

74188A

256-BIT PROGRAMMABLE READ-ONLY MEMORY

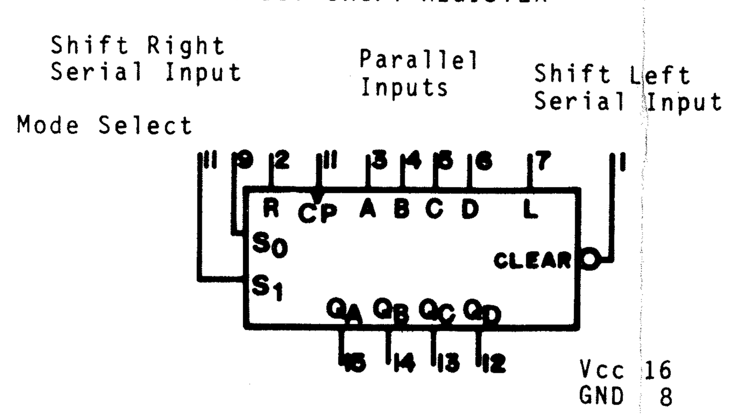


Organized as 32 Words of 8 Bits Each

74194

74194

4-BIT SHIFT REGISTER



82129

1024-BIT BIPOLAR PROGRAMMABLE ROM

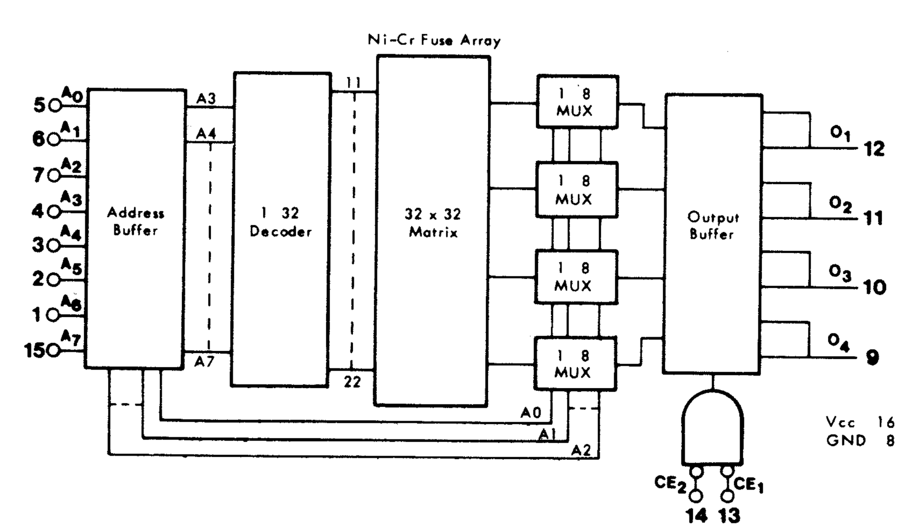


Figure D-9c FPP IC Guide