B 1720 SERIES CENTRAL SYSTEM

TECHNICAL MANUAL

Burroughs

FIELD ENGINEERING

AA401141 AA470338

Printed in U. S. America
10/75
For Library Binder 113
Form 1066941

Correspondence regarding this document should be forwarded using the Remarks Form at the back of the manual, or may be addressed directly to Systems Documentation, Technical Information Organization, TIO-Central, Burroughs Corporation, Burroughs Place, Detroit, Michigan 48232.

Burroughs

PUBLICATION CHANGE NOTICE

PCN No.: 1066941-002 Date: August 1978

Publication Title: B 1720 Series Central System Technical Manual

Other Affected Publications: None

Supersedes:

Description:

Pages Added

4-1 thru 4-33

F.E. Dist. Code BB

Printed in U. S. America

August 1978

Library Binder 241

PCN 1066941-002

LIST OF EFFECTIVE PAGES

Page	Issue
Title	Original
ii	Original
iii	**PCN-002**
iv	**PCN-002**
v thru x	Original
xi thru xii	PCN-002
xiii	PCN-001
xiv thru xviii	Original
xiv	PCN-001
xx thru xxi	PCN-002
xxii	Blank
1-1 thru 1-7	Original
1-8	Blank
1-9 thru 1-79	Original
1-80	Blank
1-81 thru 1-85	Original
1-86	Blank
1-87	Original
1-88	Blank
1-89 thru 1-107	Original
1-108	Blank
2-1 thru 2-17	Original
2-18	Blank
2-19	Original
2-20	Blank
2-21 thru 2-177	Original
2-178	Blank
2-179	Original
2-180	Blank
2-181 thru 2-300	Original
4-1 thru 4-33	PCN-002
4-34	Blank
5-1 thru 5-33	Original
5-34	Blank
5-35 thru 5-63	Original
5-64 thru 5-66	PCN-001
6-1 thru 6-24	Original

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Cont)

LIST OF ILLUSTRATIONS (Cont)

LIST OF ILLUSTRATIONS (Cont)

LIST OF ILLUSTRATIONS (Cont)

LIST OF ILLUSTRATIONS (Cont)

LIST OF ILLUSTRATIONS (Cont)

LIST OF ILLUSTRATIONS (Cont)

LIST OF TABLES

SECTION 1

INTRODUCTION AND OPERATION

INTRODUCTION

The B 1720 Series Central System is a small- to medium- scale data processing system. It is similar to the B 1710 Series Systems in overall operating concepts, but offers improved speed of operation through use of a higher clock rate and additional hardware. Additional features of the B 1720 Series System include an M-(micro) memory processor, an S-(system) memory, a port interchange, and a "soft" I/O subsystem.

MICRO INSTRUCTIONS

The B 1720 Series Central System performs its operations by executing a set of low-level (micro) instructions which are fetched from the M-memory. A micro may be defined as an instruction which causes the least amount of work possible to be performed within the system. As such, it represents a component part of the functions necessary to make up a complete machine operation. There are 32 of these micro instructions for performing the necessary machine functions. The micros are composed of 16-bit fields, and are decoded within the processor. The outputs thereby produced activate the appropriate registers, pseudo registers, logic, and arithmetic sections to cause the desired action to be performed.

MACRO-INSTRUCTIONS

A macro instruction is the maximum amount of work that can be done by one instruction. This corresponds to a "complete machine operation" in other types of equipment.

In the B 1700 these macro instructions are "built" by the programmer from a series of micro-instructions. An example of this would be an add operation, where a Macro performs a complete add with automatic storage of the sum and detection of carries. There is no micro-instruction which does a complete add. However, it is possible, through a series of selected micro instructions to cause the machine to perform all the necessary actions which comprise an add. Therefore the add is "built" by the programmer.

MICRO-PROGRAMMING

Micro-programming, as used in the B 1720 Series processor, has several definite advantages. One advantage which is most obvious is ease in modification of the macro. That is, a macro can be changed through software rather than requiring a modification of the logic circuitry. The second advantage is that micro programming allows the hardware to be configured to suit the requirements of a particular language such as COBOL, FORTRAN, etc. Thus the system is much more versatile than those which utilize predetermined (inflexible) machine language instructions.

PHYSICAL CONFIGURATION

The B 1720 Series Central System consists physically of a double bay or triple bay cabinet (See figure 1-1). This cabinet houses four (or six) card chassis each having a maximum capability to accommodate 28 logic cards. All logic circuitry, plus both S- and M-memories are contained on the removable cards. This provides ease of troubleshooting and repair,

and it facilitates memory expansion in the field when required. Control of the system is from the front control panel and supervisory printer (SPO). Power for the logic elements is provided by logic power supplies, one of which is installed in the base of each cabinet bay. The outputs of these supplies are connected in parallel, and they operate in a master/slave mode. A separate memory power supply is provided for each unit of S-memory, and these can number from one to four in a given system, depending on the installed memory size.

NOTE

In systems employing three logic supplies, the supply located in the extension cabinet stands alone.

For ease of maintenance, the side panels of the cabinet are removable, and both front and rear covers are hinged. In addition, the logic power supplies may be extended on slides and tilted for access to the underside.

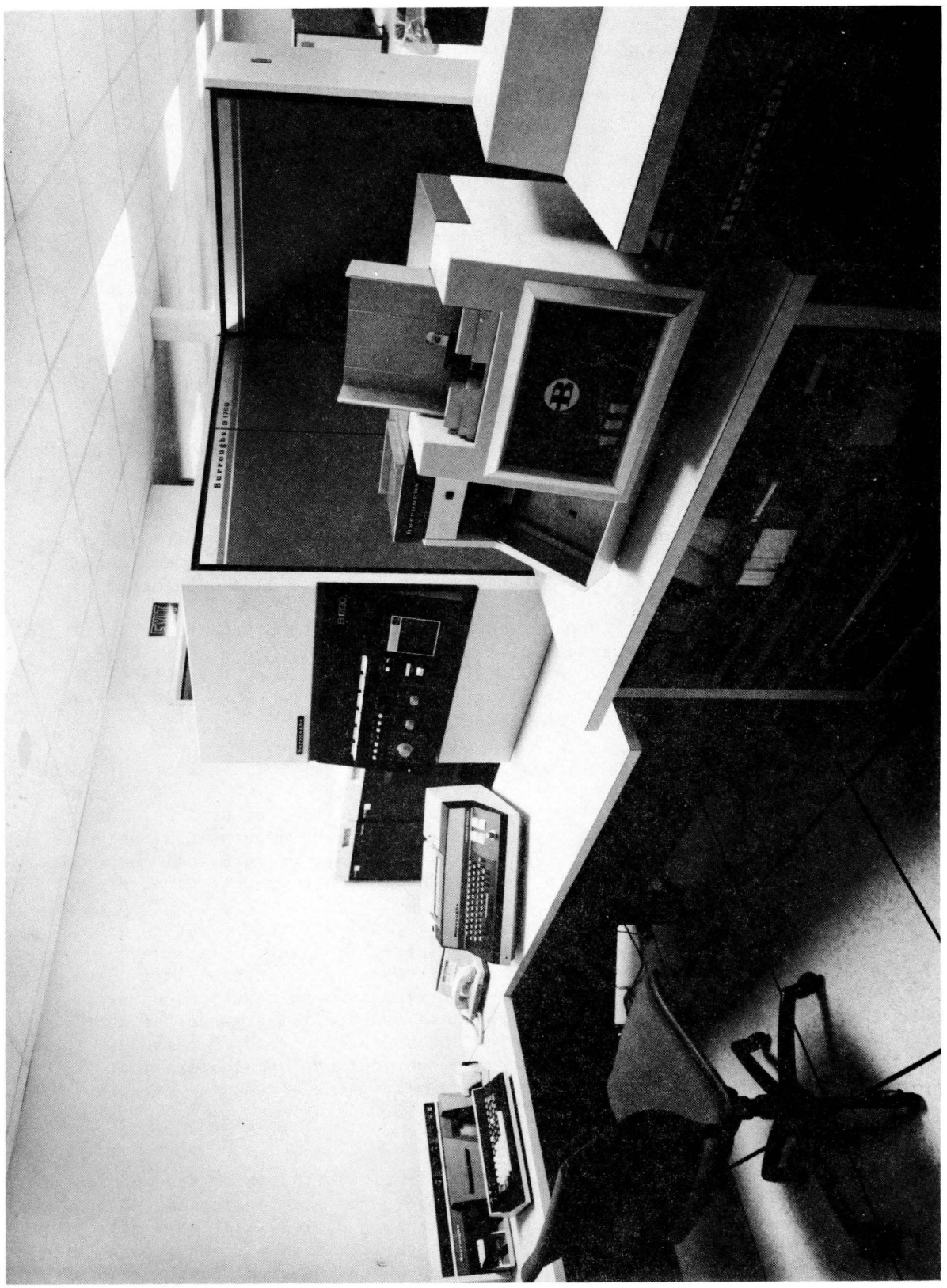


Figure 1-1. B 1720 Series Central System (Triple Bay Configuration)

SYSTEM ARCHITECTURE

The B 1720 Series Central System is divided into the following functional areas:

a. M-memory processor

b. I/O subsystem

c. Port interchange

d. Main (S) memory

The relationship of these sections is illustrated in figure 1-2. Although each is described in detail elsewhere in this section of the manual, a brief summary of the interrelationships which exist will be presented.

The M-memory processor is the heart of the B 1720 Series System. It contains all the necessary circuit elements to perform the arithmetic and data manipulation functions required of a data processing system. In addition to the M-memory which provides storage of micro-instructions, the processor contains micro decoding and execution logic, function boxes, registers, data movement and storage facilities, a control panel, and provisions for interfacing with other devices.

The I/O subsystem is basically the means provided by which the processor may be interfaced with external input and output devices. These may be any of the commonly used items such as a line printer, card reader/punch, disk drive, tape drives or single line data comm.

The peripheral complement of a given system depends on the requirements of the functions it is to perform, and may be chosen from a wide variety of available devices. Each such device has a corresponding control located in the I/O base. It is beyond the scope of this manual to provide a more detailed description of I/O operations. Therefore, refer to the I/O Base manual (Form No. 1053352), if further information is desired.

The port interchange is the interface between S-memory and the devices which use it, of which the processor is but one. This arrangement was necessary because the B 1720 Series System was designed to allow independent access to memory by up to three different devices. At the present time, only access by the processor and a multi-line control have been implemented. Disk-pack access may be utilized in the future. The port interchange performs all memory access operations, and is preset to give priority to user devices in a predetermined order.

S-memory is the main data storage unit within the system. It employs dynamic random access memory chips, and is modularized to permit easy changes in size. S-memory offers very fast access time, and a great deal of flexibility in storage and retrieval of data. It is used for a multitude of purposes in normal data processing operations. Note that peripheral devices such as magnetic tape or disk are also used for data storage. The manner of storage selected is mainly dependent on the rapidity with which access is required. M-memory is the fastest, but is limited in size and must be used over and over during normal operation. In addition, it is dynamic in nature, and does not store data unless the system is operating.

M-MEMORY PROCESSOR

The M-memory processor is composed entirely of integrated circuit chips, which are mounted on 12" X 14" removable logic cards. These cards are 16 in number, and are interconnected by way of point-to-point backplane wiring and multiconductor frontplane cables. In terms of logic circuits the processor comprises data storage facilities, paths for data movement, data manipulation devices, and control circuitry. Those portions which are significant are discussed individually in the following paragraphs. Since the console controls are, for the most part, connected directly to the processor logic, they are considered to be a part of the processor.

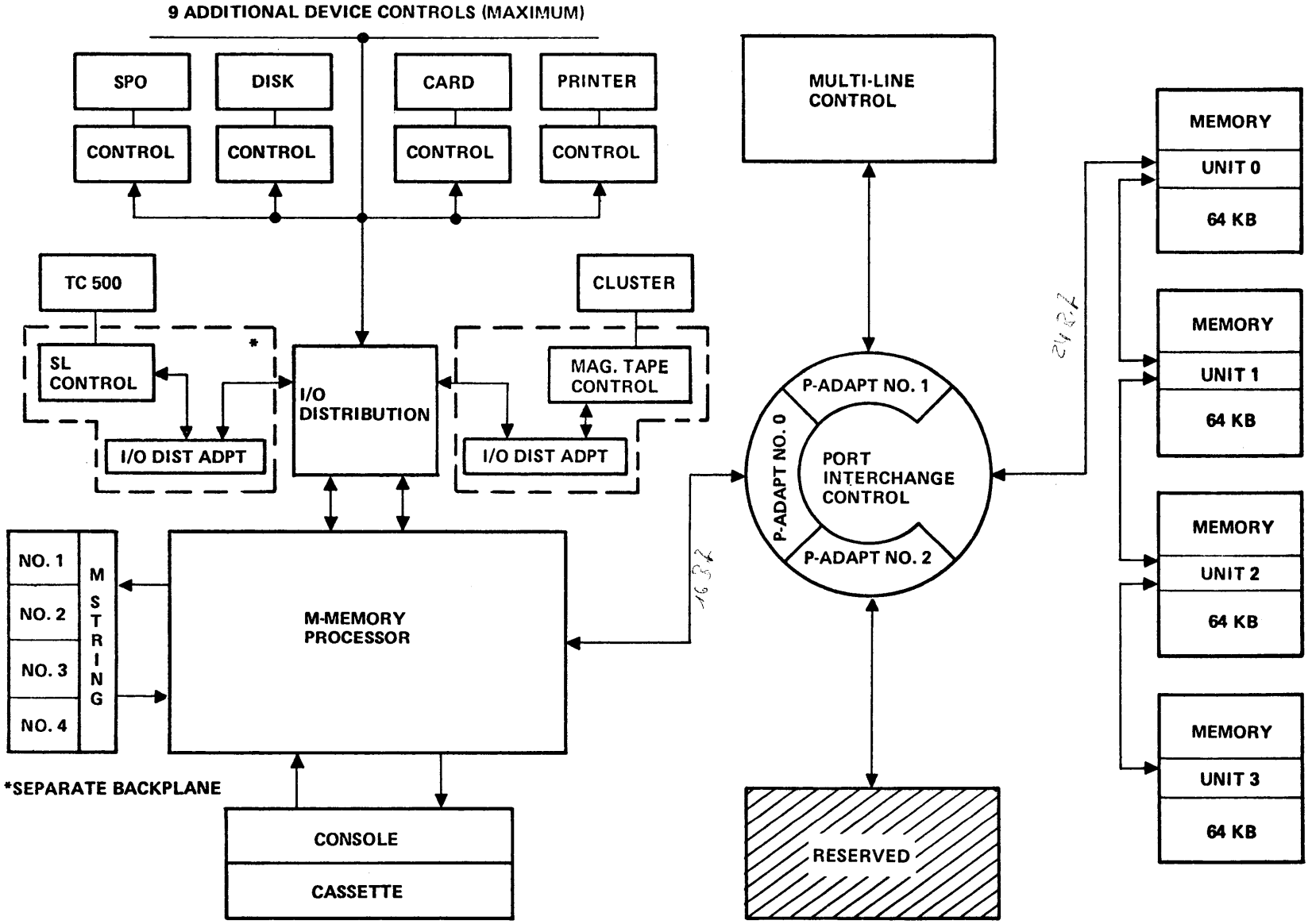


Figure 1-2. B 1720 Block Diagram

FUNCTIONAL LAYOUT

The M-memory processor is functionally arranged as shown in figure 1-3. Operationally, control of the machine is exercised via micro decoding. These micros are stored in the M-string memory, and in the absence of instructions to the contrary (from a micro itself) are fetched and executed in the sequence in which they were written into. Note that micro instructions and all other data are treated exactly alike within the processor, and are transported over the same routes. This is true except for M-memory, which is reserved for storage of micro instructions, and the M-register, which is the micro path to the decoding and execution logic. This latter section is truly the heart of the processor, and exercises control over all the other circuits. Since the micro instructions are handled as data, they may be modified, manipulated or even created by actions of the processor. It is on this basis that the systems operational concept involving high level programming languages, S-instructions, interpreters, etc., was developed.

INTERNAL PROCESSOR OPERATIONAL CONTROL (CURRENT STATES)

The B 1720 Series System Processor may be defined as a "current state" machine. This refers to the five operating states which may exist within the control logic, and which exert an overall determining influence on the actions which are permitted to occur. The current states are derived from conditions existing within the processor, and are determined for the most part by prior events. Each is characterized by a single logic term which serves to enable the proper circuitry for carrying out the actions possible in that state. Only one current state is true at any given time when the machine is operating. The current states may be defined as follows.

Console Current State (CONSCSP)

The processor is halted, and the load/display functions available from the console are enabled. This state may exist only when the processor is halted, and is therefore synonomous with halt.

Start Current State (STRTCSP)

This is a transient state entered for one clock pulse when the console START button is pressed. Start C/S serves as a delay to allow activation of logic in preparation for entering the execute current state, which always follows. In the tape mode, the cassett drive is started when STRTCSP is true.

Execute Current State (EXECCSP)

This is the normal operating state of the machine, in which continuous execution of micros takes place. Execute encompasses fetching micro-operators from M-memory, and executing them. This includes gating micros to the M-register from M-string memory (automatic) or from a processor register (in response to a micro specifying such). EXECCSP. may be entered from the start current state or fetch current state, and may exit to the fetch current state or stop current state.

Fetch Current State (FTCHCSP)

This state is entered from the execute current state when it is necessary to fetch a micro from some source other than M-memory. Fetching consists of automatic generation of control levels to access the desired source and gate the 16-bit field constituting a micro instruction to the M-register. All other processor operations are suspended when FTCHCSP. is true. Fetching occurs in the run mode whenever the address in A exceeds the value in TOPM, and results in sourcing micros from the S-memory location specified in A+MBR. In the tape mode, all micros executed are fetched from the U-register, which is fed by the cassette tape drive. The fetch current state may only be entered from, or exited to, the execute current state.

Stop Current State (STOPCSP)

This is a transient state entered for one clock pulse whenever one of the conditions which causes a system halt occurs (HALT button depressed, halt micro executed, etc). STOPCSP. stops the cassette drive if running, then exits to console current state.

CONSOLE

The front panel console (figure 1-4) provides a manual point of entry into the system from which overall control is maintained. Included are controls for ac power, the normal processor operating functions, plus facilities for displaying the contents of any register or portion of S- or M-memory, and for replacing that data with any desired pattern of bits. A tape cassette reader is also provided to input certain types of program material and other functions to which it is suited. A brief description of each control's function is given below in the following paragraphs.

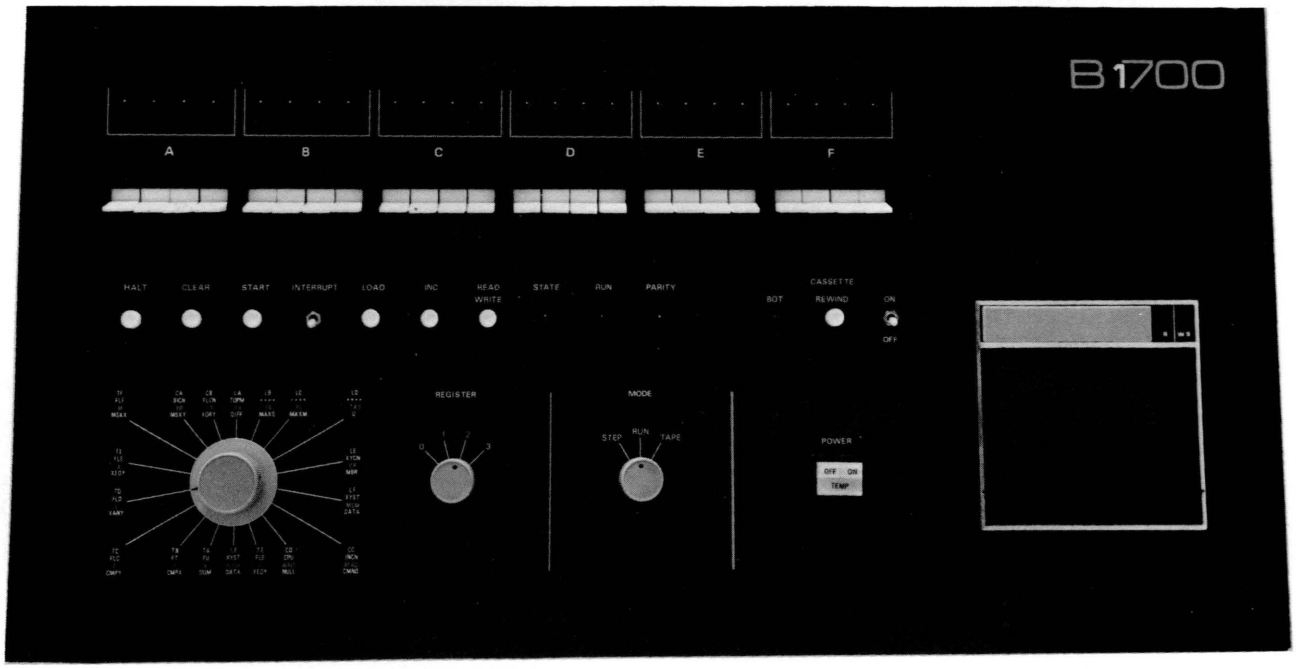


Figure 1-4. B 1720 Console

POWER ON/OFF/TEMP Switch

Controls the application of ac power to the system. The switch incorporates a three-way indicator which displays the system power status. These are as follows:

ON light: DC power is up (system operational).

OFF light: DC power down, but ac power available. This is the normal off condition of the system when it is not in use.

TEMP light: DC power cut off due to failure of one or more of the ventilating fans.

No Indication: AC power not applied to the system.

STATE Indicator

Indicates that bit 3 of the CC register is set. This function is used programmatically to provide the operator with a visual indication that selected events are occurring.

RUN Indicator

Indicates that the processor is in the run mode.

PARITY Indicator

Indicates that a non-recoverable parity error has been detected, either while reading from the cassette tape or during a micro fetch. Note that the latter may occur only during fetches from S-memory since parity detection is not incorporated in the M-memory. In either case, the processor halts. On a cassette read, the tape also halts at the next inter-record gap.

24 Console Lamps

Provide a bit-by-bit display of the contents of the main exchange at all times. When the processor is halted, a hard wired register move operation is forced to cause the contents of the register designated by the group and REGISTER select switches to be displayed.

MODE Switch

Selects the operational mode of the processor. These are defined as follows:

STEP: Execution of a single micro occurs when the START button is pressed. Simultaneously, the next micro is fetched and stored in the M-register. This is repeated each time the START button is pressed.

RUN: Continuous fetching and execution of micros occurs when the START switch is pressed, with processor actions thereafter determined by the contents of the program.

TAPE: Transfers the source of micro-instructions to the tape cassette. Pressing the START button begins tape movement, and the processor executes micros from the tape.

REGISTER Select and Register Group Switches

The register group switch (large rotary switch at the lower left corner of the console), along with the adjoining REGISTER select switch is used to access the contents of any desired register for examination and/or alteration via the console lamps and console switches. The B 1700 registers are divided into groups of four which are selected by the register group switch. An individual register within a group is then selected by moving the REGISTER select switch to that register's relative position in its group. Note that read and write in S-memory may also be selected. Refer to table 1-1. Display and/or loading of registers may only be done with the processor halted.

HALT Pushbutton

Causes the processor to halt execution at the end of the present micro instruction or series of concurrent micro instructions. The next micro to be executed is fetched and stored in M (except in tape mode, in which the last micro executed remains).

CLEAR Pushbutton

Clears (resets) all flip-flops in the processor, port interchange, memory logic, and I/O controls to the defined clear state. This pushbutton is active only when the processor is in the halt mode. If the processor cannot be halted, the system may be cleared by simultaneously depressing the HALT and CLEAR buttons.

Table 1-1. Register Table

Register Group	Register Select 0	1	2	3
0	TA	FU	X	SUM
1	TB	FT	Y	CMPX
2	TC	FLC	T	CMPY
3	TD	FLD	L	XANY
4	TE	FLE	A	XEOY
5	TF	FLF	M	MSKX
6	CA	BICN	BR	MSKY
7	CB	FLCN	LR	XORY
8	LA	TOPM	FA	DIFF
9	LB	RES	FB	MAXS
10	LC	RES	FL	MAXM
11	LD	RES	TAS	U
12	LE	XYCN	CP	MBR
13	LF	XYST	MSM	DATA
14	CC	INCN	C READ	CMND
15	CD	CPU	C WRITE	NULL

NOTES: 1. RES = RESERVED (NOT USED)

2. READ, WRITE AND MSM ARE NOT REGISTERS, BUT ARE TREATED AS SUCH FOR CONVENIENCE

START Pushbutton

Initiates execution of micros in the selected mode.

INTERRUPT Switch

Sets bit 0 in the CC register. The interrupt condition is handled as determined by software.

LOAD Pushbutton

Gates the contents of the 24 console switches to the register designated by the REGISTER select and register group switches. Active only in the halt state.

INC Pushbutton

Causes the address in the FA or A-register to be incremented by binary 16 (one word address) when performing console operations. FA is affected when READ or WRIT is selected (reading or writing in S-memory), with A being incremented when MSM is designated (reading or writing in MSM). Active only when the processor is halted.

READ/WRITE Pushbutton

Causes the contents of the S-memory location specified by the address in FA to be read and gated to the 24 console lamps for display (when READ register is selected). It also causes the contents of the 24 console switches to be gated to, and written into S-memory at the location specified by the address in FA (when WRIT register is selected). Active only when the processor is halted.

24 Console Switches

Provide manual selection of a 24-bit data field for loading in a selected register or writing into M- or S-memory.

BOT Indicator

Indicates that the console cassette tape cartridge is positioned at the beginning of the tape (or at the end of the tape).

REWIND Pushbutton

Causes the console cassette tape cartridge to rewind to the beginning of tape.

CASSETTE ON/OFF Switch

Controls power for the console cassette.

DATA PATHS

Movement of data within the processor is by three different paths: A 24-bit main exchange (MEX), a 4-bit auxiliary exchange and a 4-bit result exchange.

24-Bit Main Exchange (MEX)

This is main data path within the processor, and is used for the majority of functions which involve movement of data. All major portions of the processor are tied to the MEX. It is 24 bits wide, and all 24 bits are moved simultaneously. It is possible to utilize less than the entire width of the MEX.

4-Bit Auxiliary Exchange

Used for movement of 4-bit data fields from the 4-bit registers to the 4-bit function box. Note that some of the larger registers may be addressed in 4-bit segments as individual sub-registers.

4-Bit Result Exchange

This exchange receives the output of the 4-bit function box. Used for special handling of result bits which are highly critical in their timing. Serves as one of the two input sources for 4-bit registers. (The last significant 4-bits of the MEX is the other.)

M-STRING MEMORY

M-string memory is a high speed, all solid state memory which is used exclusively to store the micro operators which control processor operation. MSM is modular, and is available in four sizes:

2K bytes or 1024 16-bit words (one card) 6K bytes or 3072 16-bit words (three cards)

4K bytes or 2046 16-bit words (two cards) 8K bytes or 4096 16-bit words (four cards)

Access to M-memory is much faster than access to main (S) memory. This contributes to the overall greater speed with which the B 1720 Series operates. The M-memory is addressable as both a source and as a sink. Access to M-memory is normally gained by using the contents of the A-register as an address. Word addresses only are possible (16-bit increments), and counting up the A-register by one word is an automatic function each time a micro is fetched from M-memory. Exceptions to the use of A for M-memory addressing are with the 7E and 2F micro operators. See the descriptions of these micros for further details. The M-memory lay-out is shown in figure 1-5.

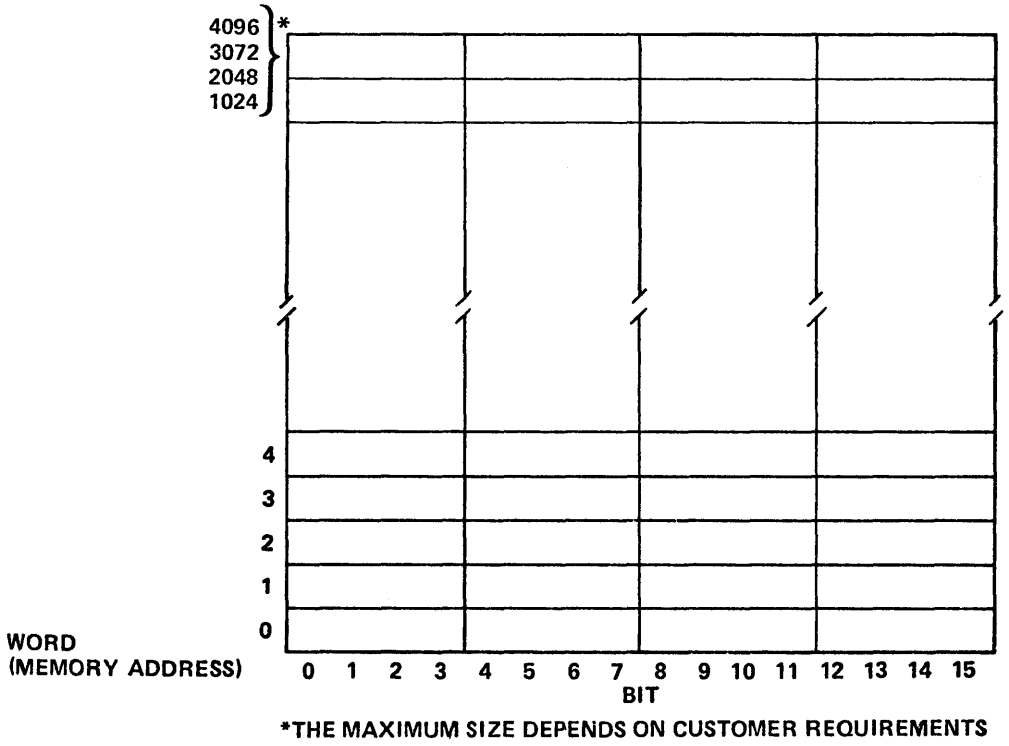


Figure 1-5. M-Memory Layout

The processor is wired to automatically cause S-memory to be used as an extension of M-memory when the "top of M" is reached. The top of M is specified by the TOPM register, and may be the actual physical limit of M-memory, or some lower address which is stored in TOPM programmatically. As a reference for use in setting TOPM, the size of installed M-memory is contained in another register known as MAXM. This is a wired-in value set by the field engineer.

M-Memory Cards

Each card of M-memory has a 1K (1024) 16-bit word storage capacity and consists of 64 units of 256 x 1-bit memory chips (RFDN).

The 64 chips are placed in four rows of 16 chips each. Each row stores 256, 16-bit words, and the rows are addressed in sequential order to access all 1024 words of storage. Refer to figure 1-6.

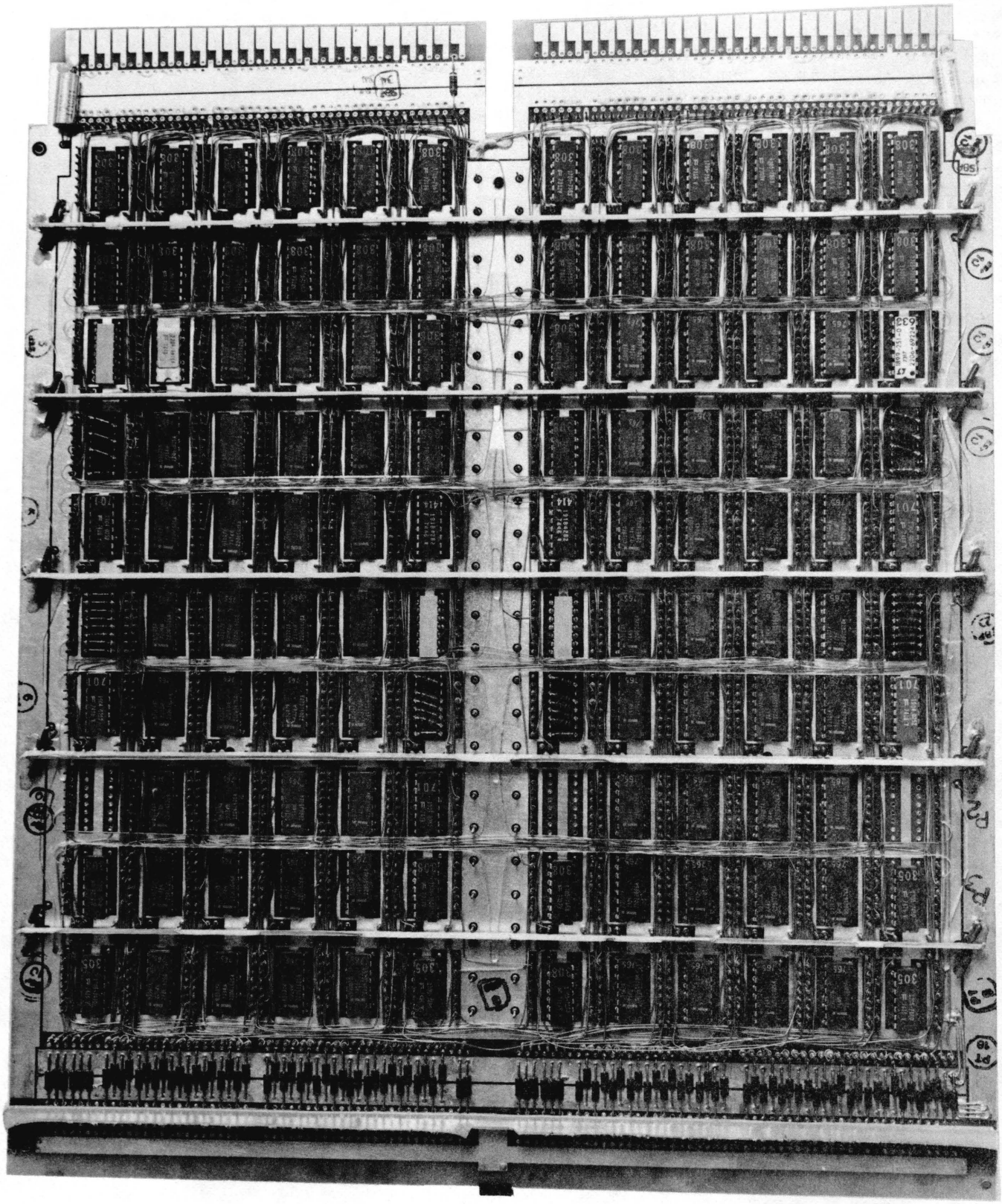


Figure 1-6. M-Memory Card

M-REGISTER

The M-register is a 16-bit discrete register used exclusively to hold the micro operator currently being executed. As such it is the gateway to the micro decoding logic which controls all processor activity. The M-register is addressable as both a source and as a sink. When it is used as a sink, the incoming data is bit ORed with the incoming micro, thereby modifying it. Note that this is not true in the tape mode of operation. M is broken into four fields for decoding, and these are known as MC, MD, ME, and MF. The MC field contains the most significant four bits (15 through 12). Figure 1-7 illustrates the bit configuration of the M-register as it would appear when viewed on the console lamps. Note that the A and B fields of the console lamps are not applicable.

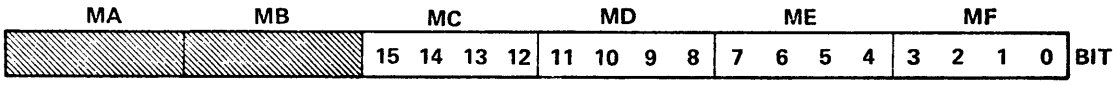


Figure 1-7. M-Register as Viewed on Console Lamps

SYSTEM CLOCK

The system clock is used to provide synchronization of all logic activity within the processor, and as such generally serves as a trigger for the circuits which have been enabled by micro decoding. The raw clock signal is generated by a crystal controlled oscillator operating at a frequency of 6 megahertz. This approximate square wave output from the oscillator module feeds two delay lines which are in series. These have multiple taps from which the desired signal propagation time may be selected. From the delay line taps, by way of jumper chips, the system clock, early clock, and adjustable clock signals are derived. These three signals are fed to a string of buffers to produce a total of 24 independent clock outputs. From the clock outputs, the signals are distributed directly to the processor logic cards by way of coaxial cables.

The relationship between the system clock, early clock, and adjustable clock signals is illustrated in figure 1-8. The three signals are generally used within the processor as follows:

System clock: General synchronization of logic activity within the system.

Early clock: Logic activation where propagation time or other factors require that certain actions be initiated early.

Adjustable clock: Delayed completion for activities which take longer than "normal".

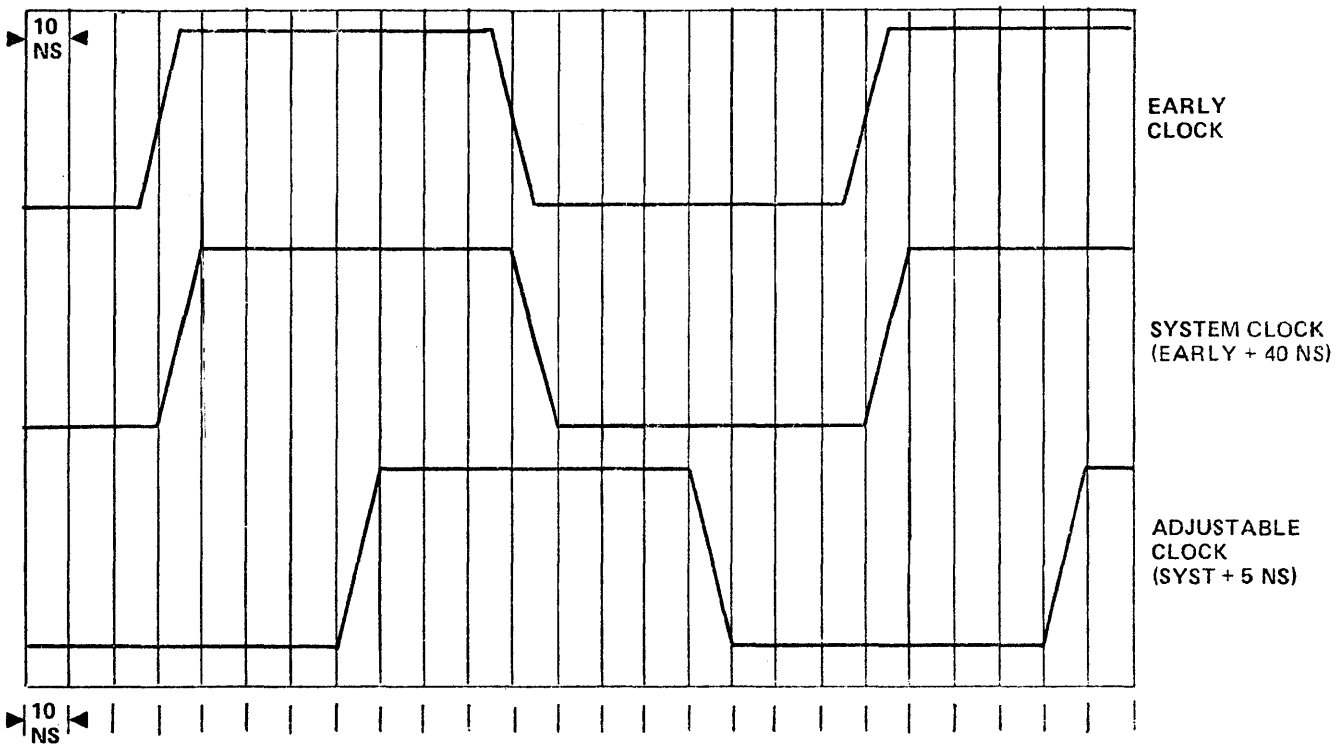


Figure 1-8. Clock Signal Relationships

24-BIT FUNCTION BOX

The 24-bit function box (24-bit arithmetic and combinatorial unit) is the main device for processing numerical data within the system, and is used for most operations which require arithmetic or logical manipulation of two operands. These operands may be up to 24 bits in length, and use the X- and Y-registers as paths to the function box. The function box output is always gated to the main exchange. In brief the 24-bit function box consists of the following sections:

a. 24-bit binary adder/subtractor with carry logic.

b. Decoding logic for binary coded decimal (BCD) sum/difference results, and BCD carry correction logic.

c. Mask generator.

d. Status and condition result logic.

The operational purpose of these sections can be stated as follows:

The binary adder/subtractor performs the actual arithmetic function between the two operands. Both the input and output of this stage are in binary form. Generation of carry and borrow outputs is automatic. Although merely adding and subtracting may seem limited in scope, it should be noted that much higher level mathematical operations may be performed by combining this with other logical functions. Creating such operations is a programming function, and beyond the scope of this manual.

The BCD decoding logic is used to convert the binary adder/subtractor output to binary coded decimal form. This is generally done for data which is destined for transmission to an I/O device. Data for storage and use within the processor is generally left in binary form. Translation to BCD is necessary for I/O operations which involve a print-out because numerical values between 10 and 15 require generation of two characters whereas only one is used in hexadecimal.

The mask generator determines the field length of the output gated to the main exchange from the 24-bit function box. This is used for operations where the desired result involves only a portion of the total output produced, and serves to protect the selected destination (sink) from extraneous data. Control of field length originates from a value stored in the CPL portion of the C-register, and is determined programmatically.

The status and condition result logic provides outputs which reflect the logical relationships existing between the 24-bit operands stored in X and Y. As such it is used for analysis and interpretation functions. The outputs include complement of X, complement of Y, X-Y logical AND, X-Y logical OR, X-Y logical Exclusive OR, mask of X and mask of Y (those bits which are not masked).

24-Bit Function Box Operation

Operationally, the 24-bit function box may be regarded as a group of nine pseudo registers which correspond to the nine distinct operations it can perform. Obtaining a specified function consists only of placing the operands in X and Y, and sourcing the appropriate pseudo register. The operands may be any length from 1 to 24 bits. In all cases, the operand length is specified by the value contained in CPL, with zero fill in the most significant bit positions for values less than 24. This is accomplished by the mask generation logic.

24-Bit Results

The nine 24-bit psuedo registers are as follows: Note that the group is composed of the adder/subtractor outputs and the status and condition result logic outputs, previously mentioned.

SUM is a 24-bit pseudo register which is equal to the sum of the X- and Y-registers plus the CYF (carry flip-flop) register. The value in CPL specifies the length (number of bits) of both operands and the sum. If the sum is greater than the length specified by CPL, the carry level (CYL) is generated. The value in CPU (control program unit) designates the unit of data desired as the output. When CPU=00, the direct output is obtained (binary sum). When CPU=01, the binary sum is passed through the BCD decoding logic and a binary coded decimal sum is obtained. CPU values of 10 and 11 are undefined.

DIFF is a 24-bit pseudo register which is equal to the difference of the X-register (minuend) and Y-register (subtrahend). The value of CYF (carry flip-flop) is also subtracted from the X-register. CPL determines the length of the operand field. When X < Y or X = Y and CYF is true, the borrow level (CYD) is generated. As with SUM, CPU = 00 specifies a binary output and CPU = 01 specifies binary coded decimal.

CMPX — Complement of X is a 24-bit pseudo register which, when sourced, produces the ones complement of the X-register. CPL determines the length of the operand field.

CMPY — Complement of Y is a 24-bit pseudo register which, when sourced, produces the ones complement of the Y-register. CPL determines the length of the operand field.

XANY — X and Y is a 24-bit pseudo register which, when sourced, produces the AND function of the X- and Y-registers. The logical AND function is defined as follows: corresponding bits of the X- and Y-registers must both be ones to produce a one-bit in the corresponding bit position of the XANY pseudo register. CPL determines the length of the operand field.

XEOY — X exclusive OR-ed with Y is a 24-bit pseudo register which, when sourced, produces the exclusive ORed results of the X- and Y-registers. The exclusive OR function is defined as follows: When one but not both of the corresponding bits of the X- and Y-registers is equal to one, the corresponding bit in the 24-bit XEOY pseudo register is a one. CPL determines the length of the operand field.

MSKX — Mask of X is a 24-bit pseudo register which, when sourced, produces the contents of the X-register referenced by a mask. The length of the mask is determined by the value of CPL.

MSKY — Mask of Y is a 24-bit pseudo register which, when sourced, produces the portion of the contents of the Y-register referenced by the value of CPL.

XORY — X or Y is a 24-bit pseudo register which, when sourced, produces the Or function results of the X- and Y-registers. The OR function is defined as follows: when either a corresponding bit of the X- or Y-register is a one, or if the corresponding bits of both the X- and Y-registers are ones, the corresponding bit in the XORY 24-bit pseudo register is a one. CPL determines the length of the operand field.

4-Bit Results

The operation of the 24-bit function box generates numerous conditions which are useful as indicators of what processor action would be appropriate following the one currently in progress. These conditions are monitored by logic included for the purpose, and are available as additional outputs. These are arranged in 4-bit groups, and are known as the pseudo registers: XYST, XYCN and BICN. The 4-bit results are gated to the auxiliary 4-bit bus, and from there are available for analysis by the 4-bit function box. The 4-bit pseudo registers are defined as follows:

XYST(X-Y states): When sourced, indicates the condition of the X- and Y-registers as compared with zero, the condition of the least significant unit of the X-register, and the presence or absence of an interrupt condition within the system. Refer to figure 1-9 for details.

XYCN (X-Y conditions): When sourced, indicates the status of the X- and Y-registers as equal to, greater than, and less than each other. It also indicates the condition of the least significant bit of the X-register. Refer to figure 1-10 for details.

BICN (binary conditions): When sourced, indicates the presence or absence of a carry level or borrow level, and the state of the carry flip-flop CYF. It also indicates the condition of the least-significant unit of the Y-register. Refer to figure 1-11 for details.

4-BIT FUNCTION BOX

The 4-bit function box (4-bit arithmetic and combinatorial unit) is an auxiliary device which provides for analyzing and altering the contents of the 4-bit processor registers. These include actual logic registers, most of which are portions of larger registers, and the various 4-bit pseudo registers. The 4-bit function box is used primarily to manipulate and generate control signals within the processor, and is therefore more a programmatic tool than a means for processing job data. Operationally, it is capable of three types of actions: straight through, logic functions, and bit testing.

Straight through is used for register to register moves, and the data is unaltered in transit. It is necessary because the output of all 4-bit registers is gated to the auxiliary 4-bit bus, whereas their inputs are from the MEX or 4-bit result bus.

The logic functions involve interactions between a sourced 4-bit register and a 4-bit literal extracted from a micro. In most cases a modification of the register data takes place, and this result is returned to the source register. The functions include set, AND, OR, exclusive OR, sum, difference, sum overflow test and difference underflow test. Refer to the operation section for further details.

Bit testing is available for the purpose of generating branch and skip control levels based on the contents of a selected register. As described in the 24-bit function box section, the bits therein indicate conditions existing within the system, and are used for programmatic decision making. If the designated condition is true, the 4-bit function box causes the processor to either skip the next micro instruction in the string being executed, or to branch to another micro address and begin execution of that string.

4-Bit Function Box Operation

Operation of the 4-bit function box can be best understood by becoming familiar with the various micro instructions which utilize it. Therefore, it is recommended that reference be made to the micro operator listings elsewhere in this section while considering the following discussion.

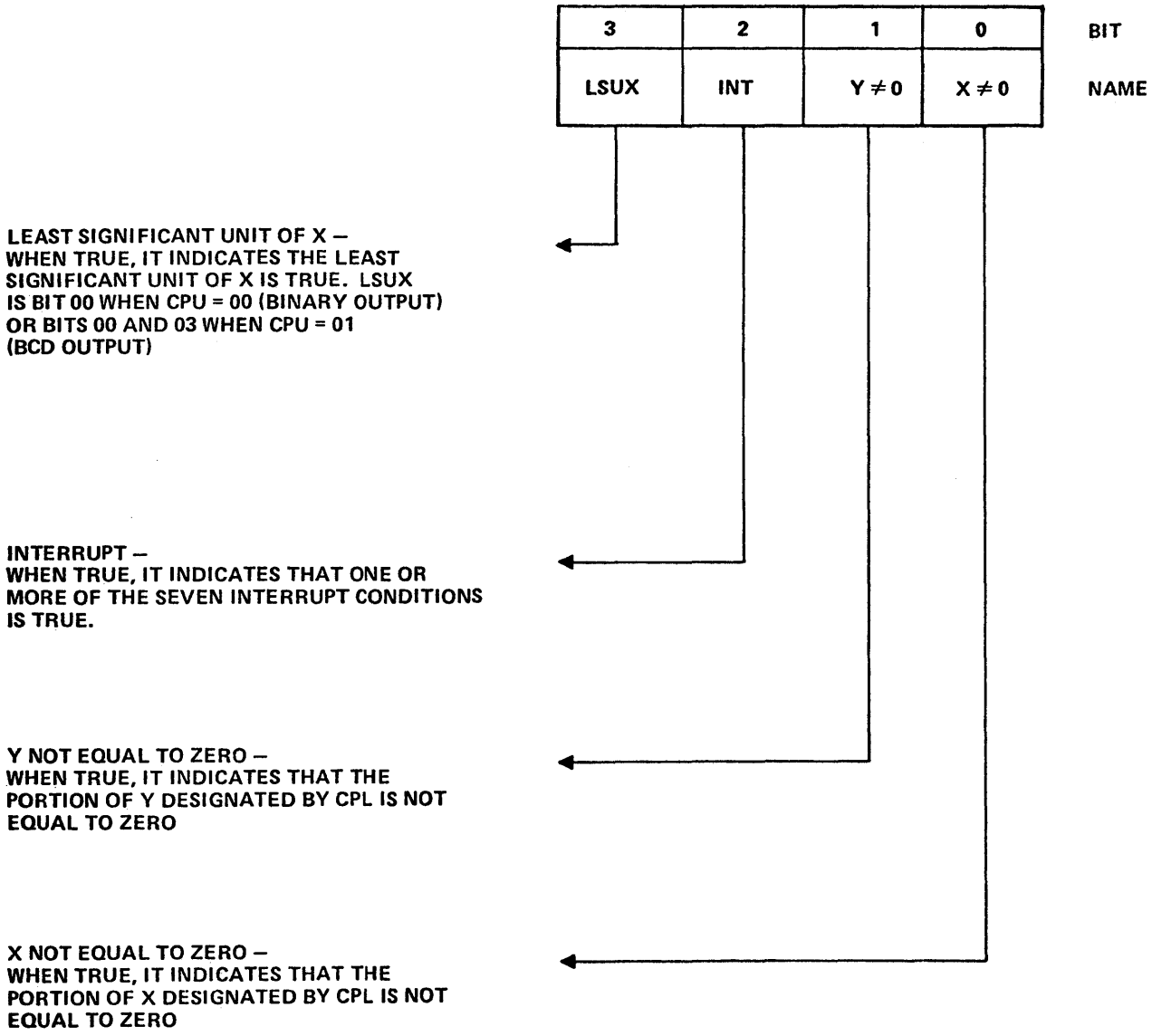


Figure 1-9. XYST 4-Bit Pseudo Register

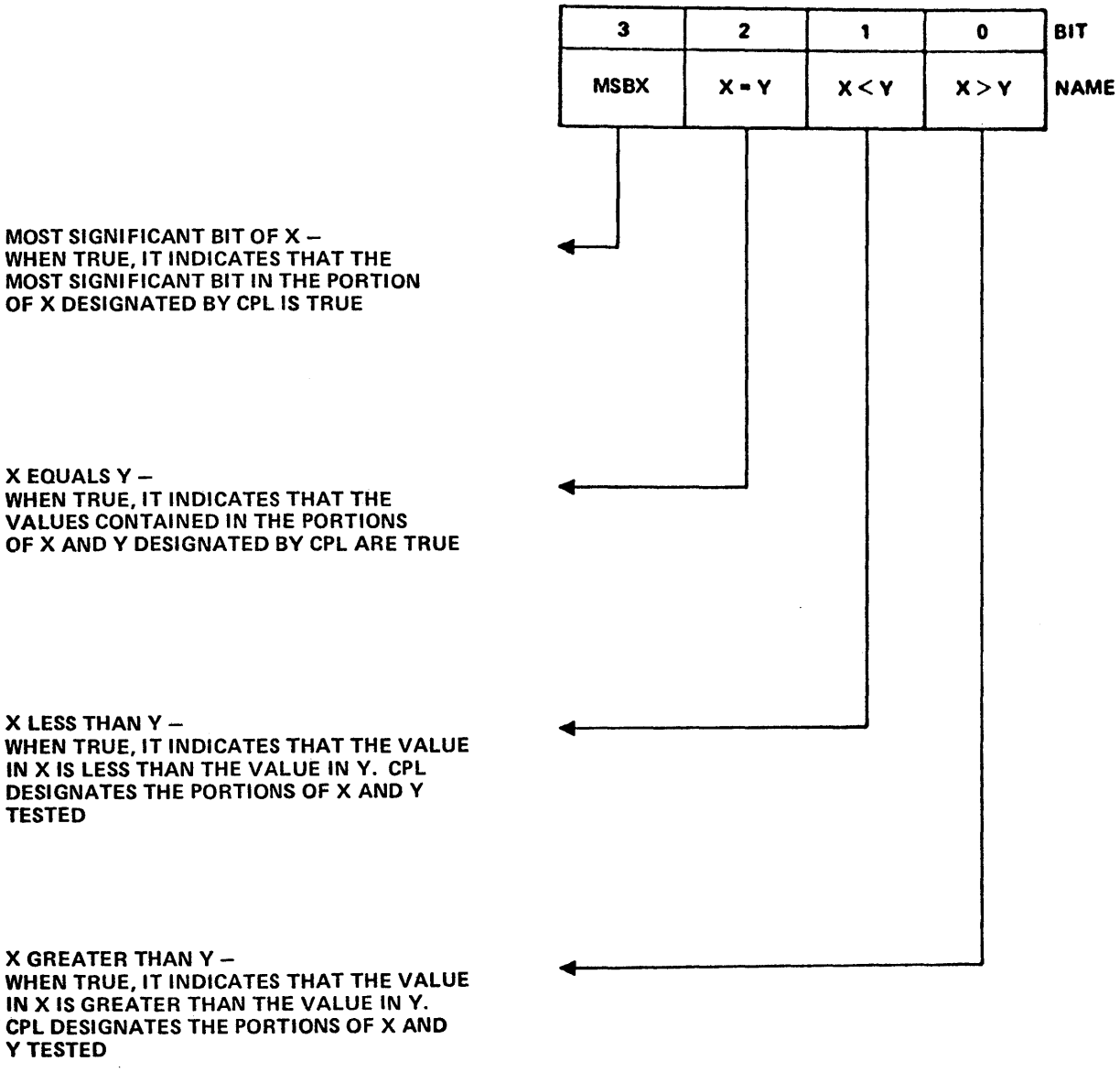


Figure 1-10. XYCN 4-Bit Pseudo Register

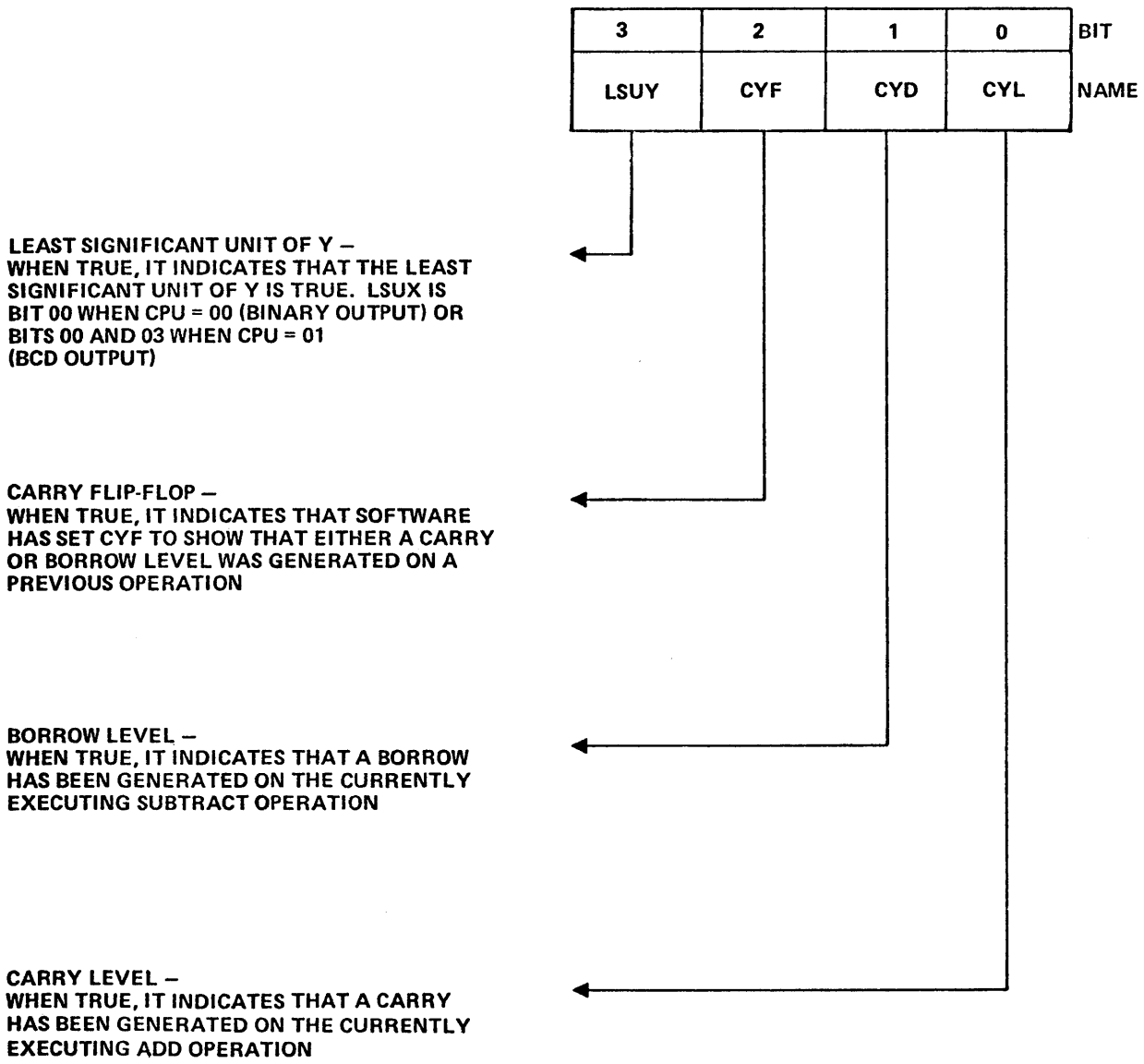


Figure 1-11. BICN 4-Bit Pseudo Register

Inputs

Any one of the following registers or pseudo registers may be sourced at any given time.

a. TA, TB, TC, TD, TF, LA, LB, LC, LD, LF, FU, FT, FLC, FLD, FLE, FLF, CA, CB, CC, CD, or TOPM.

   Each of the above is a 4-bit hardware register whose contents can be sourced for analysis, testing and altering. The results of the tests performed on these registers by various micro instructions can result in either skips or branches. Alteration results in a change in the register's contents, since results are returned to the source. This does not apply to MAXM or MAXS because these registers are hard wired to preset values.

b. CPU

   The CPU register is physically bits 5 and 6 of the 8-bit CP register. When sourced CPU is treated as a 4-bit register with the two most-significant bits always being equal to zero. CPU is addressable as either a data source or data sink.

c. XYST, XYCN, BICN, FLCN, or INCN

   Each of the above is a 4-bit pseudo register which can be sourced via the execution of various micros. The name "pseudo" means that the registers do not actually exist as such, but are grouped derivations of various control signals from within the processor logic. These registers can be sourced, tested, and analyzed by the 4-bit function box, but not altered by its actions. The XYST, XYCN, and BICN pseudo registers indicate conditions which exist primarily between the 24-bit X- and Y-registers, and were described as part of the 24-bit function box. The terms FLCN and INCN indicate independent pseudo registers, and are defined in figures 1-12 and 1-13. Also refer to the FL and SFL register descriptions in the section on registers.

4-Bit Function Box Functions

The various functions are selected by use of the appropriate micro instructions. A listing of these micro instructions, and the actions available are provided below. Note that the functions listed are those concerning the 4-bit function box only. The micro-instructions themselves can have additional variants affecting other logic.

1C (register move)

2C (scratchpad move)

Move the contents of a 4-bit register to a 4-bit or 24-bit register. No manipulation or altering of the original sourced data is provided. Input (to the 4-bit function box) is via the auxiliary 4-bit bus and output is via the least significant 4 bits of the main exchange. If a 24-bit register is selected as a sink, the four bits are moved to the least-significant four positions, and the remaining 20 bits are zero filled. Note that scratchpad may be a sink only.

3C (4-bit manipulate)

Execute the designated logical function, using the contents of the sourced 4-bit register and the least-significant four bits (literal) of the 3C micro instruction itself as operands. The functions include:

SET - Move the literal to the designated 4-bit register. This value is also placed on the 4-bit result bus.

AND - Perform the logical AND function between the 4-bit register contents and the literal. (Both corresponding bits must be true for a true result bit.) The result is placed on the 4-bit result exchange.

OR - Perform the logical OR function between the 4-bit register contents and literal (either or both corresponding bits being true produces a true result bit). The result is placed on the 4-bit result bus.

EOR - Perform the logical exclusive-Or function between the 4-bit register contents and the literal; (either one, but not both, of the corresponding bits being true produces a true result bit). The result is placed on the 4-bit result bus.

INC - Binarily add the 4-bit register contents and the literal. The result is placed on the 4-bit result bus. Overflow (carry) is not detected.

INC and TEST - Binarily add the 4-bit register contents to the literal and monitor the result for overflow. Overflow causes the skip function to be initiated. The arithmetic result is not used.

DEC - Binarily subtract the literal from the contents of the 4-bit register. The result is placed on the 4-bit result bus. Underflow (borrow) is not detected.

DEC and TEST - Binarily subtract the literal from the contents of the 4-bit register and monitor the result for underflow. Underflow (borrow) causes the skip function to be initiated. The arithmetic result is not used.

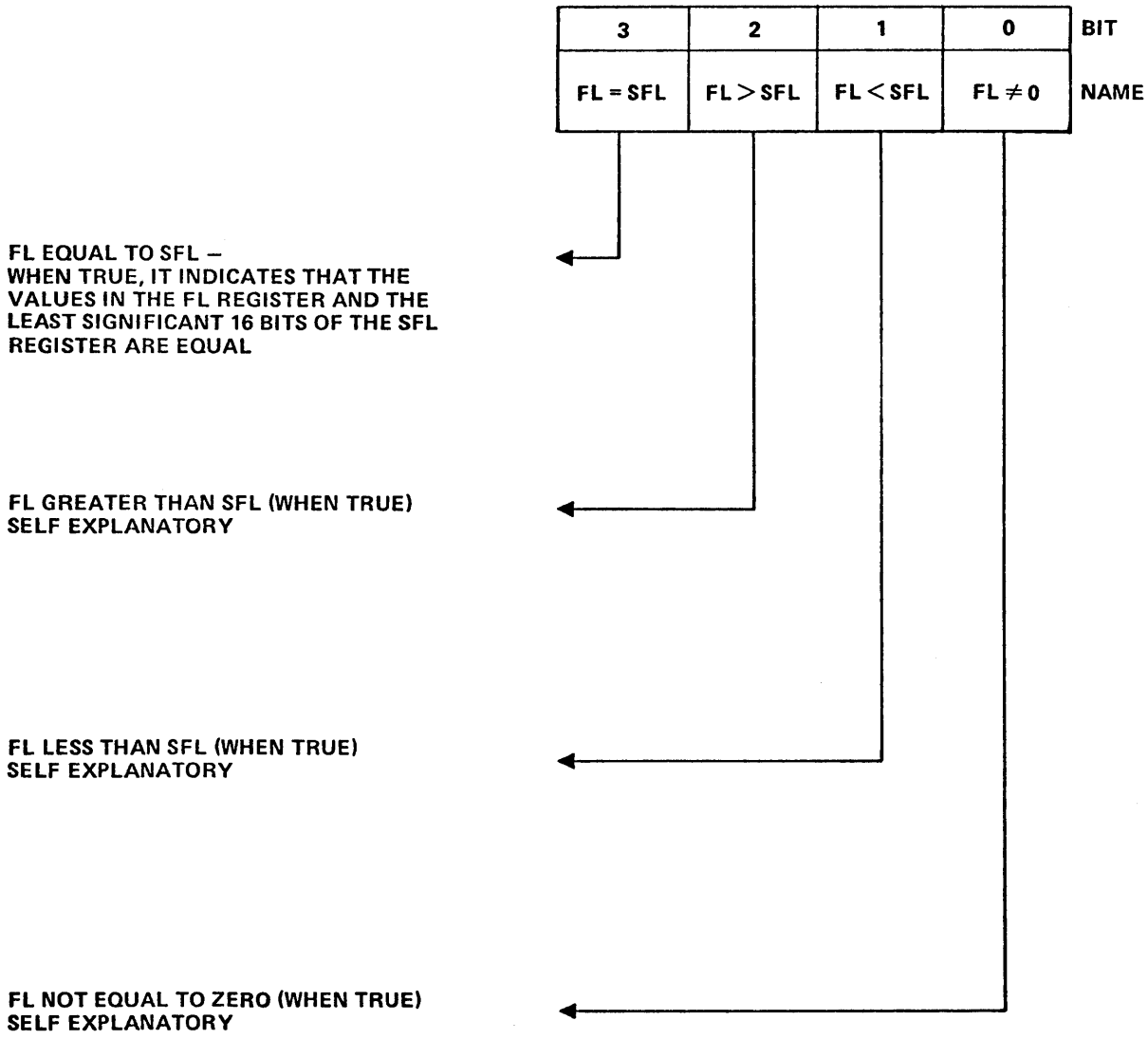
4C (bit test relative branch false) and

5C (bit test relative branch true)

Test the specified bit of the selected 4-bit register, and branch on the results of of the test. The branch is taken if the tested bit is 0 for a 4C micro or if the bit is 1 for a 5C micro. The relative displacement of the branch (number of micros skipped) is contained in the literal, and may be a maximum of 15 instructions for a positive branch or 13 for a negative branch. Negative branching results in looping back to a micro instruction in lower sequential order. No output is provided, as the source data is not altered.

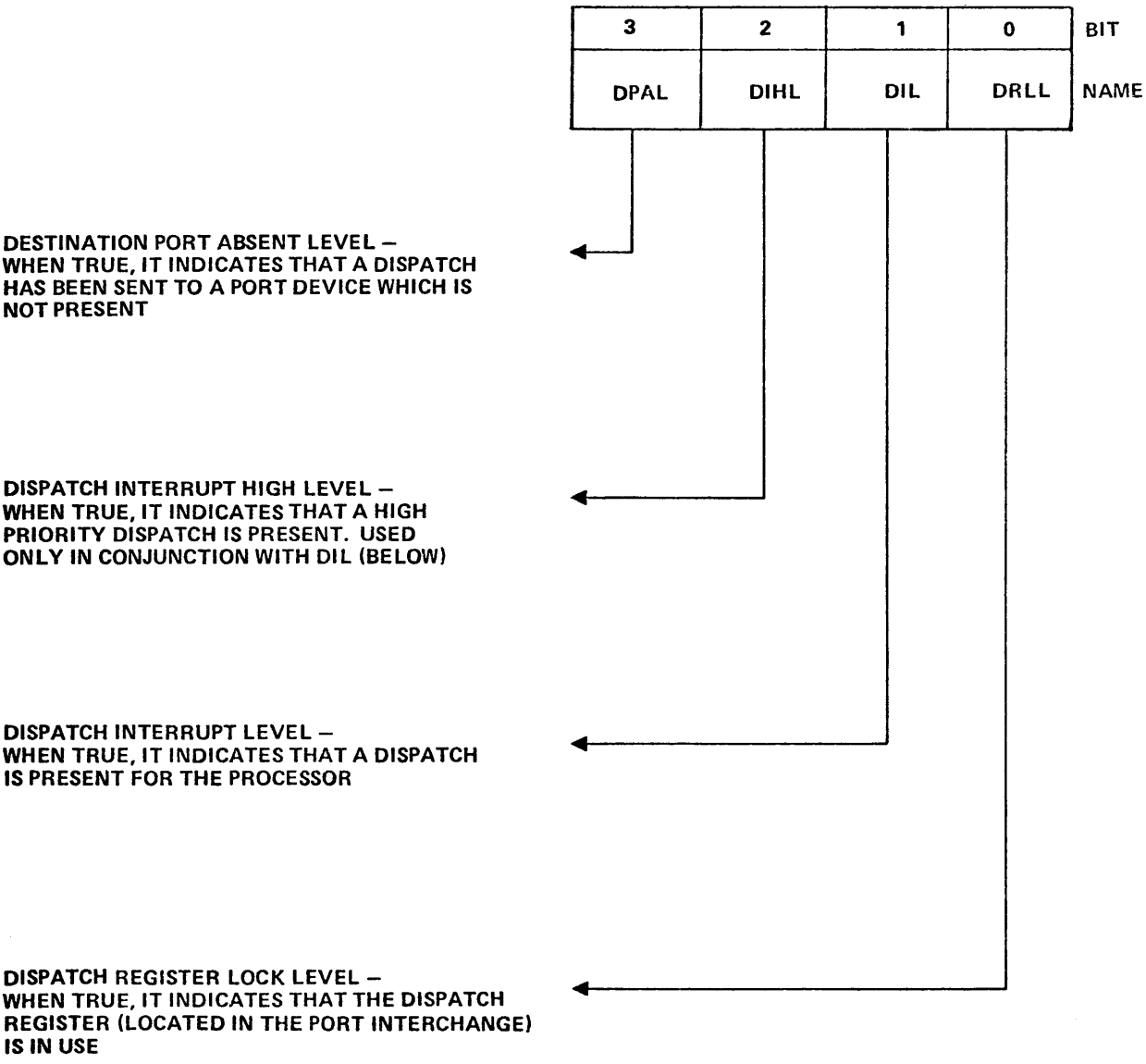
6C (skip when)

Test the specified bits in the selected 4-bit register as determined by the variants contained in the micro. Perform the action specified, based on the results of the test. Refer to figure 1-14 for the possible tests. The bits to be tested are specified by the literal contained in the 6C micro, which is known as the mask. Only the bits in the 4-bit register which are referenced by 1-bits in the mask are tested and also cleared if specified. All other bits are ignored except when "register equal to mask" comparisons are made. Since there are four test bits and three variant bits, there are 128 ($2^4 \times 2^3$) possible combinations of tests. The skip function, when executed, causes the processor to skip the next in line micro instruction. Output is via the 4-bit result bus. This is used only to clear the sourced register when the micro variants designate that it be cleared.



NOTE: FL IS THE LEAST SIGNIFICANT 16 BITS OF THE FB REGISTER
SFL IS A DISCRETE 24-BIT REGISTER WHICH ALWAYS CONTAINS THE SAME VALUE AS THE FIRST WORD (WORD 0) OF RIGHT SCRATCHPAD. ONLY BITS 00 THRU 15 (OF THE 24) ARE CONSIDERED FOR THIS FUNCTION.

Figure 1-12. FLCN (Field Length Conditions) 4-Bit Pseudo Register



NOTE: ALL INCN BITS REFLECT CONDITIONS EXISTING IN THE DISPATCH REGISTER, WHICH IS LOCATED IN THE PORT INTERCHANGE. REFER TO THE PORT INTERCHANGE SECTION FOR FURTHER DETAILS.

Figure 1-13. INCN (Interrupt Conditions) 4-Bit Pseudo Register

Variants			Test or Function
0	0	0	SKIP IF ANY REFERENCED BIT IS TRUE
0	0	1	SKIP IF ALL REFERENCED BITS ARE TRUE
0	1	0	SKIP IF SELECTED REGISTER IS EQUAL TO THE MASK
0	1	1	SKIP AND CLEAR SELECTED REGISTER IF ALL REFERENCE BITS ARE TRUE.
1	0	0	SKIP IF ALL REFERENCED BITS ARE FALSE
1	0	1	SKIP IF ANY REFERENCED BIT IS FALSE
1	1	0	SKIP UNLESS THE SELECTED REGISTER IS EQUAL TO THE MASK.
1	1	1	SKIP AND CLEAR THE SELECTED REGISTER IF ALL REFERENCED BITS ARE FALSE.

Figure 1-14. 6C Micro Functions

4-Bit Function Box Outputs

The outputs of the 4-bit function box consist of two control levels (branch and skip), and two 4-bit data results. These latter are identical, but are gated to either the MEX (four least-significant bits) or the 4-bit result exchange, in order that different destinations may be reached.

REGISTERS

A large portion of the processor logic consists of registers, the purpose of which is temporary storage of data and control levels. Since all registers are basically alike, the significance of an individual register lies more in its relationship to other portions of the system logic than in its manner of functioning. Although a number of processor registers are for general purpose use, the majority are reserved for a specific function or group of functions.

Register Selection

The various registers of the B 1720 Series System are addressed from the front console by a pair of coordinates relating to the levels and positions on the register selection switch. The numerical values assigned to these coordinates are significant in that they are also used within the micro operators (in binary notation) for selection of the registers via micro decoding.

The first coordinate is to select 1 of 16 groups of registers. The second coordinate then selects one of the four registers in that group. Table 1-1 illustrates the selection of registers with the REGISTER select and register group switches. For example, the FB register is register group 9 and REGISTER select 2.

Note that not all of the entries in table 1-1 are registers. BICN, FLCN, XYCN, XYST, and INCN are pseudo registers, and were described as part of the 4- and 24-bit function boxes. Likewise, SUM, CMPX, CMPY, XANY, XEOY, MSKX, MSKY, XORY, and DIFF are outputs of the 24-bit function box, and were described previously. The above functions are treated as registers because they may be sourced via micro decoding in the same manner as actual hardware registers.

NULL Register

This is a 24-bit pseudo register which can be designated as either a source or sink. When addressed as a source, the source data is always zeroes. When addressed as a sink, the source data is gated to "nowhere." Since sourcing a register does not affect the data stored in it, gating to Null is, in effect, a no-operation (NO-OP).

In reality, null can be considered the contents of the main 24-bit exchange. Since the MEX contents has different significances at different times, both the run and halt states of the processor must be considered to understand the effect of sourcing or sinking null.

When the processor is running in either the run, step, or tape mode, the following occurs. Note that the processor is running only while executing a single micro when in the step mode. Such execution occurs when the START button is pressed.

a. If null is designated as the sink, the source register is gated to the main 24-bit exchange only. No registers are changed, including the source register. If TAS is moved to null, the A-stack pointer is decremented by one.

b. If null is sourced, zeroes are moved from the main exchange to the destination register, effectively clearing it. If null is moved to TAS, TAS is cleared and the A-stack pointer is incremented by one.

c. A special condition exists when the console register selection switches are set to NULL and the processor is running. In this condition, the processor halts when the A-register equals the value set on the 24 console switches.

When the processor is halted, the following occurs:

a. A hardware-forced 1C micro (move register to null) is constantly executing. The data on the main exchange is the contents of the register designated by the console register selection switches. Since the console lamps reflect the contents of the main exchange, the contents of the register are effectively displayed.

b. When the LOAD button is pressed, the hardware-forced 1C micro is changed to "move null to register." This causes the contents of the 24 console switches to be gated to the main exchange as the sourced null data. This data is gated to the register designated by the console register selection switches.

CMND Register

The CMND (command) register is a 24-bit pseudo register which is used exclusively for soft I/O operations. It is effectively the 24-bit processor input/output bus, and is used exclusively to transfer data to the I/O subsystem. Therefore, CMND may be designated as a sink only. The data transferred to the I/O when CMND is sink may be either command type, command variants, channel numbers, op-codes, reference addresses, file addresses, or write data.

When CMND is designated as sink, a command active level (CA) is also sent to the I/O subsystem. The CA level signals the I/O devices that information is present on the I/O bus for one or more of them, and indicates the first part of the two part I/O cycle.

DATA Register

The DATA register is a 24-bit pseudo register which is also used exclusively for soft I/O operations. The data register is also effectively the 24-bit processor input/output bus, but is used in a different manner than CMND. This register is bi-directional, and may serve as either a source or sink for transferring information to or from the I/O subsystem.

When the DATA register is sourced, the data transferred from the I/O may be either status counts, service request mask bits, control identification numbers, special flags for certain devices, reference addresses, result descriptors, or read data.

When the data register is designated a sink, write data is transferred from the processor to the I/O.

When the DATA register is either sourced or sinked, a response complete level (RC) is also sent to the I/O subsystem, signalling that the second part of a two-part I/O cycle is complete.

READ Register

READ is a 24-bit pseudo register which is addressable as a source only. It effectively serves as the source of read data on S-memory read operations. Sourcing the READ register causes the data at the S-memory address designated by the value in FA to be gated to the MEX for distribution to the sink register selected.

With the machine halted and the READ register selected on the console register selection switches, pressing the READ/WRITE pushbutton causes an S-memory read cycle to be initiated. The memory data is latched into the console lamp register and from there displayed on the console lamps. A freeze of this data on the console lamps occurs. To remove it, the selection switches must be moved to another register or another memory read cycle must be initiated.

The FA register is incremented by 16 bits (one word address in S-memory) each time the INC pushbutton is pressed. Note that before pressing the READ/WRITE pushbutton a machine 1C micro (move READ to null) is executed. Since memory has not yet been accessed, this results in all zeroes being displayed on the console lamps. If the LOAD button is pressed, the 1C micro changes to "move null to READ." Zeroes are still displayed on the console lamps. This is a result of the hardware configuration, and serves no useful function in the halt mode.

When the machine is running, sourcing the READ register causes the contents of the 24 console switches to be gated to the MEX for distribution to the selected sink. This is a function of the same hardware configuration, and is available as a programmatic tool which allows manual insertion of data from the console while the machine is running.

WRIT Register

The WRIT (write) register is a 24-bit pseudo register which serves as a sink only, and is usable only in the halt mode. With the machine halted and WRIT selected on the console register selection switches, pressing the READ/WRITE pushbutton causes an S-memory write cycle to be initiated. The contents of the 24 console switches are gated to the MEX, and from there are written into S-memory. The address in the FA register designates where in memory the write cycle occurs. The FA register may be incremented by 16 bits (one word address) by pressing the INC pushbutton. Note that the data written in memory is not displayed on the console lamps.

As with the READ register, a machine-forced 1C micro (move WRIT to null) is executed when WRIT is selected, and before the READ/WRITE button is pushed. This results in zeroes being displayed on the 24 console lamps. Pressing LOAD changes the micro to "move NULL to WRIT", which also causes zeroes to be displayed. Neither condition is significant to machine operation.

MSM Register

The MSM register is a 16-bit pseudo register used for accessing M-string memory. It may be used as either a source or sink, and such use results in reading from or writing in M-string memory. The address to which access is gained is determined by the contents of the A-register. Such access is to a single word of MSM.

With the machine halted and MSM selected on the console register selection switches, a machine-forced 1C micro (move MSM to null) is continuously executed. The contents of the word of M-string memory addressed by A is displayed on the console lamps. Pressing the LOAD pushbutton changes the micro to "move null to MSM," causing the contents of the console switches to be gated to the MEX, and from there written into the word of M-string memory designated by A. Note that only the least-significant 16 console switches are active, as one word of M-string memory is 16 bits in length. Pressing the INC button when MSM is selected increments the A-register by 1, effectively pointing to the next higher word address in M-string memory.

Sourcing or sinking MSM when the machine is running produces the same read/write functions as when halted, except that the console controls are not involved. The MSM is utilized by the 1C, 2C, 7E, 8C, 9C, and 10C micros. When executing a move from MSM to a 24-bit register, the data is right-justified with zero fill in the eight most-significant bit positions. Sourcing a 24-bit register with MSM as sink results in truncation of the 8 most-significant bits.

T-Register (With Rotator and Mask Generator)

The T-register is a general purpose 24-bit register which may be addressed as a unit, or as six 4-bit registers denoted TA, TB, TC, TD, TE, and TF. In processor operations it serves as both a source and a sink. The T inputs and outputs are:

24-bit operations: to/from MEX.

4-bit operations: from auxiliary 4-bit exchange or MEX.

to 4-bit result exchange or MEX.

In addition, a direct path is provided between the seven least-significant bits of T and the same bits in the dispatch register. These bits are used as both a source and sink in dispatch operations. The T-register is illustrated in figure 1-15.

TA	TB	TC	TD	TE	TF	
23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0	BIT

Figure 1-15. T-Register

Associated with the T-register is the T-rotator. This device is located on the output side (between T and the MEX), and provides optional shifting or rotation of the data contained in T when it is sourced. Shifting and rotation are defined as follows:

Rotation: simultaneous lateral movement of all data contained within a specified field, with wrap around provided (data departs the field at one end and simultaneously reappears at the opposite end).

Shifting: simultaneous lateral movement of all data contained within a specified field, without wrap around (data departing the field is lost; zero fill occurs at the opposite end).

Shifting and rotation are primarily utilized for analysis of I/O service requests, and for binary multiplication and division by powers of 2. Both are accomplished with the 10C micro.

The T-rotator consists of a 24-bit two stage multiplexing device and a 24-bit mask generator. The first stage provides left rotation of the T-register data by zero, eight, or sixteen bits.

The second stage provides left rotation of from zero to seven bits. The combination of both stages allows a total rotation capability of from zero to twenty-three bits. Note that while the rotator functionally operates in the left direction only, effective right rotation may be accomplished by revolution (left rotation by (24-N) bit positions, where N is the desired magnitude of right rotation). Rotation is illustrated in figure 1-16.

The shift function is accomplished in the same manner as rotation, except that masking is utilized. That is to say the rotator functions identically, but those bits which have wrapped around are masked. The mask generator is in actuality an enabling circuit, whereby the presence of a mask 1-bit enables gating the corresponding data bit to the output. Therefore, those bit positions where the enabling bit is absent are effectively masked, producing a zero output. For a shift left operation, masking (zero fill) occurs from the right. For a shift right, a zero fill is from the left. Note that masking occurs on shift operations only (for rotation 24 mask 1-bits are generated, causing the entire rotator output to be gated to the MEX). The shift function is illustrated in figure 1-17.

The rotator is capable of one additional function, that being the operation specified by the 11C micro (extract from T-register). Extraction is similar to the shift function, except that the amount of rotation and the width of the mask are independent. Rotation is performed so that the desired data is right-justified with respect to the MEX. The mask is then applied as determined by the literal contained in the 11C micro. Zero fill is always from the left. Refer to figure 1-18.

EXAMPLE 1 – ROTATE LEFT BY 5 BITS

T-REGISTER CONTENTS

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	CONTENTS

OUTPUT TO MEX

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	CONTENTS

EXAMPLE 2 – ROTATE RIGHT BY 11 BITS

T-REGISTER CONTENTS

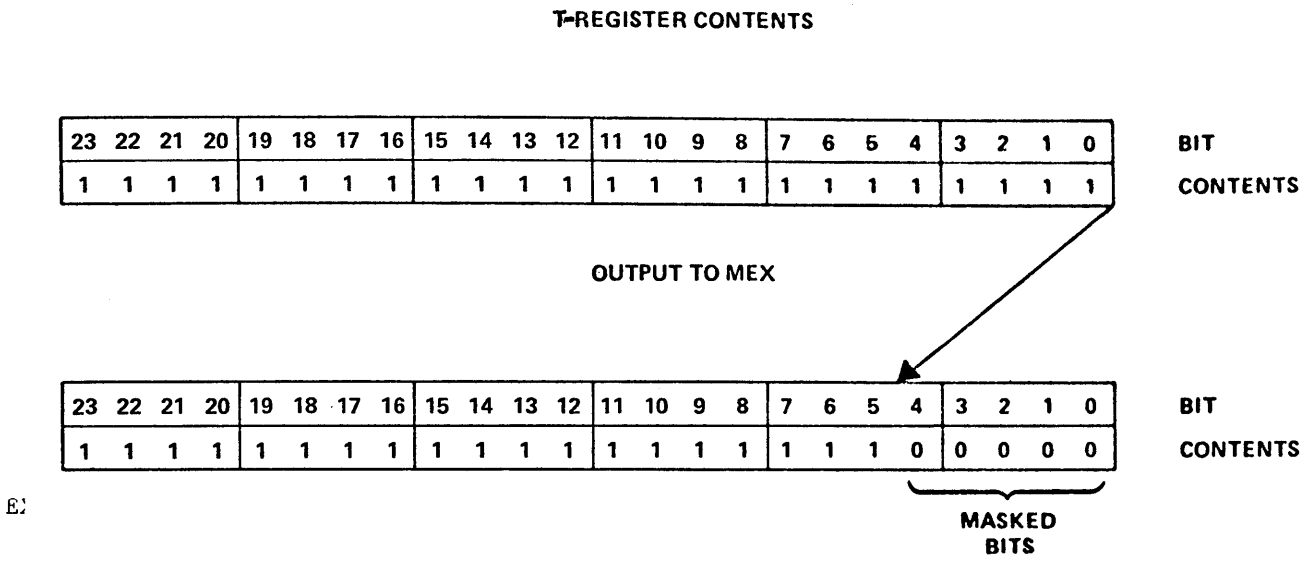
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	CONTENTS

OUTPUT TO MEX

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	1	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	CONTENTS

Figure 1-16. Rotation Function Of T-Register

EXAMPLE 1 – SHIFT LEFT BY 5 BITS



EXAMPLE 2 – SHIFT RIGHT BY 7 BITS

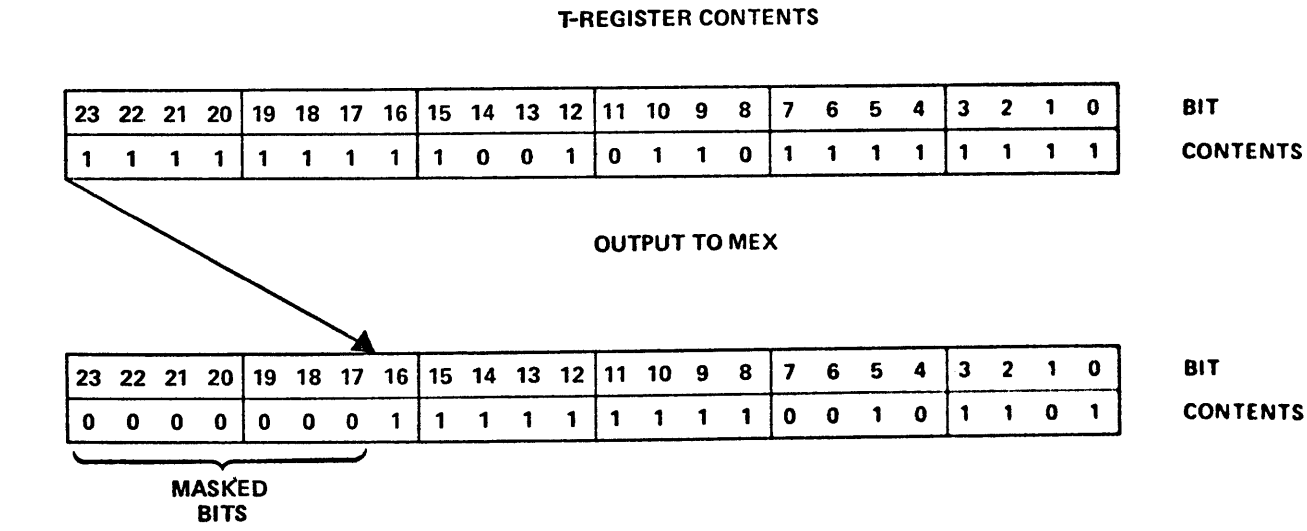


Figure 1-17. Shift Function Of T-Register

EXAMPLE: EXTRACT FROM T THE DATA FIELD CONSISTING OF BITS 8 – 12.
THIS INVOLVES RIGHT ROTATION BY 8 BITS, AND MASKING OF THE 19 MOST SIGNIFICANT ROTATOR OUTPUT BITS.

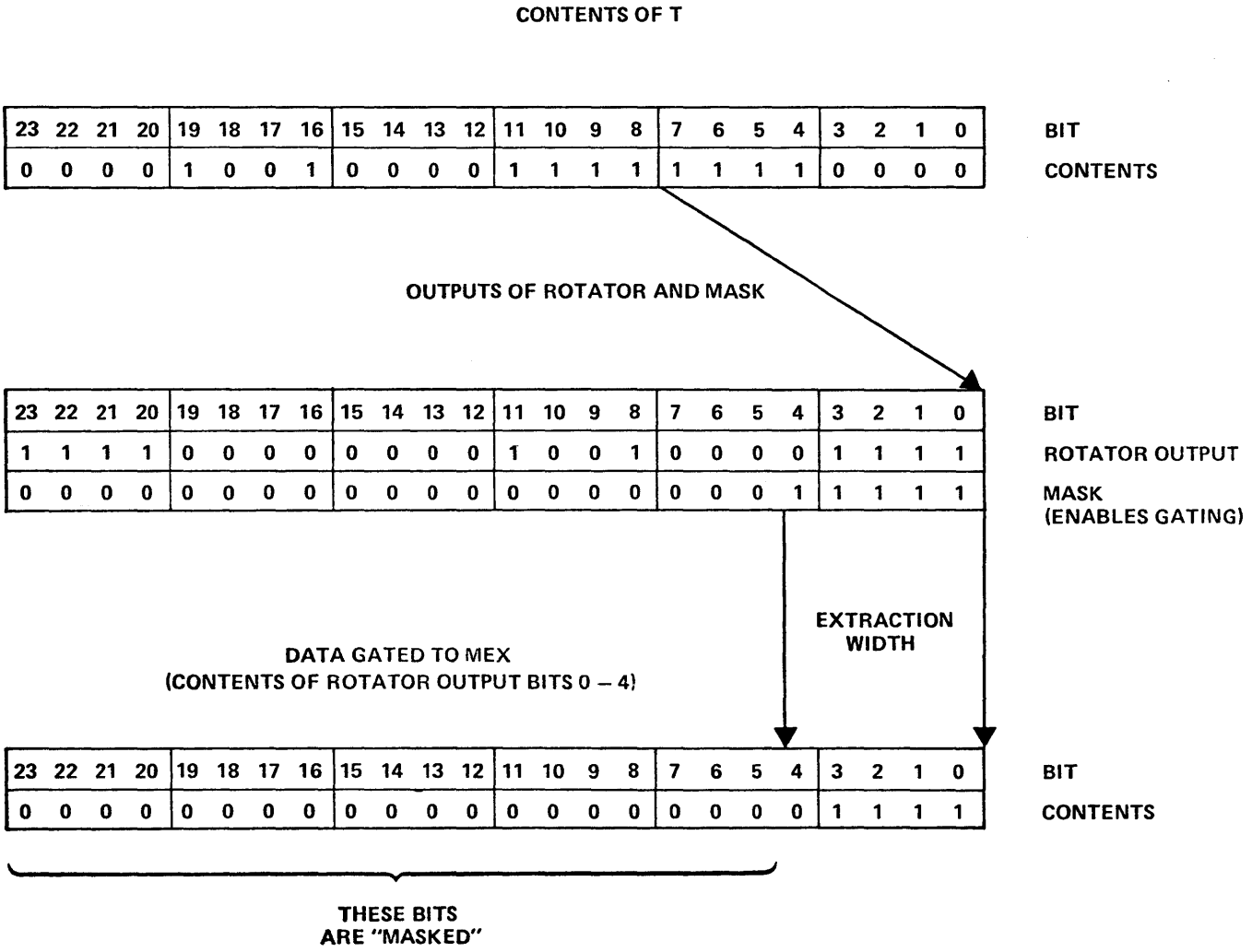


Figure 1-18. Extract From T-Function

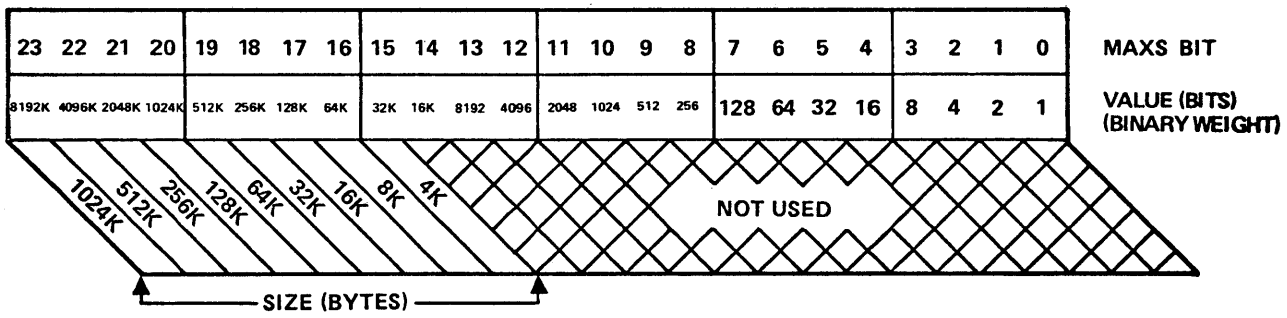
A-Stack

The A stack is a 32-word deep, 24-bit wide memory. It is generally used for storage of addresses in normal call/return programming where it is desired to perform a subroutine, and then return control to the program when the subroutine is completed. The procedure normally used is to call a subroutine by branching to the address where it is stored in memory. Meanwhile, the address where the program exited is stored in A-stack. Return from the subroutine is accomplished by moving the address stored in A-stack back to the A-register, which acts as a branch back into the normal program.
The A-stack operates as a push down stack with a last in/first out structure. The A-stack is addressed by a stack pointer which is upcounted or downcounted by moves into or out of the stack. Note that once written into A-stack, the location of a given data field does not change, but rather the word addressed by the stack pointer. Wrap around of the pointer is provided. Sixteen consecutive writes (pushes) into the A-stack or sixteen consecutive reads (pop) from the A-stack causes the pointer to address the same location as it did at the beginning of the write or read operations. Although A-stack is not intended to be used as an operand stack (data storage) it has been purposely made 24 bits wide to allow for limited operand storage, at the option of the programmer.

Top of A-stack (TAS) is the location in A stack which is addressed by the stack pointer. As such, it is a relative position, and its actual location within the stack at any given time is dependent upon previous usage. Top of A-stack is, in effect, a 24-bit register which may be either sourced or sinked, and serves as the sole path to or from A-stack. Moves into TAS first cause the A-stack pointer to be upcounted, and then the source data is stored in the location addressed by the pointer. Moves from TAS cause the data in the A-stack location which is addressed by the stack pointer to be read and moved to the destination register. After the read occurs the stack pointer is decremented, addressing a new A-stack location which becomes the new TAS.

MAXS Register (Maximum S-Memory Register)

The MAXS register is a 24-bit register which is set by the field engineer to indicate the maximum size of the installed S-memory (in bits). The MAXS register is not a discrete register, but rather is wired constant and thus addressable as a source only. The value in MAXS is binarily weighted. Refer to figure 1-19.

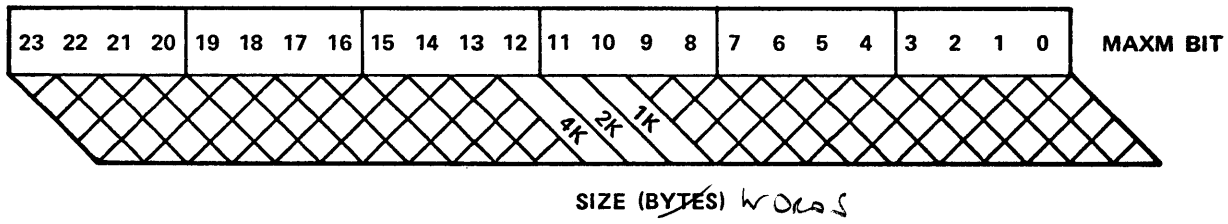


NOTE: MEMORY MAY BE EXPANDED IN 8K (8192) BYTE INCREMENTS 8K IS THE MINIMUM POSSIBLE MEMORY SIZE, HOWEVER THE MINIMUM PRACTICAL SIZE IS APPROXIMATELY 48K THE MAXIMUM PERMISSIBLE SIZE IS 256K

Figure 1-19. MAXS Register

MAXM Register (Maximum M-Memory Register)

The MAXM register is a 24-bit register which is set by the field engineer to indicate the maximum size of the installed M-memory. Like MAXS, it is a hard wired register, and is therefore addressable as a source only. The value in MAXM is binarily weighted. Refer to figure 1-20.



NOTE: M-MEMORY SIZE IS LIMITED TO 4K (4096) BYTES. words
THE MINIMUM POSSIBLE SIZE IS 1K (1024) BYTES. words

Figure 1-20. MAXM Register

A-Register

The A-register is the address register for the M-string memory, and is 14 bits wide. The contents of A constitute a word address for a 16-bit micro instruction, with a 1-bit increment of A being equal to a 16-bit address increment in MSM. The A register may be addressed as either a source or a sink. Because the least-significant bit of A is equal to a word address (binary weight of 16), A is displaced by four bit positions with respect to the main exchange. Refer to figure 1-21. When used as a source, the contents of A are effectively multiplied by 16 (least-significant bit of A, bit 00, is gated to the fifth bit, bit 04 of the MEX). The other bits of A are gated to the corresponding higher bits of the MEX. When the A-register is used as a sink, the least-significant four bits of the source are lost. This effectively divides the source by 16.

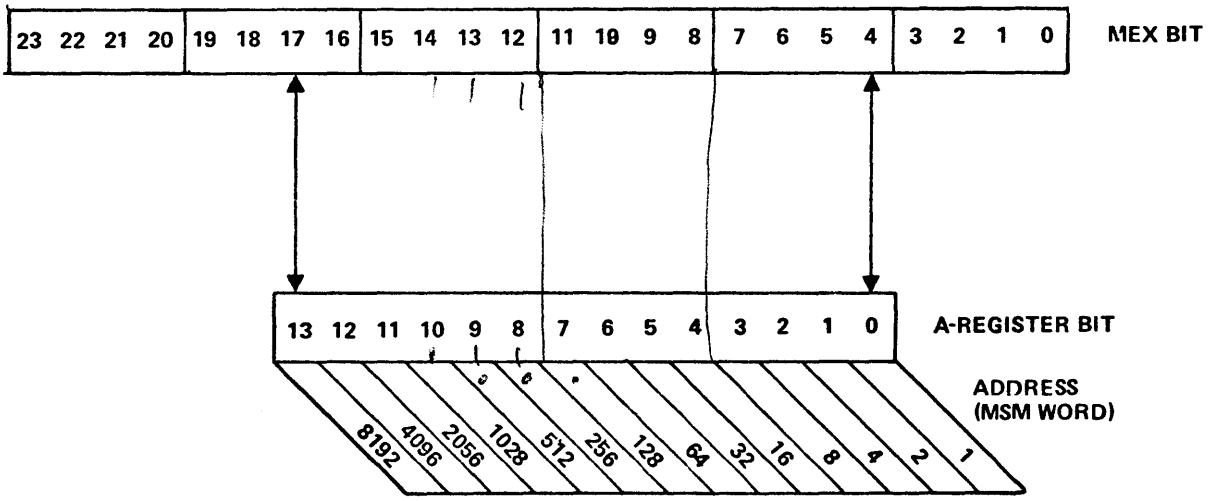


Figure 1-21. A-Register

When the machine is in the run mode, the A-register is automatically incremented by one each time a micro is gated from MSM to the M-register for execution. This is the most important feature of the micro fetch routine, by which the processor accesses and executes sequential strings of micros stored in M-string memory. The A-register may also be incremented or decremented by any value from 0 to 4095 with a high-speed adder. This facilitates micro program branching. Wrap around of A can occur, and is permitted at the largest possible address (4095 words). It should be noted that the maximum size of M-memory is 4096 16-bit words. Therefore, A-register values greater than 4095 point to addresses outside of M-string memory. Provisions have been made within the processor to permit the use of a portion of S-memory as an extension of M-string memory. This involves the use of the TOPM and MB registers. Refer to the discussion of TOPM and MBR for further information.

TOPM Register

The TOPM register (top of M-memory) is a 4-bit register which is associated with the A-register. It is used to determine the highest allowable address in MSM. When the value in the A-register is equal to or greater than the value in TOPM, micros are fetched from S-memory rather than M-memory. The TOPM register may be used as either a source or sink. It is displaced with respect to the MEX by 13 bits, which means that the most-significant bit of TOPM has a binary weight of 65,536(64K). The TOPM register is set to a value of 1000 by the system clear signal. This value corresponds to 4096 words, which is the maximum number of micro instruction addresses (locations) in M-string memory. It may also be set to 4096 or any lesser value which is divisible by 512 (binary weight of the least-significant bit) by addressing it as a sink. The top of memory register is customarily set to the actual size of M-memory in the system. The bind micro (4F) designates TOPM as sink. Refer to figure 1-22.

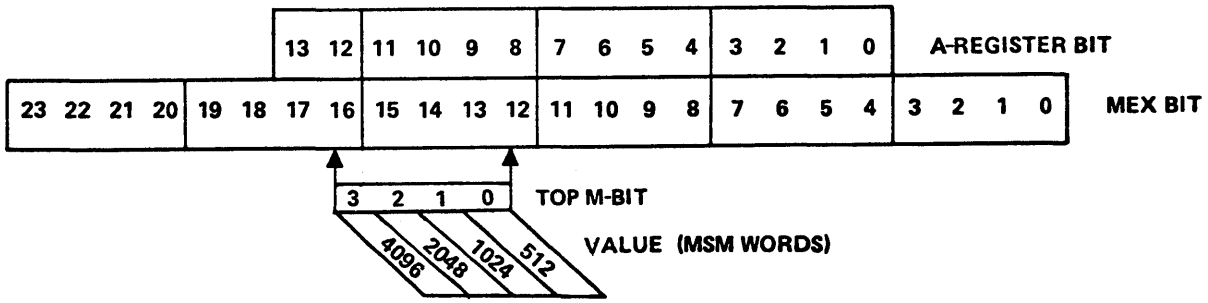


Figure 1-22. TOPM Register

MBR (Memory Base Register)

The memory base register is a 24-bit register which is used when micro instructions are sourced from S-memory. This occurs when the address in A equals or exceeds the value in TOPM, which is known as the "A out of bounds condition" (AOB). This condition causes the contents of A to be added to MBR as shown in figure 1-23. The resulting value becomes the S-memory address from which the next micro instruction is to be fetched. Note that A continues to increment each time a micro is executed in M, resulting in accessing sequential 16-bit fields from S-memory in the same manner as micros are fetched from M-memory. When AOB is true, machine operation slows appreciably, since S-memory accesses take considerably longer than fetches from M-memory.

The MBR is addressable as either a source or sink. The address contained therein is a bit address, as contrasted with that in A, which is a word address. Therefore, A is displaced by four bits (with respect to MBR) when they are added.

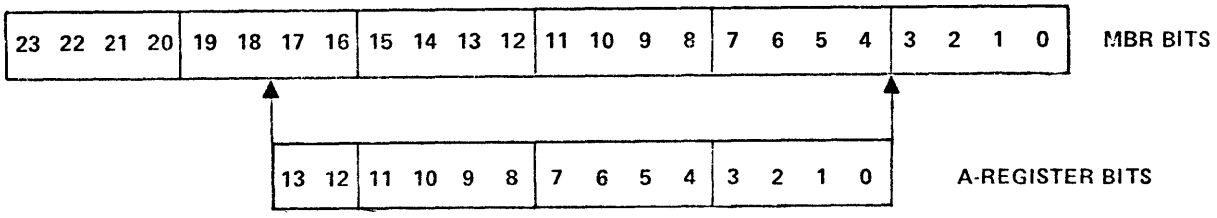


Figure 1-23. Memory Base Register

BR and LR Registers (Base and Limit Registers)

The base and limit registers are both 24 bits in length, and can be addressed (separately) as either source or sink. In operation both are used to store reference memory addresses which are used programmatically for protection of selected portions of S-memory, and for base relative addressing. A compare of the addresses in both BR and LR is made for all memory cycles (read, write, and swap). Customarily, the portion of memory between BR and LR is allotted for general usage. Those portions lying below BR and above LR are protected (bad FA addresses), and reserved for specific data. Read memory cycles are allowed regardless of the relationship existing between the addresses in FA, BR, and LR. However, if a "bad FA address" read occurs, this event is recorded by setting bit 1 of the CD register. Handling of this flag is determined by software.

S-memory write or swap cycles are not allowed to a "bad FA address" unless overridden. This is accomplished by setting the "override bit", bit 02 of the CD register. When a bad FA address write is allowed, this event is recorded by setting CD register bit 00. Handling of both bits (CD0 and CD2) is determined by software. Note that FA addresses equal to either BR

or LR are considered good and are allowed. Base relative addressing is accomplished by adding the value in FA to that in BR.

X-Register

The X-register is a 24-bit general purpose register which can serve as either a source or sink. It is used primarily to store one of the operands for 24-bit function box operations. The X-register may be shifted or rotated, and may also be normalized (shifted left in 1-bit increments until either the most-significant bit referenced by CPL equals 1 or until FL equals 0). In addition, X may be concatenated with the Y-register to effectively form a 48-bit register. When this is done, X occupies the 24 most-significant bit positions. Note that X may be addressed in its entirety only.

Y-Register

The Y-register is a 24-bit general purpose register which serves the same basic functions as the X-register. Like X, it may be used as either a source or destination, and is addressable in its entirety only. The Y register may not be normalized, but is capable of being shifted or rotated. The most common use of Y is to store one of the two operands for 24-bit function box operations (the other is stored in X).

L-Register

The L-register (logical register) is a general purpose 24-bit register which can serve as either a source or sink. It is addressable as a whole, or in 4-bit segments designated LA, LB, LC, LD, LE, and LF. Accordingly, these 4-bit registers are available for alteration and/or analysis by the 4-bit function box. Inputs to L are from either the MEX or 4-bit result exchange, with outputs to the MEX or 4-bit auxiliary exchange. Refer to figure 1-24. The L-register is one of four data register (X, Y, T, and L) which can be used as a source or sink for S-memory read/write operations.

LA	LB	LC	LD	LE	LF	
23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0	BIT

Figure 1-24. L-Register

The dispatch micro (1E) uses the L-register as the source or sink of a 24-bit message (usually an address) which is either written into/read out of S-memory at address zero.

The overlay M-string micro (2F) uses L as the source of a starting M-memory address to be used in the overlay operation.

The bind micro (4F) uses the L-register as the source of the 24-bit value to be moved to the MB register.

The exercise MSM micro (7E) uses the L-register as the source of the M-memory address to be used for the operation.

FA Register

The FA (field address) register is a 24-bit register which is used primarily to hold the address currently being accessed in S-memory. The address contained in FA is an absolute bit address, i.e., any location within S-memory may be accessed, beginning at any desired bit.

The significance of the various bit positions of FA is shown in figure 1-25. The FA register may be used as either a source or sink, and, in addition, may be incremented or decremented by the value contained in a micro instruction or by the value of the CPL register. Inputs and outputs of FA are the MEX and left scratchpad. Also associated with FA is a 24-bit FA adder which provides the increment/decrement function. Neither overflow nor underflow of FA is detected. Therefore, the value of FA may go through the register's maximum or minimum bit value and wrap around.

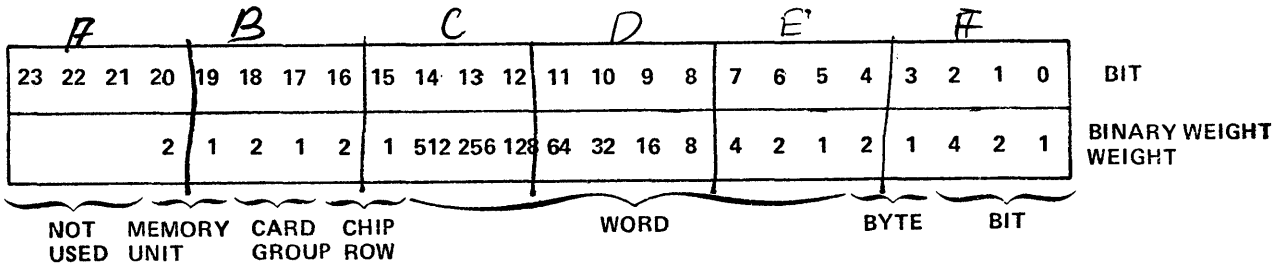


Figure 1-25. FA Register

The scratchpad relate FA micro (8D) provides increments/decrements of FA by the value of a left scratchpad word.

The exchange double pad word micro (7D) provides swapping the contents of both FA and FB with the contents of a double scratchpad word (both left and right scratchpad words of the same address.)

The load F from double pad word micro (5E) provides loading both FA and FB with the contents of a double scratchpad word.

The store F into double pad word micro (4E) provides storing the contents of both FA and FB into a double scratchpad word.

The count FA/FL micro (6D) provides incrementing/decrementing FA by the literal value contained in the micro instruction.

FB Register (Including FU, FT, FLC, FLD, FLE, and FLF)

The FB register is a 24-bit special purpose register which is functionally divided into several smaller units. The first subdivision consists of 4-bit FU and FT portions, plus the 16-bit FL portion. The FL portion is further subdivided into the 4-bit FLC, FLD, FLE and, FLF registers. The entire FB register is addressable as a whole, as is the 16-bit FL portion. In addition all four-bit portions (FU, FT, FLC, FLD, FLE, and FLF) are addressable separately as either a source or sink. Refer to figure 1-26.

The FU (field unit) register is used to hold the length of the unit which makes up a field of data in S-memory. The FU register can describe fields up to 15 bits in length. It is used primarily as a bias source for the CP register. Refer to the bias micro (3E).

The FT (field type) register holds a field type designator which has specific meaning only to software, and can therefore vary depending on the program in use. The FT register is not used by the hardware unless specifically sourced or sinked during execution of a micro.

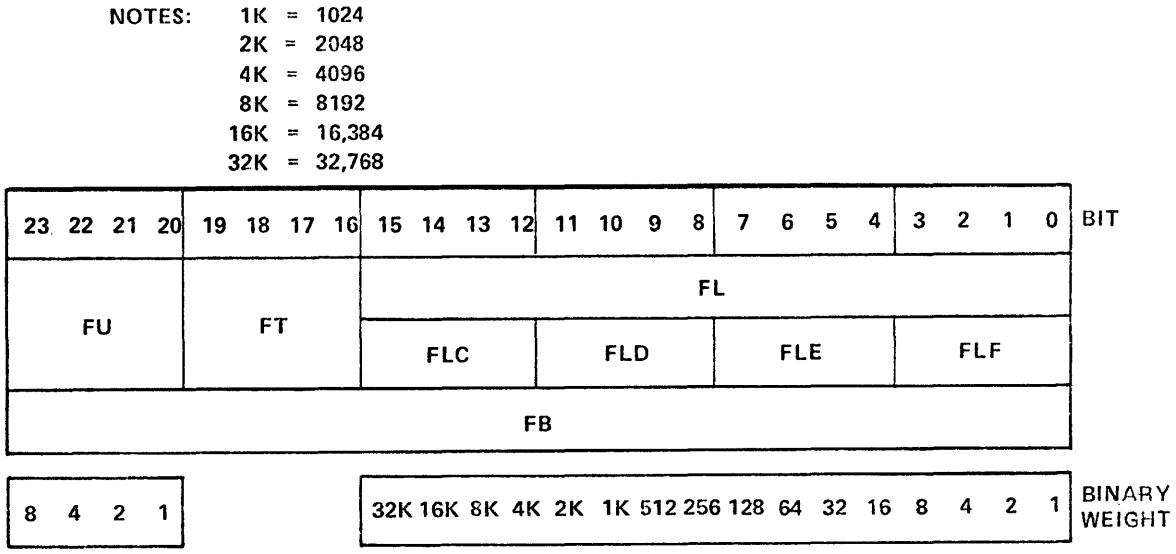


Figure 1-26. FB Register

The FL (field length) register holds the total length of a field of S-memory data. The value in FL is binarily weighted, and can describe fields up to 65,536 bits in length. It may be incremented or decremented by the literal value in a micro instruction or by the value in the CPL register. Overflow and underflow of FL can occur. Overflow is not detected, and causes FL to wrap around. Underflow is detected, and does not wrap around. When underflow occurs, a value of zero is left in the FL register. This register is also used as a bias source for the CP register, and for a static compare with the first word of right scratchpad. The static compare results are available when the pseudo register FLCN (field length conditions) is sourced.

Since the FB register is addressable as 4-bit sources or sinks, analysis and alteration of each of the 4-bit groups are provided via the 4-bit function box. The FB register has the ability (as a whole) to be loaded, stored, or swapped along with the FA register into a double scratchpad word. These three functions are provided by the 5E, 4E, and 7D micros, respectively.

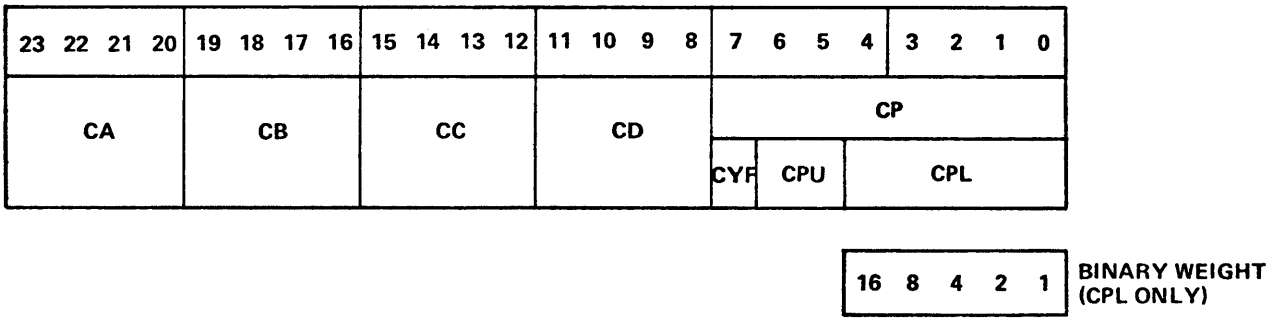
Inputs and outputs for FB are as follows: FB (as a whole) can receive or gate data from/to the MEX (24 bits). In addition, there is a 24-bit path directly to and from right scratchpad.

The FL portion may also receive or gate data from/to the MEX. Since FL is 16 bits long, only the least-significant 16 bits of the MEX are utilized. In addition there is a 16-bit path to/from the 16-bit FL adder which is used for incrementing/decrementing FL.

The 4-bit portions of FB (FU, FT, FLC, FLD, FLE, and FLF) may receive data from the 4-bit result bus, and gate data to the 4-bit auxiliary bus.

C-Register

The C-register (control register) is a 24-bit register which is not addressable as a whole. It is subdivided into four 4-bit registers (CA, CB, CC, and CD) plus the 8-bit CP register. The CP portion may be further divided into the 5-bit CPL portion, the 2-bit CPU portion, and the 1-bit CYF. Figure 1-27 illustrates the C-register. The functions of the various portions of C are described below.



NOTES: 1. **CA, CB, CC, CD AND CP ARE ADDRESSABLE AS SOURCE OR SINK**

2. **CPU IS ALSO ADDRESSABLE AS SOURCE OR SINK AS SUCH IT IS TREATED AS A 4-BIT REGISTER WITH THE LEFTMOST 2 BITS ALWAYS EQUAL TO ZERO**

Figure 1-27. C-Register

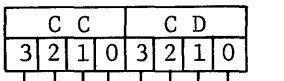
CA and CB Registers

The 4-bit CA and CB registers have no special function. They are available as general purpose storage, and are addressable as either source or sink. Inputs to both are from the 4-bit result bus or the least-significant four bits of the MEX. Output is to the 4-bit auxiliary exchange.

CC and CD Registers

The 4-bit CC and CD registers are used for storage of various processor states and conditions. Both can be addressed as either source or sink. The various flag bits are set automatically by the processor when the corresponding state or condition exists, and this occurs regardless of what has been stored in the register previously. The significance of the flag bits is shown in table 1-2. Since both CC and CD are addressable as source and sink, alteration and analysis of either by the 4-bit function box is possible. Inputs are from the 4-bit result bus or the least-significant four bits of the MEX. Output is to the 4-bit auxiliary exchange.

Table 1-2. CC - CD Registers Defined



CD0 Write/swap address out-of-bounds interrupt condition.
The FA register is either less than the BR register or more than the LR register.
The memory cycle occurs only when CD2 is true.

CD1 Read address out-of-bounds interrupt contion.
The FA register is either less than the BR register or more than the LR register.
The memory cycle will occur.

CD2 Write/swap address out-of-bounds override control.
When true, a memory write/swap cycle will occur regardless of the out-of-bounds condition.

CD3 Memory read data parity error interrupt.

CC0 Console control panel interrupt condition.
Indicates the Interrupt switch on the console is on.

CC1 I/O bus interrupt.
Indicates one or more of the soft I/O controls requires servicing by the processor.

CC2 Timer interrupt.
Set every 100 milliseconds by the real time clock.

CC3 Console control panel state lamp.
Set via the software microprogram, to indicate via the state lamp on the console, that the system is in the Control State (MCP in control). When the lamp is off, normal state (user's program executing) is true.

NOTE

With the exception of CD0, CD2, and CC3, the state of the CC and CD levels have no significance to the hardware. They must be examined by way of micro-instructions and used accordingly.

CP Register

The eight-bit CP (control parallelism) register is addressable as either a source or sink, using the eight least-significant bits of the MEX. The CP register consists of three basic sub-registers (CYF, CPU, and CPL), each having its own special functions:

CYF: The carry flip-flop (CYF) consists of bit 7 (most-significant bit of CP). This flip-flop is used to store a carry level on add operations or to store a borrow level on subtract operations. This function is activated by the associated CYF logic. The CYF flip-flop is not addressable as such, but may be sourced or sinked as part of the CP register. Such manipulation is a function of software.

CPU: The control parallelism unit (CPU) register consists of bits 5 and 6 of the CP register. It is used to designate the type of arithmetic operation the processor performs. If CPU equals 00, the operand data and result are handled as binary. If CPU equals 01, 4-bit binary coded decimal (BCD) is specified. If CPU = 10 or 11, the operation is undefined. The contents of CPU are used to determine the least-significant unit of Y (LSUY) and least-significant unit of X (LSUX) functions which are obtained when pseudo registers (BICN and XYST, respectively) are sourced. The CPU register is addressable separately as either a source or sink, and is handled as if it were a 4-bit register. The two (non-existent) most-significant bits are always equal to zero when CPU is sourced. Accordingly, CPU may be analyzed and/or altered by the 4-bit function box. Inputs to CPU are from the least-significant two bits of the MEX or 4-bit result bus. Output is via the auxiliary 4-bit bus only.

CPL: The control parallelism length sub-register (CPL) consists of bits 0 through 4 of the CP register. It is used to designate the field length of data to be handled on arithmetic operations. In addition, CPL designates the data field length (in bits) of S-memory read/write operations when FL is equal to zero. The CPL registor is not addressable separately, but must be sourced or sinked as part of the 8-bit CP register.

U-Register

The U-register is a 16-bit register which is used exclusively as an entry device for data from the cassette tape unit on the console. Since the tape data is read in serial fashion, the U-register serves as a buffer by accumulating bits until full, then gating them simultaneously to the MEX. As such, U serves as a source register only. Associated with U is an 8-bit shift register which is used for parity error correction of the tape data. This circuitry is capable of correcting errors in up to three sequential data bits from the tape, and utilizes the cyclic redundancy character (CRC) which is part of the tape data format. Refer to the B 9490 Cassette Tape Subsystem Field Engineering Technical Manual, form number 1067402 for further information if desired. The U-register functions somewhat differently in the processor tape and run modes of operation. Therefore, each is described separately.

Tape mode: Selecting the tape mode causes the contents of U to be automatically moved to the M-register, and implies that the tape data consists of 16-bit micro instructions which are to be executed. Therefore, sequential execution of micros from the tape continues unless the micro being executed dictates otherwise. An example of this would be a register move micro designating U as source and some register other than M as sink. In this case the automatic U to M move is disabled, and the next 16 bits of data are moved to the selected sink.

Run mode: In the run mode, cassette operation is controlled by the 2E (cassette control) micro. Tape data is obtained by sourcing the U-register, and it is necessary that the program running provide the appropriate micros to do this. Since cassette operation is relatively slow in comparison with the processor, completion of a micro sourcing the U-register is delayed until U is full (all 16 bits of serial data have been latched into the register).

SCRATCHPAD MEMORY AND SFL

Scratchpad memory is a 16-word x 48-bit wide random access memory which is used for general purpose temporary storage of data to which frequent or repeated access is required. It may be thought of as a group of registers which are available for use at the discretion of the programmer. Scratchpad is functionally divided into left and right sections. Left scratchpad consists of the 24 most significant bits of each of the 16 words, while right scratchpad contains the corresponding 24 least significant bits of each. The 48-bit words are addressable as a whole, or as two 24-bit words. While all words may be sourced or sinked at will, word zero of right scratchpad is reserved for a special purpose. It is divided into two sections which are known as the 16-bit SFL register and the 4-bit SFU register. These correspond to the FU and FL portions of the 24-bit FB register. Special monitoring logic performs a constant comparison between SFL and FL, and sets certain bits in the FLCN register based on the results. The contents of FLCN are used in decision circuits which deal with the execution of the bias micro (3E). The SFU and SFL registers are not addressable separately as sources or sinks, but may be accessed by addressing word zero of right scratchpad as a whole. Scratchpad is illustrated in figure 1-28.

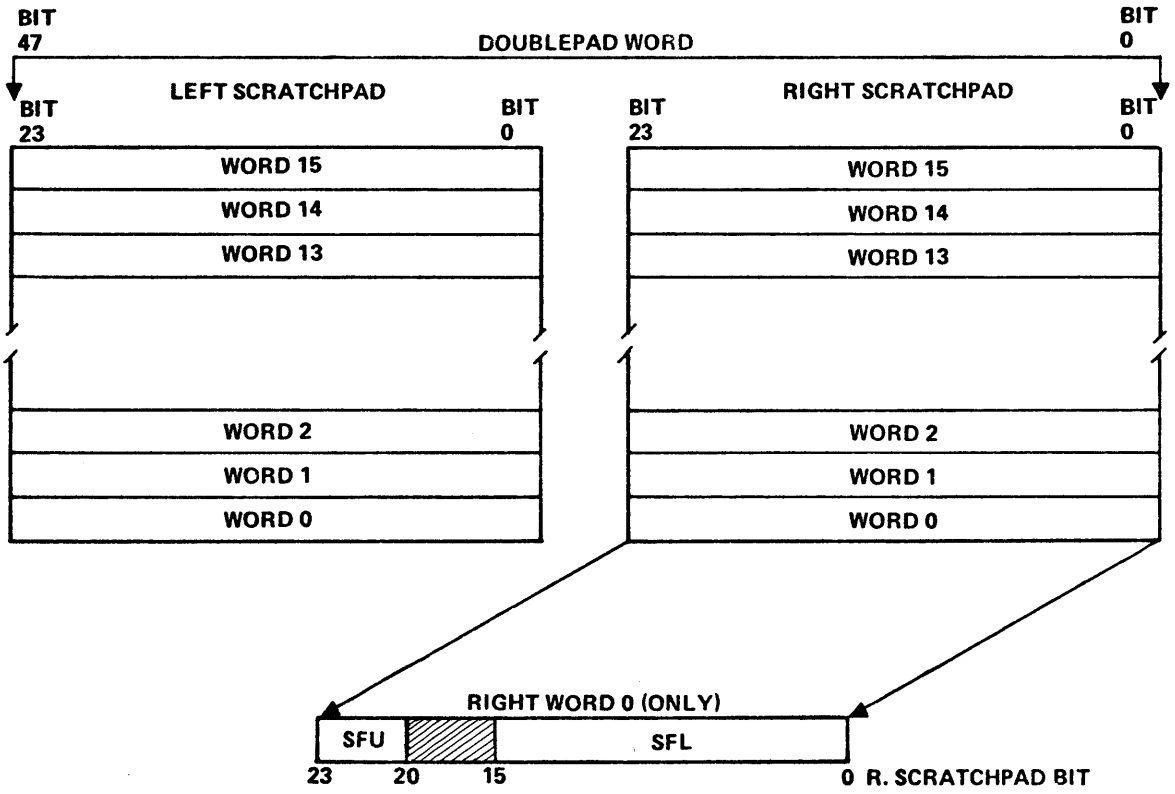


Figure 1-28. Scratchpad Memory

Several micro instructions are capable of addressing scratchpad. The scratchpad move micro (2C) can address any 24-bit word in either left or right scratchpad as source or sink. The exchange double pad word micro (7D) is capable of addressing any 48-bit word of scratchpad as both source and sink. In this case, the incoming data is stored in scratchpad's input latches until the resident data has been gated to FA & FB. The scratchpad relate FA micro (8D) is capable of addressing any 24-bit word of left scratchpad as a source.

The store F in double pad word micro (4E) is capable of addressing any 48-bit word of scratchpad as sink for the contents of both the FA and FB registers.

The load F from double pad word micro (5E) performs the reverse function, moving 48 bits from scratchpad to FA and FB.

PORT ADAPTER/PORT DEVICE INTERFACE CONTROL

The port adapter/port device interface control (PAPDIC) is the processor section which produces the control signals and functions necessary to access S-memory. Its overall purpose can be stated as exercising control over memory operations by gating addresses, data, and information concerning the memory cycle to and from S-memory at the appropriate times. The PAPDIC functions in a manner similar to that of an I/O control, and is required because S-memory is not a captive sub-unit of the processor within the B 1720 Series System. Instead, S-memory is arranged to provide data storage for several user (port) devices, of which the processor is but one. Access to S-memory for the processor or other port device is by way of a port adapter and the port interchange (see System Architecture in section 1).

To accomplish the task of controlling memory access, PAPDIC has been provided with the following functional capabilities:

a. Decoding of the basic three (read, write, and swap), overlay and dispatch micro operators which involve access to S-memory.

b. Requesting memory access from the port interchange. Access is granted or denied as determined by priority resolution logic (in the port interchange) which monitors the requirements of all user devices.

c. Independent control of its own operating sequence to insure gating addresses, data and other control signals to/from S-memory at the proper times.

d. Informing the port interchange of the type of memory cycle to be performed.

e. Transferring data field length and direction information (read/write operation) or destination port and channel number (dispatch operation) to the port interchange.

f. Allowing or disallowing writing in memory outside the bounds set by the processor BR and LR registers.

g. Gating address information and write data to memory, and read data from memory.

Memory Cycles

To understand the functioning of PAPDIC, a knowledge of the types of memory cycles which may be performed is needed. There are eight basic types, and although all involve reading and/or writing in memory, each is distinct because of its significance to processor operations. The memory cycles are described briefly below. If a more detailed explanation is desired, refer to the corresponding micro listing.

Read: Source the contents of a specified address in S-memory and gate to the MEX for distribution within the processor. The stored data is not destroyed by the read operation.

Write: Move a field of data from the MEX to S-memory, and store it at a specified location. Data previously stored at that location is destroyed.

Swap: Exchange a field of data from a source within the processor (X-, Y-, T-, or L- register) with the contents of a similar field stored at a specified location in S-memory. Both read and write operations are performed.

Fetch: source the contents of the address indicated by A+MBR, and move this data (actually a micro instruction) to the processor M-register for execution. This memory cycle is distinct in that is does not result from the execution of a micro instruction, but rather is hardware controlled, and is initiated by a single fetch signal. Handled by PAPDIC as a read operation, with preset data length and direction.

Console Read: source the contents of a specified address in S-memory and move to the console lamp register for viewing. Executed from a hardware forced micro instruction created by the console register selection switches. Handled by PAPDIC as a read operation with preset data length and direction. Performed only when the processor is halted.

Console Write: move the contents of the console switches to S-memory and write into a specified address. Also executed from a hardware forced micro created by the console register selection switches. Handled by PAPDIC as a write operation with preset data length and direction. Performed only with the processor halted.

Dispatch Write: lock the dispatch register in the port interchange to prevent access by another port device, then move the contents of the L-register to S-memory and write at address zero. This data comprises a reference address pointing to the location in S-memory where the transfer data destined for the receiving port device is stored.

Dispatch Read and Clear: read the data stored at address zero in S-memory and move to the processor L-register, and clear the dispatch register in the port interchange. This data was stored during a dispatch write operation by another port device, and comprises a reference address for incoming transfer data.

All of the above memory cycles, except as noted, are caused by the execution of a specific micro instruction. The first three (read, write, swap) are known as the basic three, and are distinct in that concurrent operation is allowed during their execution. Concurrency means that two micro instructions are permitted to execute at the same time, and is possible because of the semi-autonomous nature of the PAPDIC logic. Since a memory cycle for the most part involves circuits external to the processor, it is feasible to carry out non-conflicting processor operations while awaiting the cycle's completion. Concurrency is utilized because it allows increased processor operating speed. Refer to the operation section for further information on concurrent operation.

Dispatch operations differ from the basic three memory cycles in that transfer of information to another user device of S-memory is involved. The S-memory is used for intermediate storage of the data being transmitted, and moves into and out of it are done by performing normal read/write operations. The dispatch micro is used to signal or test for the presence of such transfer data, and to store or retrieve a reference address pointing to the data location in S-memory. Note that the dispatch micro has several variants which, if selected, cause operations that do not involve access to S-memory. These are the lockout variants which lock the dispatch register in the port interchange, and the port absent variant which is used for exercising the port interchange logic when performing test routines. Refer to the discussion of the port interchange for further information.

PAPDIC Operation

The overall functioning of PAPDIC is best explained in terms of its inputs and outputs. These are shown in figure 1-29. When a memory access micro is decoded, it is first necessary to gain access to memory. This situation exists because the processor competes with other port devices within the system for memory utilization. The processor memory request is transmitted

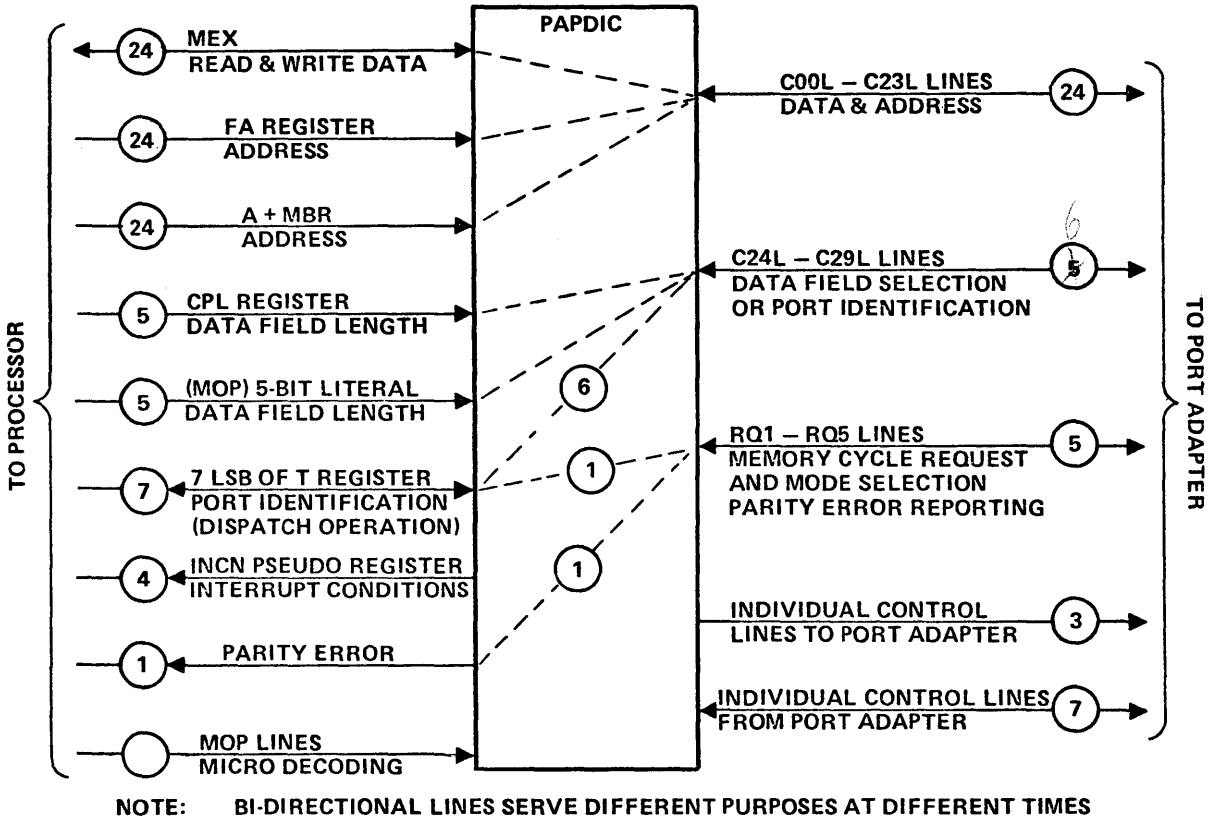


Figure 1-29. Port Adapter/Port Device Interface Control (PAPDIC) Inputs and Outputs

to the port interchange by way of the five request lines. Contained within the request is a coded signal which describes the type of memory cycle which is to be performed. The code is illustrated in figure 1-30.

Simultaneously with requesting memory access, the desired memory address, plus data field length and direction information are placed on the control lines. The address may come from the FA register or A+MBR register, as directed by the micro. Likewise, the data field length information may come from the CPL register or from a 5-bit literal in the micro itself. Both the processor and PAPDIC then wait in an idle condition until the request is granted. This is signaled by an "address accepted" level from the port interchange, which initiates the remainder of the operation. At this time control of the memory cycle is passed to PAPDIC, and the processor is freed to perform a concurrent micro, provided that 1) the memory cycle is one of the basic three, and 2) the next sequential micro is one from the concurrent set (12C, 13C, 6D, 7D, 8D, 9D, 3E, 4E, and 5E)

Since a memory operation involves many steps, it is not possible to control with a single activating signal. Therefore, PAPDIC is equipped with decision logic which selects a course of action which is determined in part by intermediate results.

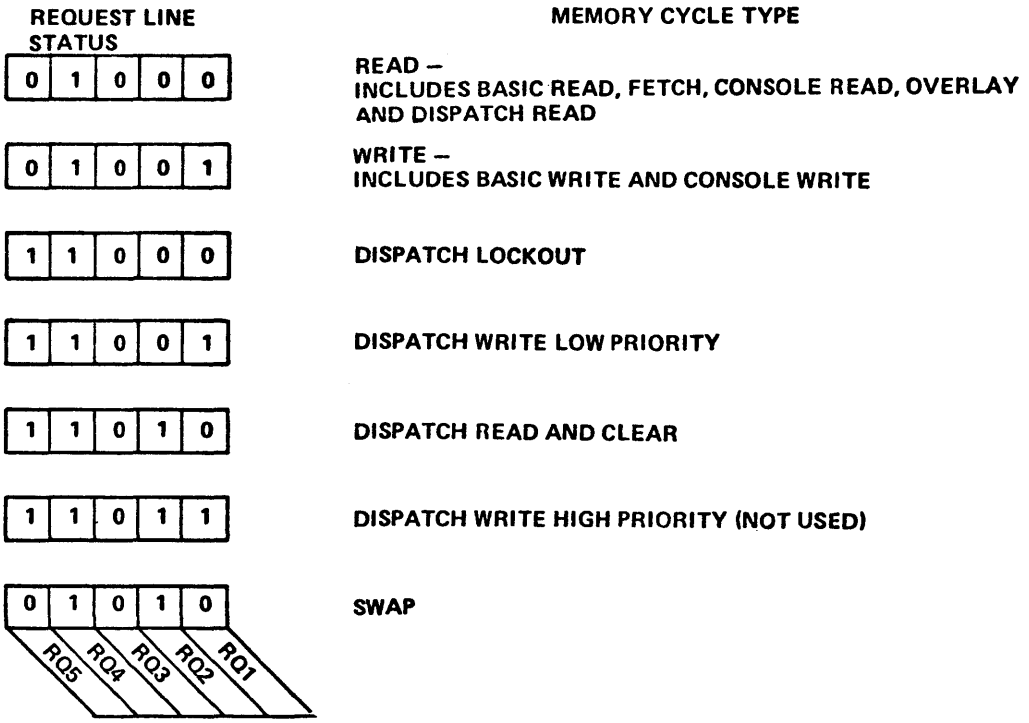


Figure 1-30. Memory Cycle Mode Signals (as Appearing on the PAPDIC Request Lines)

The decision logic controls a device known as the sequencer, which can produce eight activating signals, each of which conditions the PAPDIC operational circuitry to perform a specific function. These eight conditions are known as current states, and every memory cycle involves some combination of them. Note that PAPDIC current states are separate and distinct from those of the processor. The current states are known as idle, address, read, swap, write, wait, hold, and skip. They are defined as follows:

Idle: No operation. The logic is ready and waiting for a memory access micro. The address current state always follows idle, but the request code and data field length and direction are set in idle before proceeding.

Address: Transfer of address information to S-memory. The PAPDIC logic remains in this state until the address has been accepted. Always follows the idle state, but can branch directly to any other state.

Read: Receive read data from S-memory. Once entered, this state continues until the data has been gated to the MEX.

Swap: A one-clock-long writing state used exclusively by the 2D swap micro. Swap is basically the same as write except that it is always followed by the read state. Note that even though a swap operation involves reading the desired data from memory before writing over it, the write data is gated through PAPDIC before the read data is accepted from memory.

Write: A one-clock writing state entered from the address state. The write data is gated from the MEX to S-memory.

Wait: Similar to the read state except that the read data is not accepted. Entered from the address state, and exit to the read state. Wait is used to delay completion of a memory cycle until good read data is present.

Hold: A pseudo-idle read state which is entered from the (actual) read state. Read data is frozen in the console lamp latches for viewing. The hold state continues until a new console read is initiated or the register selection switch is moved off READ. Hold is not used when the processor is running.

Skip: A one-clock state entered from read, and used to announce the results of a dispatch lockout attempt.

The significance of the current states becomes apparent when the sequence in which they are used is related to the memory cycles which are composed thereof. The following flow chart (table 1-3) describes the memory cycle/current state relationship. Note that all cycles begin and end with idle, with address always being the second step.

Table 1-3. Sequencer Flow

Memory Cycle	Sequencer States
Basic 3 Read	Idle→ Address→ Read→ Idle
Basic 3 Write	Idle→ Address→ Write→ Idle
Basic 3 Swap	Idle→ Address→ Swap→ Read→ Idle
Dispatch Write Lock	Idle→ Address→ Write→ Idle
Dispatch Read and Clear	Idle→ Address→ Swap→ Wait→ Read→ Idle
Fetch S-Memory	Idle→ Address→ Read→ Idle

PORT ADAPTER

The port adapter is functionally a part of the port interchange, and serves to interface it with the data and control lines from the processor. It consists of line drivers and receivers for the 24 bi-directional data lines and the 21 bi- and uni-directional control lines, plus associated enabling logic to provide gating at the proper time. The port adapter's sole function is to control gating of the various signals back and forth as determined by its own control inputs from the port interchange. A block diagram of the port adapter is shown in figure 1-31.

A port adapter is contained on its own individual logic card to allow inclusion or omission of this input-output circuitry as determined by the number of port devices in a given system (the processor is a port device). All port adapters are identical, regardless of the type of port device employed.

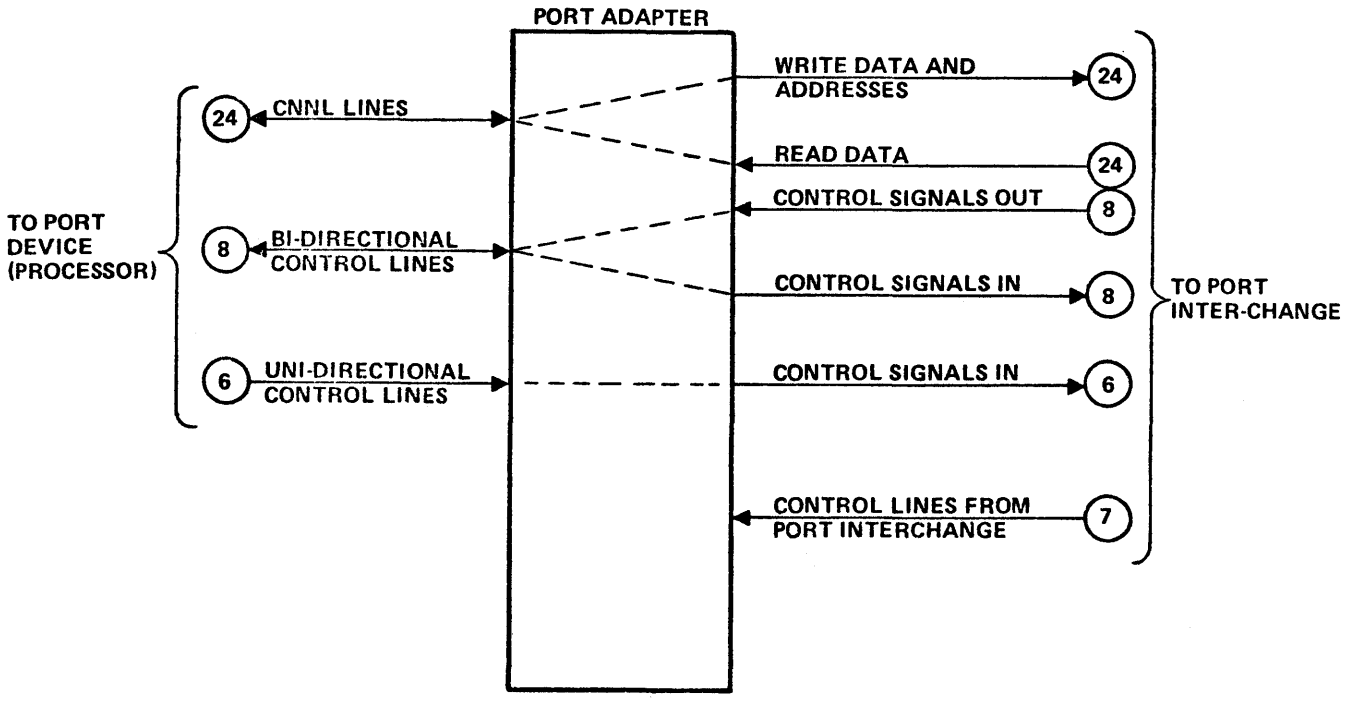


Figure 1-31. Port Adapter

PORT INTERCHANGE

The port interchange is that part of the system which controls all access to S-memory, providing a means of entry for up to three user (port) devices. Its overall function can be stated as follows:

a. Gating addresses to memory, and data to and from memory.

b. Data rotation to provide bit addressability for both read and write operations.

c. Generation and checking of parity.

d. Memory refresh address generation.

e. Priority resolution of memory access requests from user devices.

f. Control signal storage for dispatch operations.

To accomplish these functions, the port interchange has been configured as shown in figure 1-32. Each of the sections shown is described individually below.

In addition to its operational functions, the port interchange produces the following signals:

a. Port interchange clock. This 6-megahertz synchronizing signal is derived from the system clock in the processor. It is used within the port interchange, and supplied to each port device.

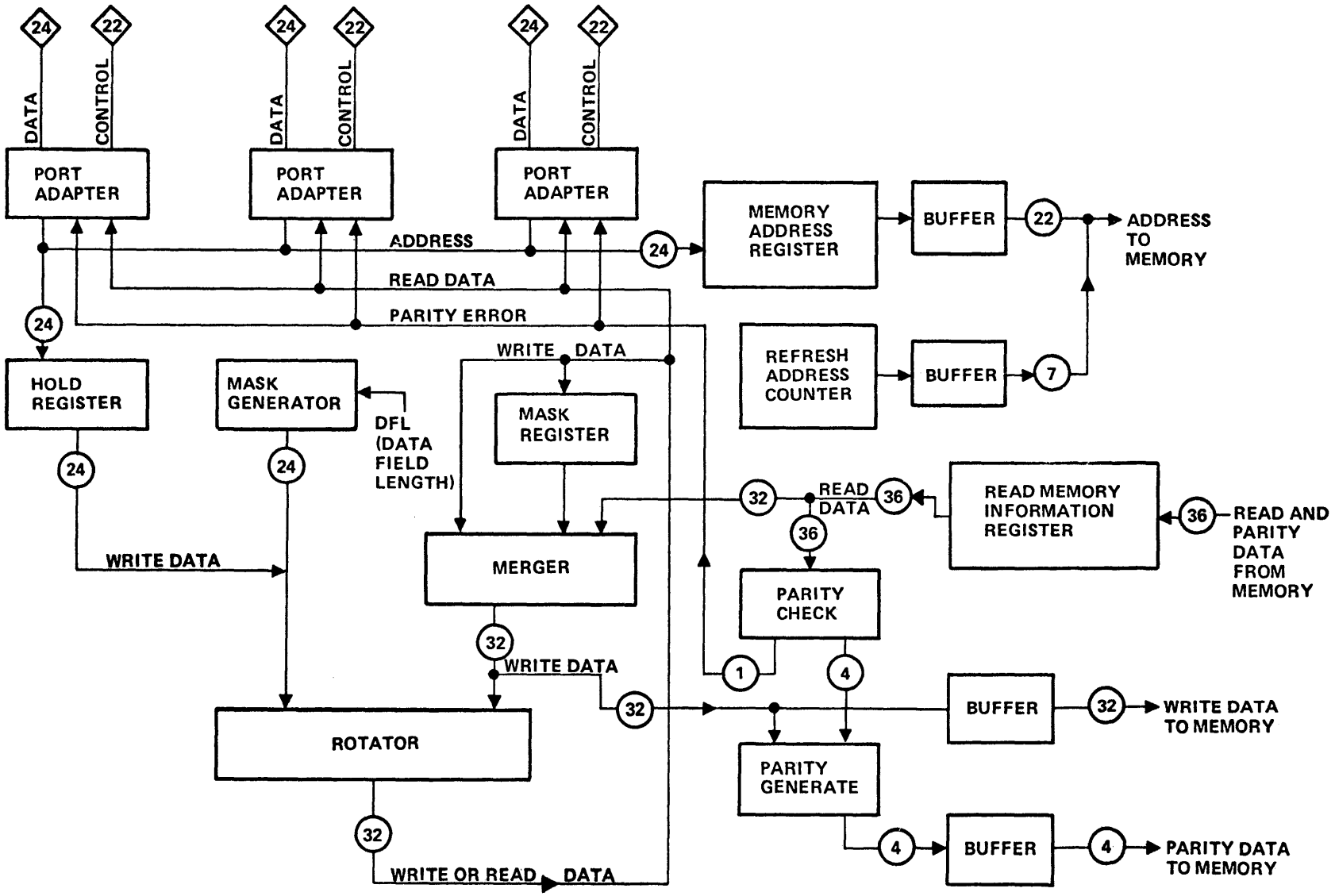


Figure 1-32. Port Interchange Block Diagram

b. Real time clock. This is a 10-hertz signal derived from the ac power frequency which is supplied to each port device for use in setting interrupts.

c. Clear signal. This is a one-time pulse which is generated in response to the master clear signal from the processor control panel, or to the power up clear signal from the processor logic power supply. It is distributed to each of the port devices.

PORT INTERCHANGE OPERATIONAL SECTIONS

Each of the following sections performs a function necessary to the overall operation of the port interchange. Refer to the operation section for a description of how the unit functions as a whole.

Hold Register

The hold register is a 24-bit wide latch which serves to store the incoming memory write data until the port interchange logic is ready to accept it.

Mask Generator

The mask generator is a device for translating the specified data field length into a corresponding number of enabling signals which determine the number of bits to be written into or read from memory. Its output is fed by way of the rotator to the mask register, and then to the merger. The mask generator is capable of producing up to 24 such 1-bits.

Mask Register

The mask register is a 32-bit-wide register used to store the output of the mask generator until it is used in the merging operation. It is 32-bits wide because this is the data length of all memory cycles regardless of the amount of data (up to 24 bits) which is entered from or gated to an external port device. The bit positions which the mask bits occupy are determined by the action of the rotator. The mask register may be used in either of two operational modes, depending on whether a read or write is being performed. The direct set mode is used for write operations, and this allows the mask 1-bits to be gated into the register.

Read operation entails use of the complement mode, which causes the complement, or inverse output of the mask generator, to be set into the register. The significance of these is explained in the merger description which follows.

Merger

The Merger is a switching device which is arranged to allow selectable through-gating either of two input lines for each of its 32 bit positions. These input lines carry respectively, read data, from memory and write data from the port device. Which of these is gated through the merger is determined individually for each bit position, and depends upon the contents of the corresponding bit position in the mask register. A mask 1-bit enables the write data input, with the converse being true (read data input enabled) for a mask 0-bit.

The Merger is used to combine old and new data during write operations, and to extract desired fields of data during read operations. Both of these actions are accomplished by selective application of the mask bits. During a write, mask 1-bits are present at those bit positions where the new write data from the port device is to be inserted in the 32-bit field read from memory. Therefore, the write data is allowed to pass to the output. In all other bit positions the existing read data passes through the merger unchanged. The entire 32-bit field is then written into memory, with the new write data displacing the previous contents of its bit positions, and that portion of read data which remains being restored to its original location. During a read, mask 1-bits are present at those bit positions where

read data from memory is to be excluded (the inverse of the desired extraction field). This results in the write data inputs of the merger being enabled for those bit positions. Since no write data is present during a read operation, zero fill occurs. In the desired extraction field the read data inputs are enabled by zero mask bits and the data from memory passes through to the rotator.

Rotator

The rotator is a two-stage multiplexing device, used to reposition a selected data field contained within the 32-bit block on which the port interchange operates. Functionally, it allows bit addressability for both reading and writing in memory, extending memory's versatility beyond the confines of the "per byte only" limitation imposed by the architecture and parity scheme employed. In addition, it serves to reposition the read or write data bytes to the proper sequence for being placed in memory or gated to the output. This is necessary whenever the 32-bit field addressed occupies more than one word, or level of memory. Refer to the field address logic portion of the S-memory discussion for further information. The rotator is similar to the rotator associated with the processor T-register, but is 32 bits in length. It is capable of providing data rotation (only) up to 32 bit positions, and functions in the right direction only. Effective left rotation is accomplished by revolution, as in the T-rotator. Note that it is also possible to pass data through the rotator without any rotation.

On a write cycle, the rotator is effectively located so as to reposition the incoming write data before it is merged with the read data which it is to modify. On a read, rotation is performed after extraction, but before the read data is gated to the port device.

MAR Register

The memory address register (MAR) is a 21-bit register which is used to store the starting address in S-memory where a read, write, or swap of data is to take place. It is used for temporary storage of the address, in the same manner that the hold register is used to store write data. Although 24 address lines enter the port interchange, MAR is limited to containing 21 address bits because the least-significant three (which specify a bit address) are not useful in addressing S-memory. Rather these three address lines are used exclusively by the rotation control logic which provides the desired bit addressability through action of the rotator. The significance of the various bit positions in MAR is illustrated in figure 1-33. Note that address bits 03 and 04 are used by both the rotation control and memory address logic. Address bits 21, 22, and 23 are not used for addressing, but rather to detect an address out of bounds condition.

Read Memory Information Register

The read memory information register is a 36-bit-wide register used for temporary storage of data read from memory. Like the Hold and MAR registers, it serves as a buffer between a data source and destination. It is 36 bits wide because each of the four data bytes (eight bits) read from memory has an accompanying parity bit. All 36 bits go to the parity checking logic, with only the 32 data bits continuing to the merger.

Refresh Address Counter

The refresh address counter is used to generate addresses for memory refresh operations. Refresh is required because of the dynamic storage characteristics of S-memory, and is accomplished by simply doing a read operation, but not using the data. The counter is preset to increment automatically by a binary weight of 32, then wrap around when the top of memory is reached. Refresh activity is carried on between memory cycles, and suitable logic has been provided to prevent interference between the two. Refer to the section on S-memory for further information.

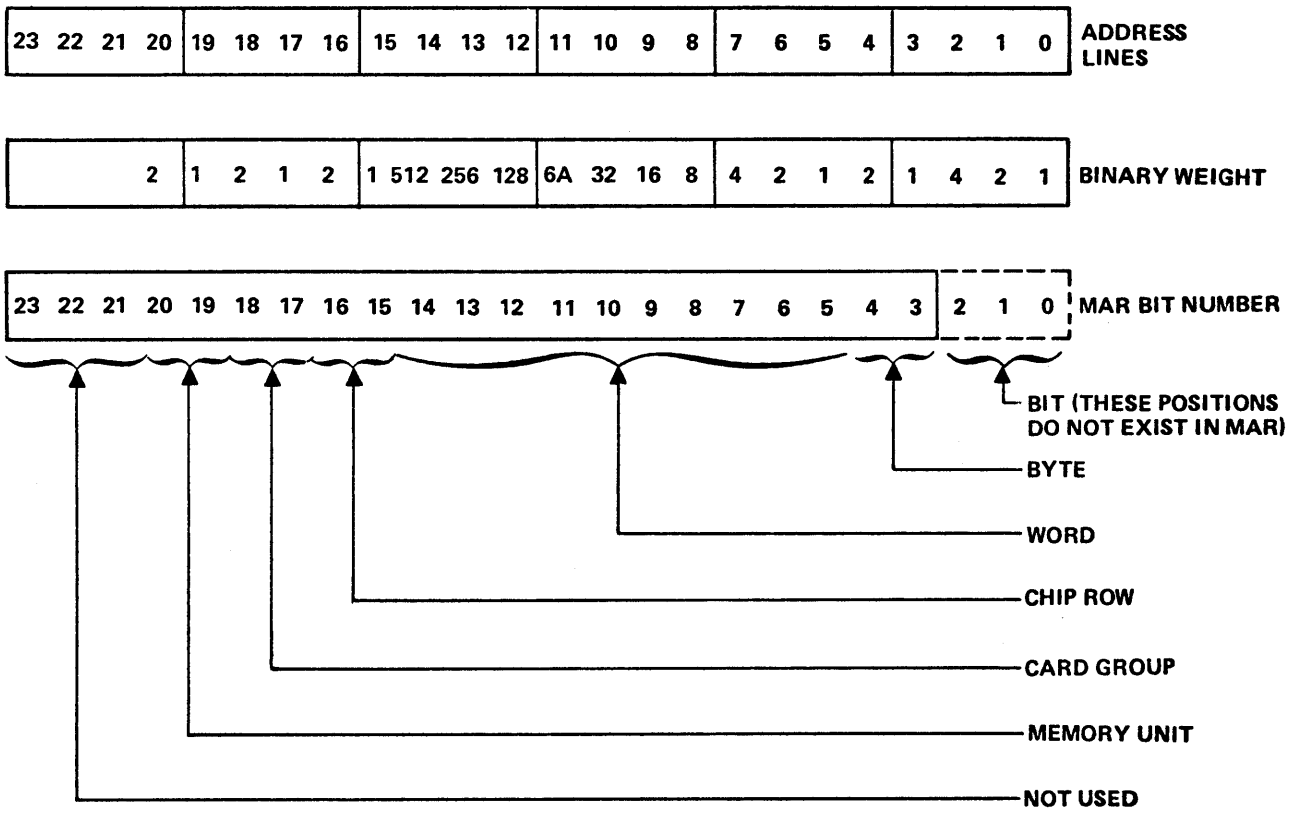


Figure 1-33. Memory Address Register Structure

Parity Generation and Checking Logic

The parity generation and checking logic is provided as a means of monitoring the proper functioning of S-memory data storage. It operates by generating and storing alongside each eight bits (byte) of write data a parity bit which has an arithmetical relationship to it. When this same data is read from memory, the parity bit accompanies it and an automatic compare is performed. Detection of errors is reported to the requesting port device, where they are handled in the manner prescribed for it. In the case of a processor micro fetch, detection of a parity error causes the machine to halt.

Generation of parity is accomplished by counting the number of 1-bits in each byte of write data, and either adding or omitting a one parity bit to make an odd total. Parity bits are stored in S-memory in the same manner as data, and are always present as either a 1 or 0, as appropriate. Upon reading the stored data from memory, the 1-bits within the data byte and parity bit position are counted, and if an even total is found, bad parity is flagged by way of a control level.

Priority Resolution Logic

Since each port device is independent, and can request memory access at any time, there must be some means of determining which of two or more simultaneous requests is to be granted. This is accomplished by assigning priorities to the user devices, and granting the memory

request of the highest unit when a conflict occurs. In the B 1720 Series System it is possible to have up to three port devices whose priorities are assigned as follows:

Unit	Value
Processor	0
Multi-line control	1
Other device (future use)	2

These priorities are known as line numbers, and the device with the highest line number bears the highest priority. Note that the processor is always the lowest priority user device of S-memory.

Dispatch Register

The dispatch register is a 14-bit register which is used for storage of control information for dispatch operations. Each of the 14 bits serves a specific purpose as shown in figure 1-34. In general terms the register accepts control information from a source port device during a dispatch write operation, and holds it until released by a read and clear operation performed by the destination port device. In so doing, it allows the two devices to operate independently and without need of synchronization with each other. They may thus communicate using S-memory as intermediate storage.

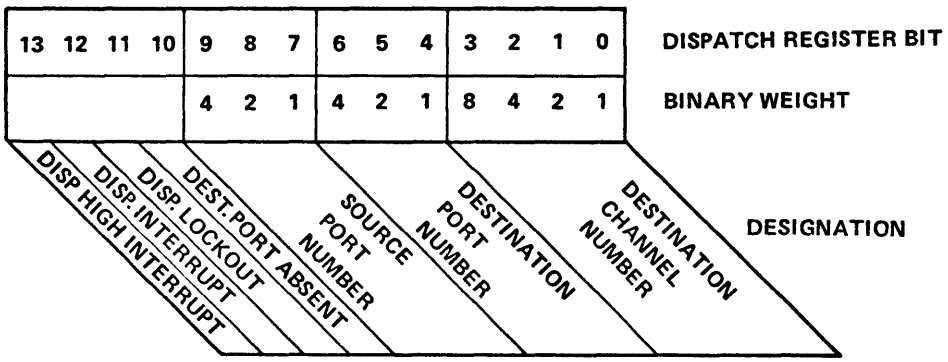


Figure 1-34. Dispatch Register

Although considered a 14-bit register, the dispatch register is not addressable as such. As viewed by a port device, the register's bits are accessible as follows:

Bits 00 through 03 = destination channel number (source or sink)

Bits 04 through 06 = destination port number (sink only)

Bits 07 through 09 = source port number (source only)

Bits 10 through 13 = dispatch conditions (source only)

The manner in which these designator groups are utilized is explained in the following individual descriptions:

Bits 00 through 03: these bits contain the channel number of the destination port device. They are loaded into the register by the source, and read out by the sink (destination) device, and generally define the peripheral unit associated with a port device which is to receive the dispatch data.

Bits 04 through 06: contain the destination port number. This number is the same as that assigned by the priority resolution logic, and is loaded simultaneously with the channel number. On a processor dispatch write, bits 00 through 06 come from the seven least-significant bits of the T-register. Note that the destination port number is used only within the port interchange to select the sink port device.

Bits 07 through 09: contain the source port number. The contents of these bits are derived directly from the priority resolution logic in the port interchange, and are loaded into the register when a port device initiates a dispatch write operation. The source port number is gated to the destination port device over the same lines it uses to send destination port numbers to the dispatch register. This serves to inform the destination port device of where the dispatch data originated.

Bits 10 through 13: contain dispatch conditions which are flagged by hardware actions within the port interchange. They are monitored continuously and are available to all port devices as control levels. In the processor these four levels are known as the INCN (interrupt conditions) pseudo register, and are analyzed by the 4-bit function box. Note that they appear in a different order when used in the processor. The derivations of the interrupt conditions is as follows:

Bit 10: destination port absent. This bit is set when a dispatch write operation to a missing port device is attempted. It is derived from the absence of a port device present level which each active port device sends to the port interchange. DPAL is used within the sending port device to signal that the attempted operation failed.

Bit 11: dispatch lockout. This bit is set when memory request lines RQ4 and RQ5 are true indicating that a dispatch operation has been initiated. It serves as a warning flag to port devices (other than the one performing the operation) that the dispatch register is busy.

Bit 12: dispatch interrupt. Set when memory request line RQ5 is true. This signal notifies the destination port that a dispatch operation is pending. The dispatch interrupt signal is a discrete level which is sent to only one port device at a time. Dispatch interrupts are handled by software within the processor.

Bit 13: dispatch high interrupt. Set when memory request lines RQ1, RQ2, RQ4, and RQ5 are true. Notifies all ports that a high priority dispatch operation is pending. This type memory cycle is not used at present.

PORT INTERCHANGE OPERATION

The port interchange is capable of performing four types of memory access operations: normal read or write and dispatch read or write. They constitute the basic hardware functions of which the memory cycles discussed in connection with the PAPDIC logic are composed. Note that these memory cycles may involve one or more of the basic hardware functions, and, in addition, be required to satisfy the external requirements. Since each of the four possible memory access operations involves a distinctly different procedure, separate descriptions are provided.

Read Operation

A read operation involves accessing a selected byte boundary location within S-memory, and gating the data contained within a 32-bit field above or below that address to the port interchange. There the desired portion (up to 24 bits) of this data field, as described by the bit address and data field length, is extracted and repositioned to coincide with the output data lines. This read data is then gated to the requesting port device for its use. Concurrently with retrieving the 32 bits from memory, a parity check is performed, and the results reported to the port device.

A read operation exercises the hardware in the following manner (refer to figure 1-35):

a. The incoming memory address is gated to the memory address register for temporary storage. From there it goes to the control logic in the memory unit, where the actual addressing is performed. Also gated to the control logic is the field direction sign which causes selection of the memory data field above or below the specified address, depending on how it is set. Note that the five least-significant bits of the address are used by the port interchange to determine the amount of data rotation required to access the desired bit address.

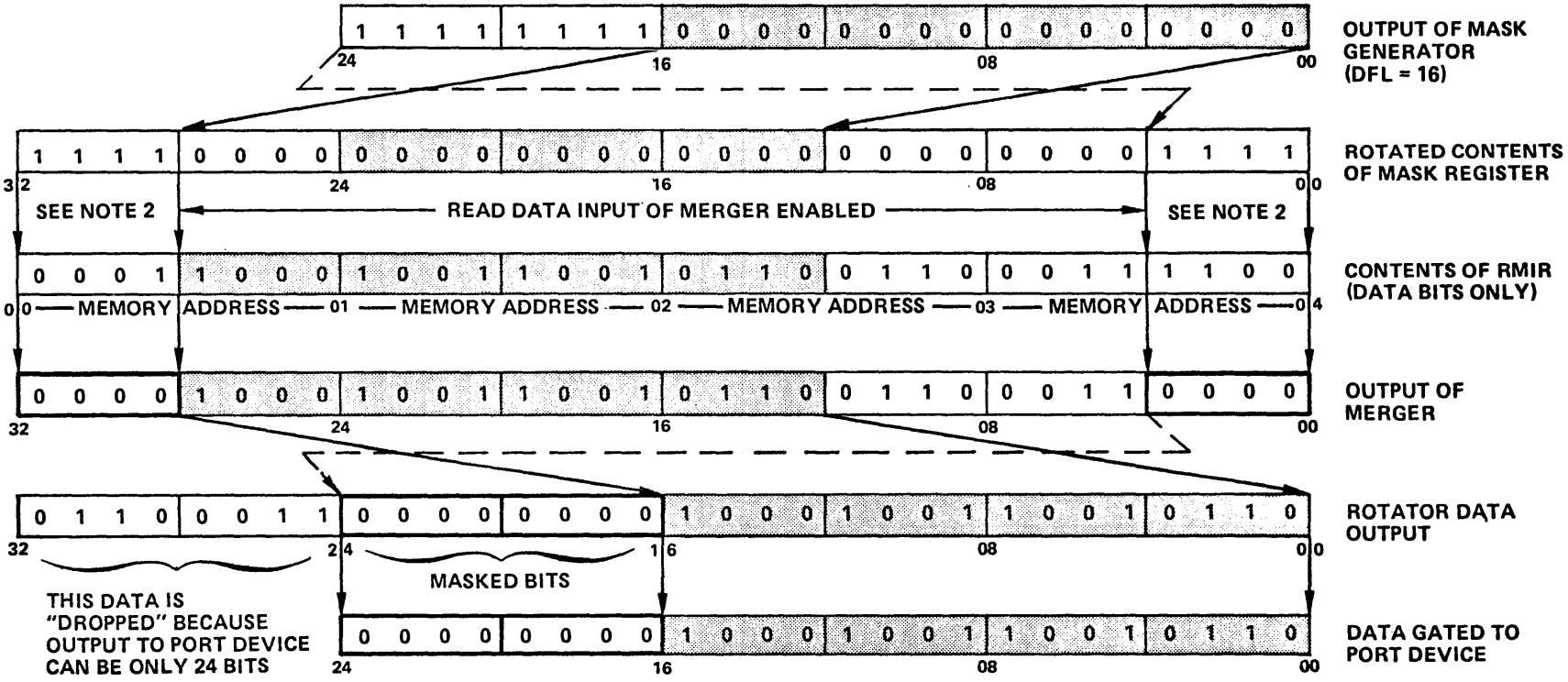
b. Data field length information is gated to the mask generator, where its binary encoded value is used to produce the appropriate number of mask bits. These mask 1-bits are gated to the rotator, where they are repositioned as a block to create a mask corresponding with the desired data within the 32-bit read data field from memory. Since this is a read operation, the mask register operates in the complement mode. This means that the mask has 0-bits in those bit positions where read data is to be allowed through the merger, and 1-bits in all others.

c. The addressed 32-bit read data field is gated from memory to the read memory information register, from which it is available at the input of the merger. In those merger bit positions where a mask 0 bit is present, the read data bit is allowed to pass through to the rotator. Where a mask 1-bit is present, the write data input is enabled, overriding the read data input. Since there is no write data present during a read operation, zero fill occurs in these bit positions.

d. The output data from the merger, consisting of data bits and zero bits, is repositioned via the rotator so that the selected read data is right-justified with respect to the output data lines. Note that because all read operations involve sourcing 32 bits from memory, there are always at least eight undesired data bits passed to the rotator. This occurs because the mask can be no larger than 24 bits. However, this undesired data is never gated to the output because it is always repositioned to the eight most-significant bit positions of the rotator. The data in the eight most-significant bit positions is dropped because no output data lines exist for them.

Write Operation

A write operation, like a read, involves accessing a selected byte boundary location in S-memory and gating the data contained within the 32-bit field above or below that address



NOTES:

1. THIS EXAMPLE DEPICTS A 16-BIT DATA FIELD BEING READ FROM MEMORY ADDRESS 00, BIT 04. SIGNIFICANT MASK BITS AND READ DATA ARE TINTED. FIELD DIRECTION SIGN IS POSITIVE.

2. MASK GENERATOR OPERATES IN THE COMPLEMENT MODE FOR READ OPERATIONS. EACH MASK 1 BIT ENABLES THE CORRESPONDING WRITE DATA INPUT OF THE MERGER. SINCE NO WRITE DATA IS PRESENT ON READ OPERATIONS, ZERO FILL OCCURS IN THESE BIT POSITIONS.

Figure 1-35. Reading From Memory

to the port interchange. Refer to figure 1-36. This 32-bit field contains the address to which the write data coming from the port device is destined, and merging of the new data with the existing contents of the 32 bits occurs in the port interchange. The product of this process is then written back into memory at the originally accessed location. In the merging process, the write data replaces the read data as specified by the data field length and bit address. That read data which is not written over remains, and is restored to its original location in memory. A parity check is performed on the read data as it emerges from memory, and parity generation occurs for the composite old/new data as it is written into memory. Bad (even) parity is rewritten if bad parity was detected on the read operation for a given byte.

A write operation entails the following hardware actions (refer to figure 1-37):

a. As in a read operation, the memory address is gated to the memory address register from which it is used for the actual addressing. Along with the address goes the field direction sign for selection of the data field above or below the designated address. Note that the five least-significant bits of the address are likewise used to determine the rotator action necessary to reposition the incoming write data for merging at the specified bit address.

b. Data field length information is gated to the mask generator where it is used to produce the appropriate number of mask 1-bits. After being generated, the mask is repositioned by rotator action to match the specified bit address, and stored in the mask register. The mask register operates in the normal mode for a write operation, meaning that a mask 1-bit is generated for each write data bit which is to be allowed through the merger. The mask, therefore, corresponds to the portion of the merger where data replacement is to take place. It masks the undesired read data inputs by enabling the write data inputs.

c. The addressed 32-bit read data field is gated from memory and stored in the read memory information register, from which it is available at the input to the merger. Similarly, the write data from the port device is gated into the hold register, where it is stored pending the following steps of the operation. At the appropriate time, this write data is gated to the rotator where it is repositioned, as was the mask, to match the specified bit address.

d. The rotated write data and stored read data are then combined in the merger, with the former replacing the latter in those bit positions where a mask 1-bit is present. This 32-bit field is written back into memory at the same location from which it came.

Dispatch Read Operation

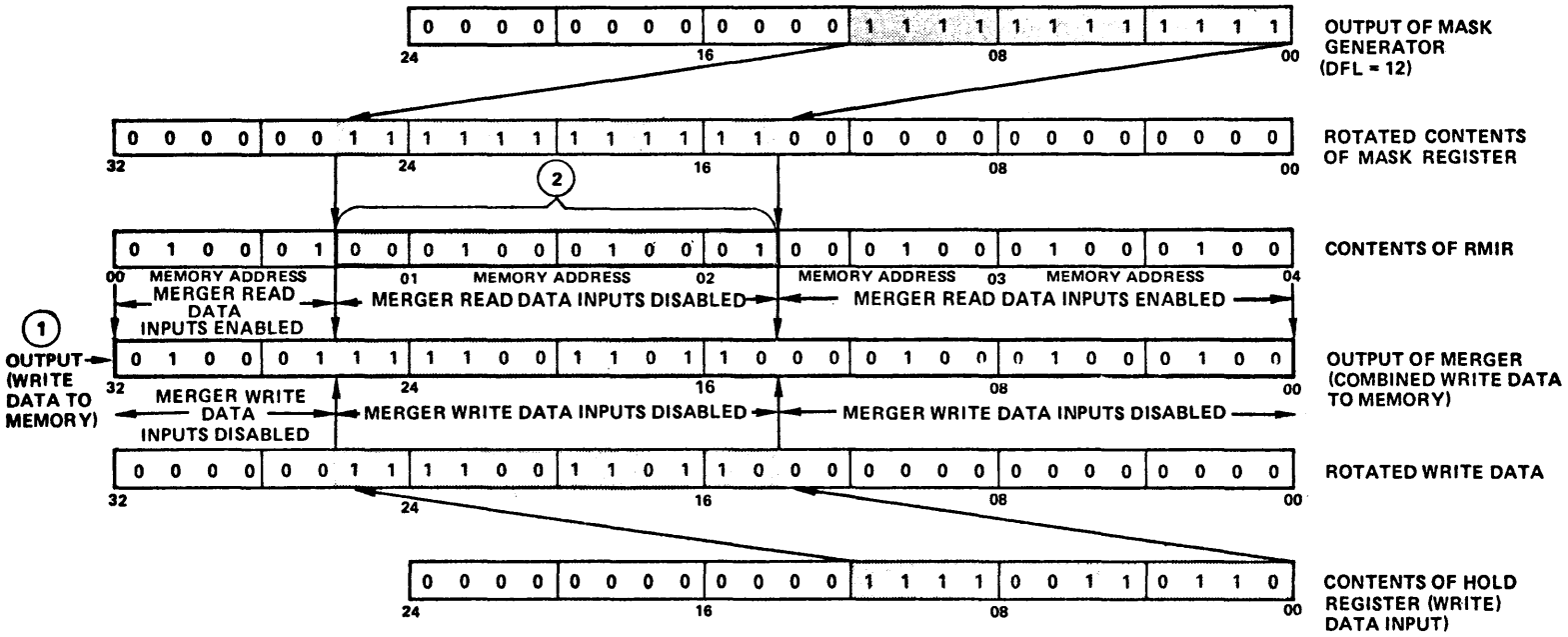
A dispatch read is similar to a normal read operation (see above), except that the disptach register is also utilized. Use of this register to store dispatch information does not affect the read process except that address zero of memory is always used. This is accomplished by resetting the memory address register. The data stored in the dispatch register is gated to the port device which requested the read via control lines. Doing this clears the register.

Dispatch Write Operation

A dispatch write is similar to a normal write except that, like a dispatch read, the dispatch register and address zero of memory are used.

S-MEMORY (SYSTEM MEMORY)

S-memory is the main memory within the B 1720 Series System. It is used for storage of data, micro programs (or the portions thereof which exceed M-memory's capacity), S-language



NOTES:

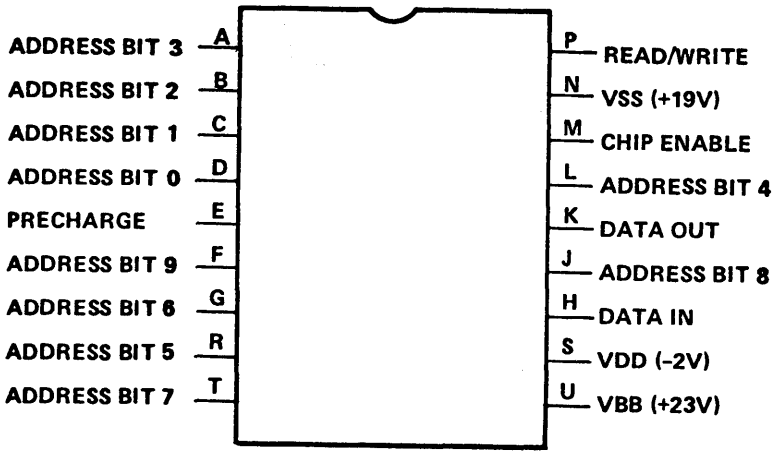
1. THIS EXAMPLE DEPICTS A 12-BIT DATA FIELD BEING WRITTEN INTO MEMORY AT ADDRESS 00, BIT 06. SIGNIFICANT MASK AND WRITE DATA BITS ARE TINTED. THE LAST STEP (OUTPUT) IS MARKED ① AND IS LOCATED IN THE CENTER OF THE ILLUSTRATION.
2. MASK GENERATOR OPERATES IN THE NORMAL MODE FOR WRITE OPERATIONS. EACH MASK 1 BIT ENABLES THE CORRESPONDING WRITE DATA INPUT OF THE MERGER. THE READ DATA INPUT REMAINS ENABLED FOR THOSE BIT POSITIONS WHERE MASK 1 BITS ARE ABSENT. THE READ DATA THUS EXCLUDED IN THE ABOVE ILLUSTRATION IS INDICATED BY ②.

Figure 1-36. Writing In Memory

programs, and interpreters. The S-memory is composed of a network of random access memory chips, and is dynamic in nature (stores data only when power is applied). The storage chips are mounted on plug-in cards, each pair of which have a maximum capacity of 16,384 (16K) bytes of data. This permits extreme flexibility in assembling a memory of any desired size. In the B 1720 Series System a memory size from 8K bytes to 256K bytes may be selected. In practical systems, 48K of memory is the minimum memory size required to achieve the degree of performance of which the hardware is capable.

MEMORY CHIPS (RAM)

The memory chips used within S-memory are contained in an eighteen (18) pin DIP package. Each memory chip contains 1024 (1K) bit locations which are randomly accessible. Addressing these locations is accomplished by way of 10 binarily-weighted address lines (A0 through A9), with decoding internal to the chip. This involves simply applying the binary coded address, as control levels, to the chip address line inputs. The nature of data storage is by the presence or absence of an electrical charge on a capacitor element, each of which represents a bit location. Due to normal leakage, such charges require periodic renewal. This is accomplished by a refresh cycle which is described elsewhere in this section. The functions of the various pins on a memory chip are shown in figure 1-37.



BINARY WEIGHT OF ADDRESS BITS

BIT	WEIGHT
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512

Figure 1-37. Memory Chip

MEMORY LAYOUT

Since each memory chip can address only one bit position at a time, the minimum number required for a practical memory is determined by the quantity of data bits handled simultaneously during read or write cycles. This memory width is equal to 36 bits in the B 1720 Series System: 32 data bits (4 bytes) plus 4 parity bits. Accordingly, memory is constructed in increments of 36 chips. The chips are installed in sequential rows on the memory cards employed with the 36 chip width being split between two such cards. In all cases, performing a memory cycle causes a binarily-coded address to be gated to a group of 36 chips, accessing a single bit location in each. Because each individual chip contains 1024 such bit locations, such a 36-chip group is capable of storing 4096 data bytes (1024 addresses x 4 bytes). Note that the physical location of the chips actually addressed during a memory cycle may not be confined to one row, card group, etc. This is due to the fact that addressing is possible at any selected byte boundary, with necessary overlap into higher or lower physical divisions of memory being automatically implemented. The relationship between the bit addresses within memory and the physical arrangement of memory chips is shown in figure 1-38.

Note that a memory size of 65,535 bytes (64K) is shown. This may or may not be the size actually used in a given installation.

Viewed externally, S-memory may be regarded as a continuous bank of storage locations which are randomly accessible at any selected byte boundary. The physical divisions of chip rows, card groups, memory units, etc., are invisible to user devices.

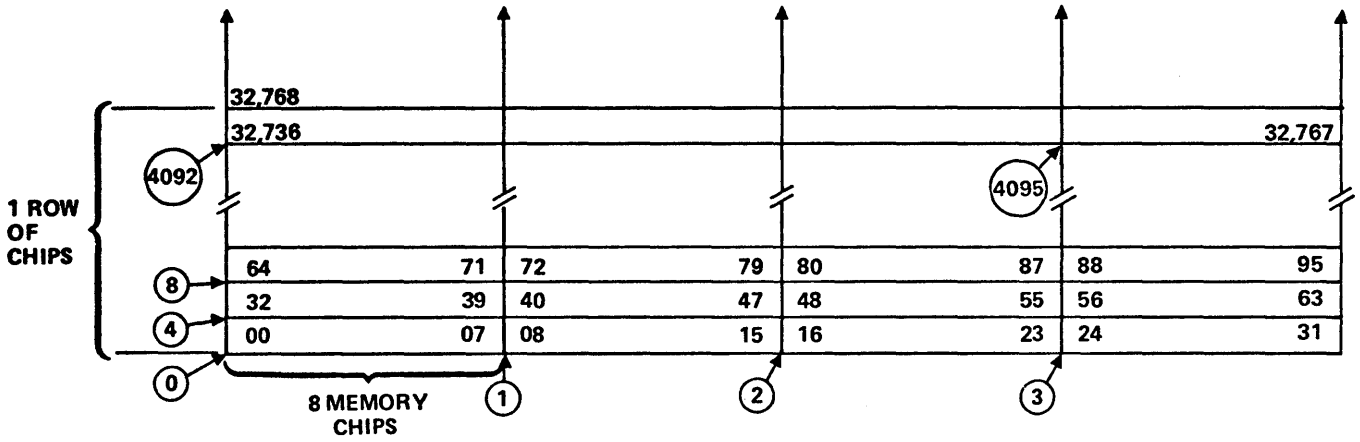


Figure 1-38. S-Memory Layout

MEMORY CARDS

The memory cards are standard size, B 1700 logic cards. Mounting of memory chips on the cards is by byte groupings. Each card is divided into two sections which each constitute a byte column as shown in S-memory layout illustration, figure 1-38. Note, however, that the card has four rows of chips, whereas only one is shown in figure 1-38. Each row within the byte column has nine chips; eight for data storage and one for parity. The byte columns represent 4096 bytes of data storage, with the full card (two columns) being equal to 8192 (8K) bytes. To allow further flexibility in the choice of memory size, half-populated memory cards are available. These contain only two rows of chips (4096 bytes total storage). All memory cards of the same storage capacity are identical, and therefore directly interchangeable without modification. The relative position a card represents within memory is determined by backplane connections and external control logic. Full- and half-populated memory cards are illustrated in figure 1-39.

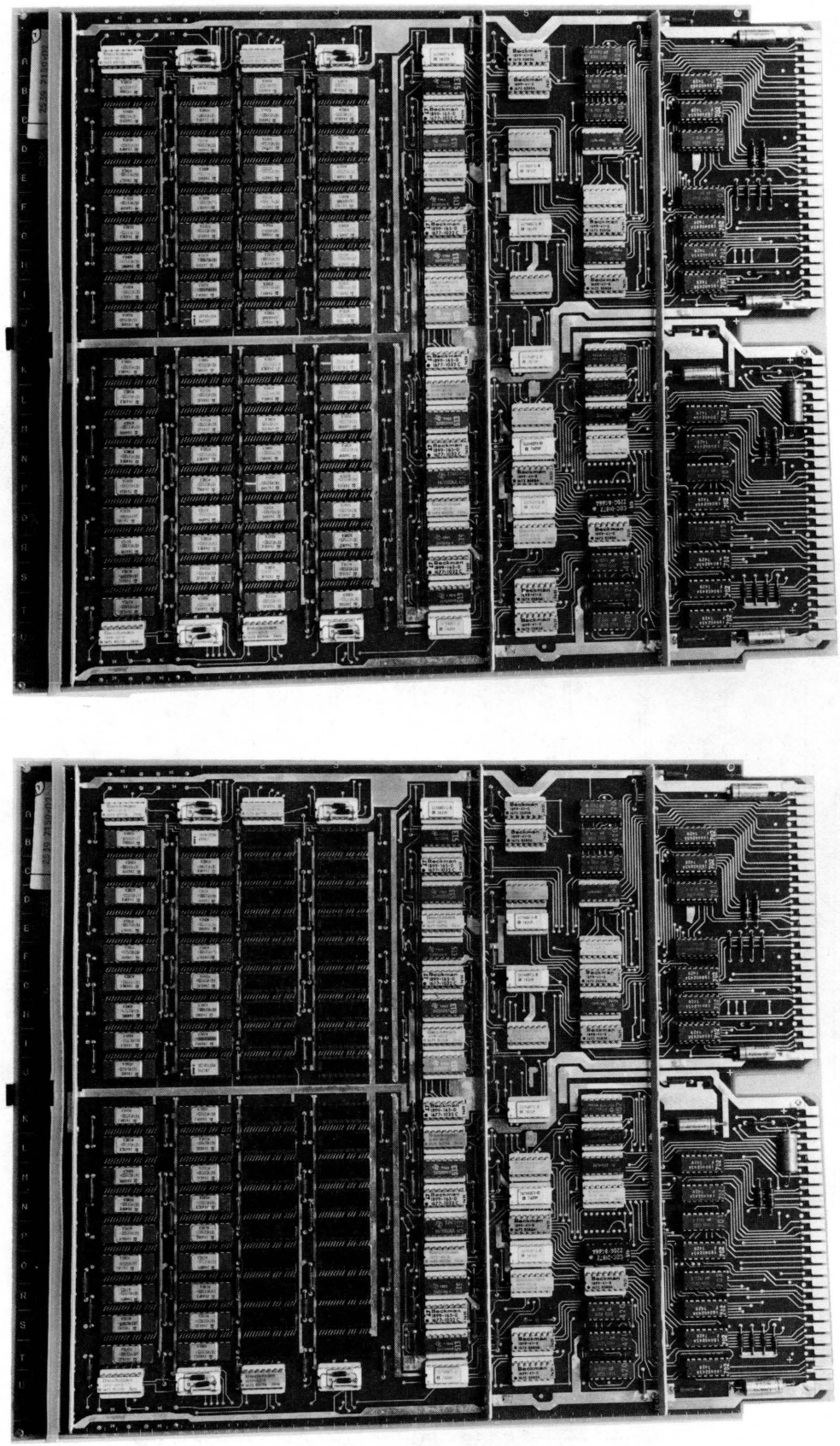


Figure 1-39. Full- and Half-Populated Memory Storage Cards

MEMORY GROUPING

The memory cards are always used in pairs, which are known as card groups. This is due to the fact that the full width of memory as discussed above requires 36-chip rows, whereas the individual cards have only 18 per row. Therefore, byte columns 0 and 1, as shown in figure 1-38 are always located on the first card of the group, with 2 and 3 on the second. A fully-populated card group contains 16,384 (16K) bytes total storage.

Memory card groups are assembled into memory units, each of which can contain a maximum of four such groups, for a total storage capacity of 65,536 (64K) data bytes. Each memory unit is accompanied by its own field address and interface control logic which is located on two additional logic cards.

A complete memory unit, therefore, consists of 10 cards: 8 memory and 2 logic. Each such is contained on its own backplane, and is powered by a separate memory power supply. Since memory units are independent entities, they are connected together in parallel with regard to address and data lines. Each contains wired reference values which are set by the field engineer to inform the memory unit of its relative position within memory. Because only one address may be accessed at one time, conflicts do not occur. A typical memory unit is illustrated in figure 1-40. Since the B 1720 Series System can employ up to 256K bytes of memory, the memory subsystem may comprise as many as four memory units. A frontplane view of such a memory subsystem is shown in figure 1-41.

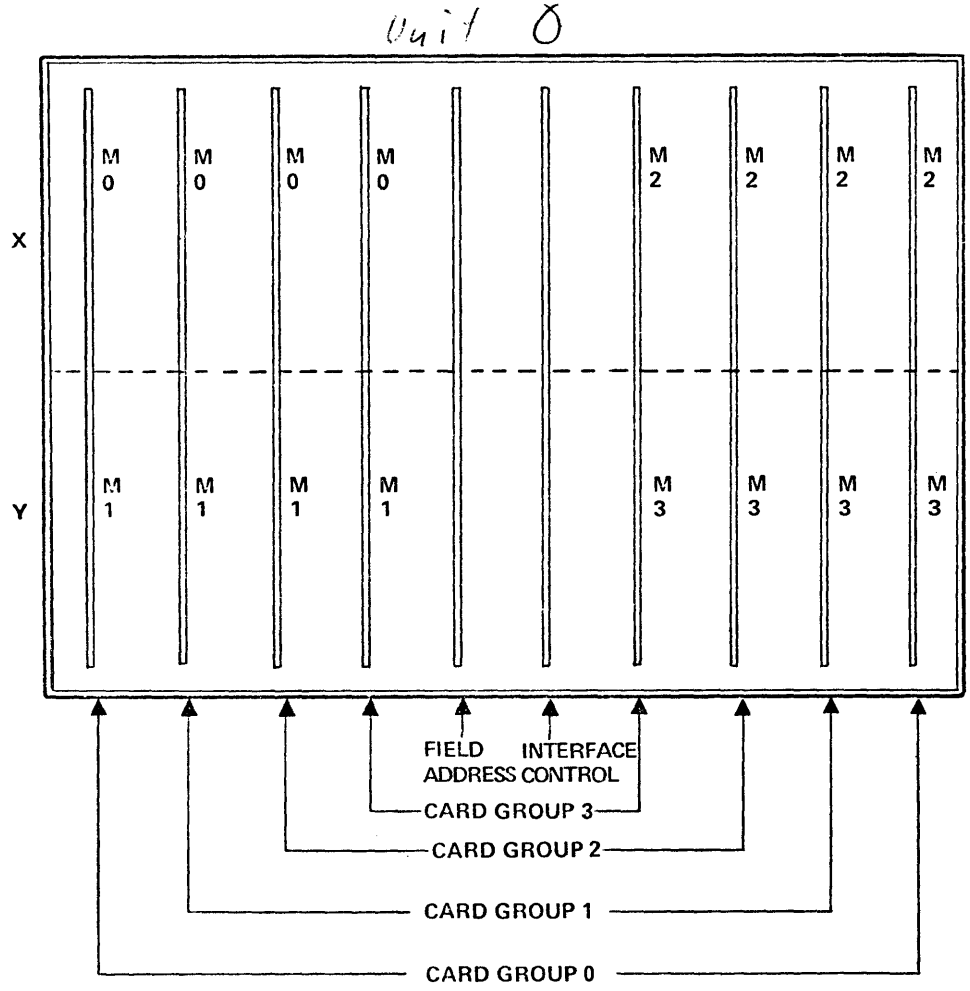


Figure 1-40. The 64 K-Byte S-Memory Unit (Viewed from Frontplane)

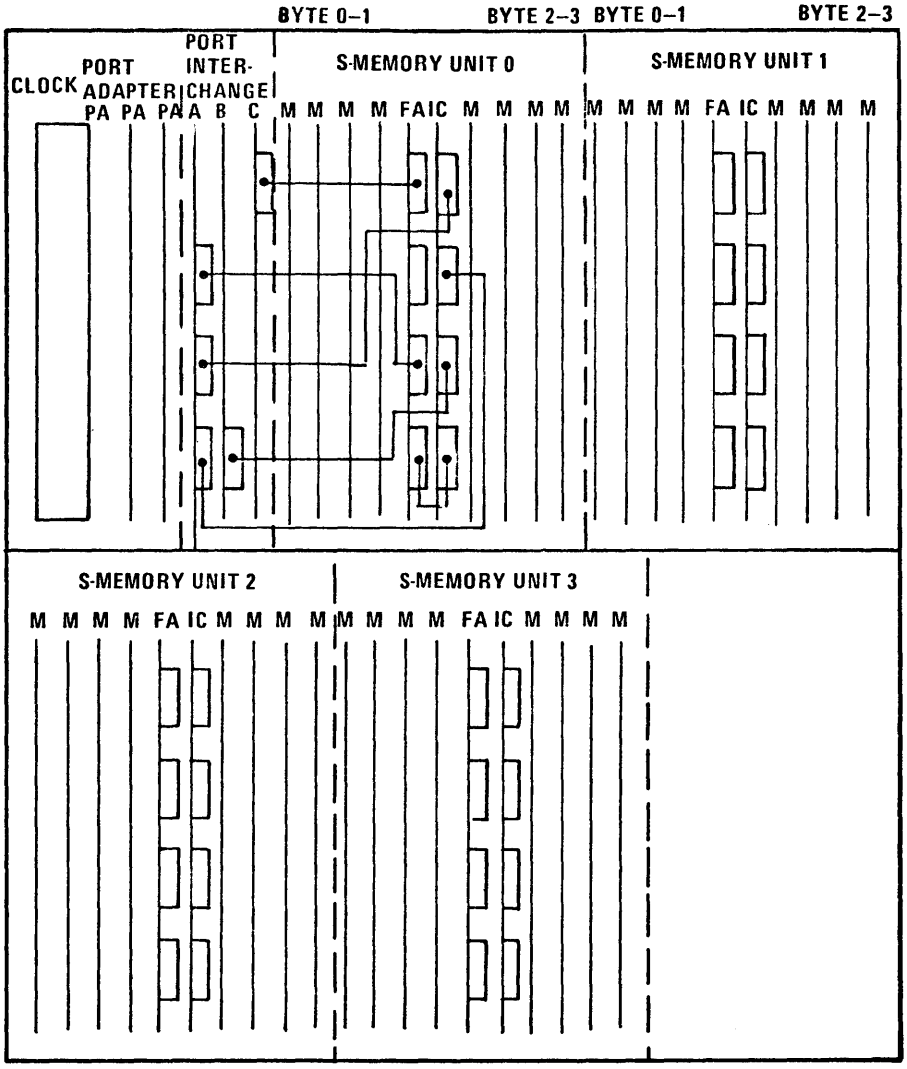


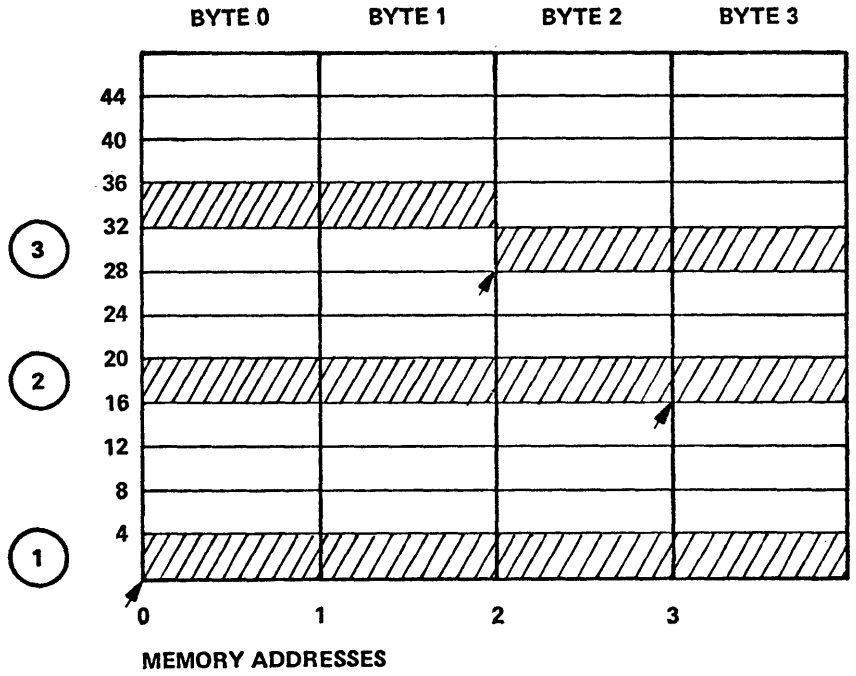
Figure 1-41. The S-Memory Subsystem Frontplane

INTERFACE CONTROL AND FIELD ADDRESS LOGIC

The interface control and field address logic controls all activities within the memory unit. The field address logic and interface control logic are each located on an individual card, and these occupy the center two positions of the memory unit. Their functions are distinctly different, and therefore, each is described separately.

Field Address Logic

The field address logic performs the basic function of translating the memory address received from the port device to the enabling signals needed to access the physical hardware devices containing the desired data. Although it would appear that selecting memory units, card groups, chip rows, and bit locations within chips is a simple procedure, the byte addressability and selectable field direction features of S-memory introduce additional complications. The reason for this is shown in figure 1-42. All memory cycles involve accessing a 32-bit (4-byte) data field. Unless the field selected begins at byte zero and proceeds in a forward direction, or begins at byte three and proceeds in a negative direction, it occupies two levels or words of memory. This fact requires that a means of address modification be provided to allow the physically separate portions of memory to function as a continuously accessible whole. The overlap, or differentiation between memory levels, can occur within the bit levels of a chip row, or between rows, and also between card groups and memory units. Therefore, the necessary correction logic must be capable of modifying all address bits up to and including those which select memory units.



NOTES:

(1) **MEMORY ADDRESS 0, FORWARD DIRECTION - NO CORRECTION REQUIRED (ALL BYTES ARE AT SAME MEMORY LEVEL)**

(2) **MEMORY ADDRESS 19, REVERSE DIRECTION - NO CORRECTION REQUIRED (SAME REASON)**

(3) **MEMORY ADDRESS 30, FORWARD DIRECTION - CORRECTION REQUIRED (BYTES 0 AND 1 ARE AT HIGHER MEMORY LEVEL)**

Figure 1-42. Address Modification

To provide address modification on a "per byte" basis, the distribution of address bits 5 through 18 is broken down into four individual channels, each of which controls the activation of these storage elements representing a byte column as illustrated in figure 1-43. Within each channel, the word, or memory level address, may be incremented, decremented, or passed unchanged to the enabling circuitry. The action which occurs is dependent on the "key byte" specified by memory address (bits 03 and 04), and the field direction sign. In all cases, however, it results in accessing either the preceding or following three bytes of memory. Note that only a one-word change (adding or subtracting one unit at the memory address bit 5 position) in the memory address is possible. However, because the memory address is binarily structured, this change can result in the activation of the succeeding chip row, card group, or memory unit. Even though the memory unit level is beyond the range of addresses dealt with by the field address card logic, provision has been made within them to allow activation of adjacent memory units when such a field overlap occurs.

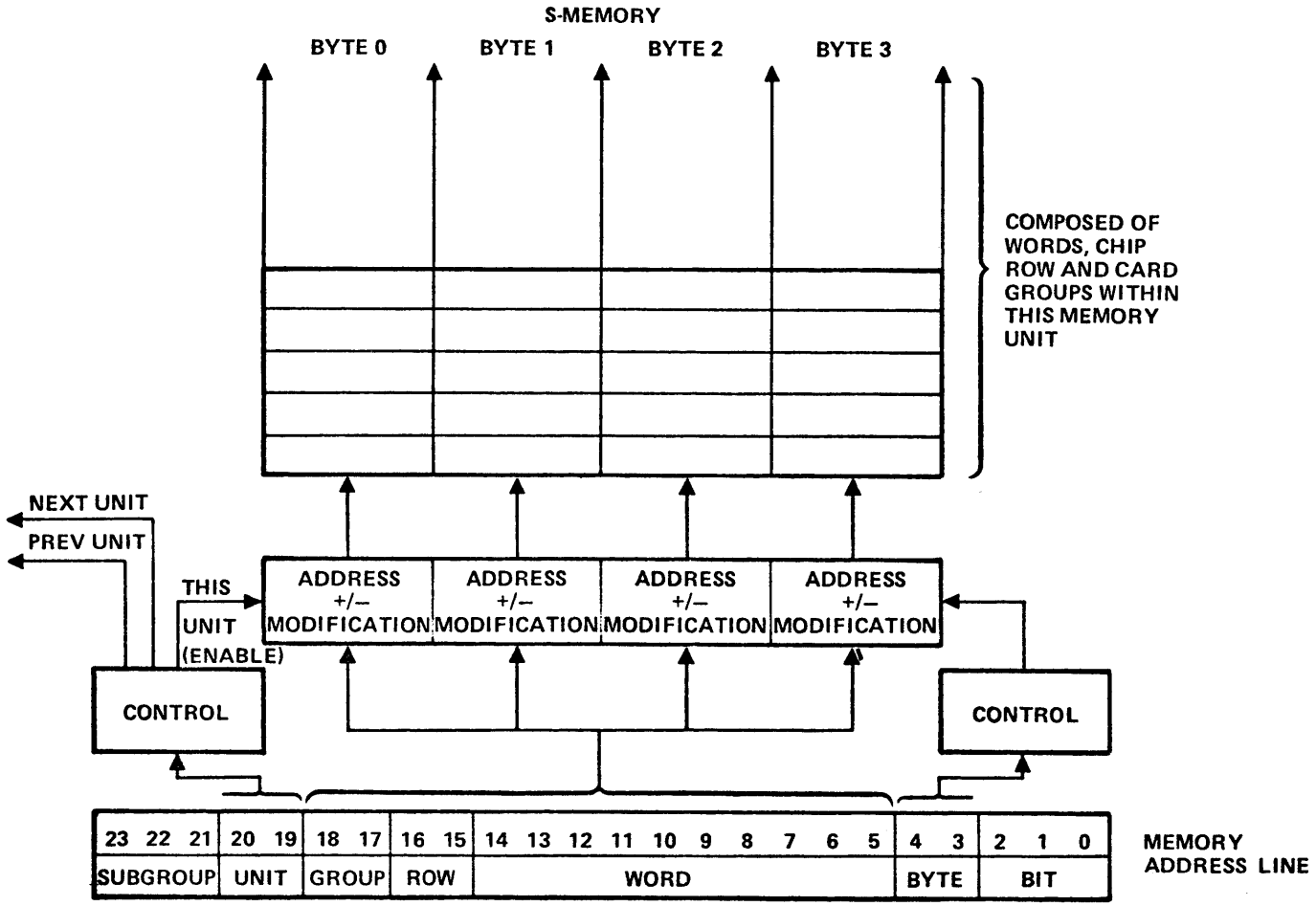


Figure 1-43. Field Address Card Address Modification and Distribution

Interface Control Logic

The interface control card contains gating logic to receive and distribute memory write data to the storage elements of memory, and to accumulate and gate memory read data from bytes 0, 1, and 2 of memory to the port interchange. The read data gates for byte 3 are located on the field address card, but are controlled from within the interface control logic, as are the others. In addition, the IC card generates the memory clock signals used for read and write timing, plus the precharge (PRE), chip enable (CE), and write enable (WE) signals which are used for control of the memory chips themselves.

I/O SUBSYSTEM

The B 1700 I/O Subsystem is comprised of an I/O base and I/O device controls, each of which serves as an interface between the I/O bus and one (or more) peripheral device(s). The controls are located close to the processor to minimize the length of the I/O bus which, in turn, minimizes propagation time.

I/O BUS

The Input/Output bus is a 24-bit wide, bi-directional bus which is used to carry either data or commands between the I/O subsystem and the processor. This bus is shared by all I/O controls attached to the processor. In conjunction with the 24-bit bus are control lines which define the operation as to phase (command active or response complete), and also as to the presence of data or commands on the bus. Refer to the detailed description of the I/O subsystem contained in section 2 of this manual.

I/O BASE

The I/O controls are installed in modules known as the I/O base and I/O base extension(s). These are packaged in a manner similar to the processor logic (see figure 1-44). Each control is constructed on one or more logic cards, and one to five of these controls may be installed in each I/O base or base extension. A maximum of fifteen I/O controls may be employed within the system (channel 16 is reserved). This would entail the installation of at least one I/O base and three extensions, depending on the number of card positions occupied by the individual controls.

The I/O base and extension(s) each contain a distribution card which interfaces it with the main I/O bus. The primary function of these cards is buffering, i.e., line receivers and drivers to minimize loading and reflection on the main I/O bus. The I/O base distribution card (only) provides the following additional functions: a) processes and distributes the system clock to its own controls (by way of the backplane) and to the distribution cards of the extensions (by way of coaxial cables) and, b) likewise generates and distributes the slow clocks.

The I/O bus is connected between the processor and I/O base distribution card by way of a strip cable. From this card, it is further connected to the distribution cards in the base extensions by other strip cables. Refer to the I/O Base Technical Manual, form no. 1053352) for further information.

I/O Controls

The I/O Controls used on the B 1720 Series System are special purpose controls, and each is unique. The controls, along with the devices with which they interface, are listed in table 1-5. In most cases a single control handles a single device. Each control is described in detail in its own I/O technical manual.

Figure 1-44. I/O Base

Table 1-5. B 1720 I/O Devices and Controls

DEVICE	MODEL	CONTROL NAME	NO. OF CARDS	SEPARATE BACKPLANE	SPECIFICATIONS
80 COLUMN CARD READERS	B9110	CARD READ CONTROL-2	1	NO	200 CPM
	B9111	CRC-2	1	NO	800 CPM
	B9112	CRC-2	1	NO	1400 CPM
	B9115	CRC-1	1	NO	300 CPM
	B9116	CRC-1	1	NO	600 CPM
	B9117	CRC-1	1	NO	800 CPM
80 COLUMN CARD PUNCHES	B9210	CARD PUNCH CONTROL-2	2	NO	100 CPM
	B9212	CPC-1	2	NO	150 CPM
	B9213	CPC-1	2	NO	300 CPM
96 COLUMN CARD READERS	B9119	MFCU-CONTROL	2	NO	300 CPM
	B9119-2	MFCUC	2	NO	1000 CPM
96 COLUMN MFCU	B9319-5	MFCUC	2	NO	1000/120/120 CPM
96 COLUMN CARD PUNCH	B9219-3	MFCUC	2	NO	120 CPM
96 COLUMN PUNCH/PRINT	B9219-4	MFCUC	2	NO	120/120 CPM
96 COLUMN READER/PUNCH	B9319-3	MFCUC	2	NO	500/120 CPM
96 COLUMN READ/PUNCH/PRINT	B9319-4	MFCUC	2	NO	500/120/120 CPM
96 COLUMN KEYPUNCH	B9419-1	MFCUC	2	NO	300/60 CPM
96 COLUMN KEYPUNCH/PRINT	B9419-2	MFCUC	2	NO	300/60/60 CPM
96 COLUMN KEYPUNCH/PRINT/SORT	B9419-6	MFCUC	2	NO	300/60/60 CPM

Table 1-5. B 1720 I/O Devices and Controls (Continued)

DEVICE	MODEL	CONTROL NAME	NO. OF CARDS	SEPARATE BACKPLANE	SPECIFICATIONS
96 COLUMN READ/PUNCH/PRINT	B9319-2	MFCUC	2	NO	300/60/60 CPM
PAPER TAPE READER	B9120	PAPER TAPE READ CONTROL	1	NO	
PAPER TAPE PUNCH	B9220	PAPER TAPE PUNCH CONTROL	1	NO	
PRINTERS	B9249-1	PRINT CONTROL-4	1	NO	86 LPM ODEC
	B9249-2	PC-4	1	NO	158 LPM ODEC
	B9245-4 thru 9	PC-3	1	NO	300/400 LPM CDC
	B9247-2-3	PC-2	1	NO	400/750 LPM
	B321-B329	PC-3	1	NO	700 LPM
	B9242	PC-1	1	NO	860 LPM UNBUFFERED
	B9243	PC-3	1	NO	860 LPM BUFFERED
READER SORTERS	B9134	READER SORTER CONT.-1	2	NO	1625 DPM SINGLE STATION
	B9134	RSC-2	4	YES	1625 DPM DUAL STATION
	B9131/32	RSC-3	2	NO	1560 DPM
	B9136-5	RSC-1	2	NO	600 DPM 8 POCKETS
	B9136-6	RSC-1	2	NO	600 DPM 12 POCKETS
MAGNETIC TAPES	B9391	MAGNETIC TAPE CONT.-1	6	YES	90 IPS 7TR 556/800 BPI
	B9392	MTC-2	6	YES	90 IPS 9TR 800 BPI
	B9394-1	MTC-1	6	YES	120 IPS 7TR 556/800 BPI
	B9394-2	MTC-2	6	YES	120 IPS 9TR 800 BPI

Table 1-5. B 1720 I/O Devices and Controls (Continued)

DEVICE	MODEL	CONTROL NAME	NO. OF CARDS	SEPARATE BACKPLANE	SPECIFICATIONS
OEM MAG. TAPE	B939-1-2-3	MTC-3	6	YES	90/120/150 IPS 9TR 1600 BPI
CLUSTERS	B9380	MTC-1	6	YES	36KB 7TR
	B9381	MTC-2	6	YES	36KB 9TR
	B9382	MTC-3	6	YES	72KB 9TR
	B9383	MTC-2	6	YES	36/72KB 9TR
OEM MAG. TAPE	B9391-2	MTC-2	6	YES	12.5 IPS 9TR 800 BPI
2 x 8 MAG. TAPE EXCH.	B9391/92 B9394-1-2	MTC - /MTC2 PLUS 2 x 2 ADAPTER	10	YES	1 ADAPTOR IS USED FOR EVERY TWO UNITS.
DISK CARTRIDGE	B9480-1-2	DISK CART. CONTROL-1	3	NO	SINGLE/DUAL 2200 BPI 203 TR
	B9481-1-2	DCC-1	3	NO	SINGLE/DUAL 2200 BPI 406 TR
	B9482-1-2	DCC-2	3	NO	SINGLE/DUAL 4400 BPI 406 TR
DISK FILES	B9272	DISK FILE CONTROL-1	4	YES	1 DFEU 1A 3 DFSU 1A-2
	B9373-3	DFC-1	4	YES	1 DFEU 1C 5 DFSU 1C-3
	B9375	DFC-1	4	YES	1 DFEU 1C 5 DFSU 1C-4
4 x 16 DISK FILE EXCHANGE-1	B9372/73 or 75	4 x DFC-1 PLUS 4 ADAPTERS	20	YES	16 DFEU's 48/80 DFSU's
DATA COMM.		SINGLE LINE CONTROL	4	YES	ST.AD. TC/TU/TD/DC
CONSOLE PRINTER (SPO)	B9340	CONT. - 1	1	NO	

POWER

Power for the B 1720 Series Central System is supplied by a minimum of three power supplies, all of which are contained within the central system cabinet. Two logic power supplies are always provided, as is one memory power supply for each memory base unit installed. The base units may number from one to four. Figure 1-45 illustrates the physical location of the supplies within the central system cabinet, as well as the ac power distribution assembly, dc power outlet assembly, fan assembly, and other basic assemblies of the central system.

POWER DISTRIBUTION

The basic distribution of ac power within the B 1720 Series Central System cabinet is as illustrated in figure 1-46. The primary power source normally required is single phase, 188 to 233 volts ac, 47 to 63 hertz. However, it is also possible to use other voltages in a three-phase configuration. Refer to section 6 for further information on this. The ac input is connected to the ac power distribution assembly as shown. Contained within the ac power distribution assembly is a 50-ampere circuit breaker and a 15-ampere circuit breaker. The 50-ampere breaker controls all power supplied to the central system, and from it are operated the logic power supplies, memory power supplies, and fan assemblies. The 15-ampere breaker supplies power to peripherals and convenience outlets. The 24-volt ac control contains additional relays which control application of power to the central system in response to the position of the main OFF/ON switch. Also included is a circuit for monitoring airflow through the card racks, with provision for automatic powering down if the power ventilation is lost. In addition, this module supplies a 24 volt ac, 50/60 hertz signal to the real time clock circuit, which is used to derive the square wave 100 millisecond signal which is its output.

LOGIC POWER SUPPLIES

The logic power supplies convert the ac input voltage to four output voltages used throughout the central system to operate the CTuL logic and the line drivers and receivers. The two main supplies operate in a master/slave relationship, with both of the high current outputs connected in parallel. The four output voltages are defined as follows. The current levels are also shown for each supply.

a. +4.75 volts: used as the Vcc of CTuL logic, with a nominal output current of 200 amperes.

b. -2.0 volts: used as the Vee of CTuL logic, with a nominal output current of 200 amperes.

c. +12.0 volts: used mainly as a supply voltage to line drivers, with a nominal output current of 6 amperes.

d. -12.0 volts: used mainly as a supply voltage to both line drivers and receivers, with a nominal output current of 18 amperes.

The logic power supplies are physically mounted in 19-inch RETMA rack cabinets which may be extended from the central system frame for ease of access when being serviced. When extended, they may also be rotated upwards 90 degrees for access to the underside. Refer to the B 1700 Logic Power Supply Technical Manual, form number 1070281, for further details.

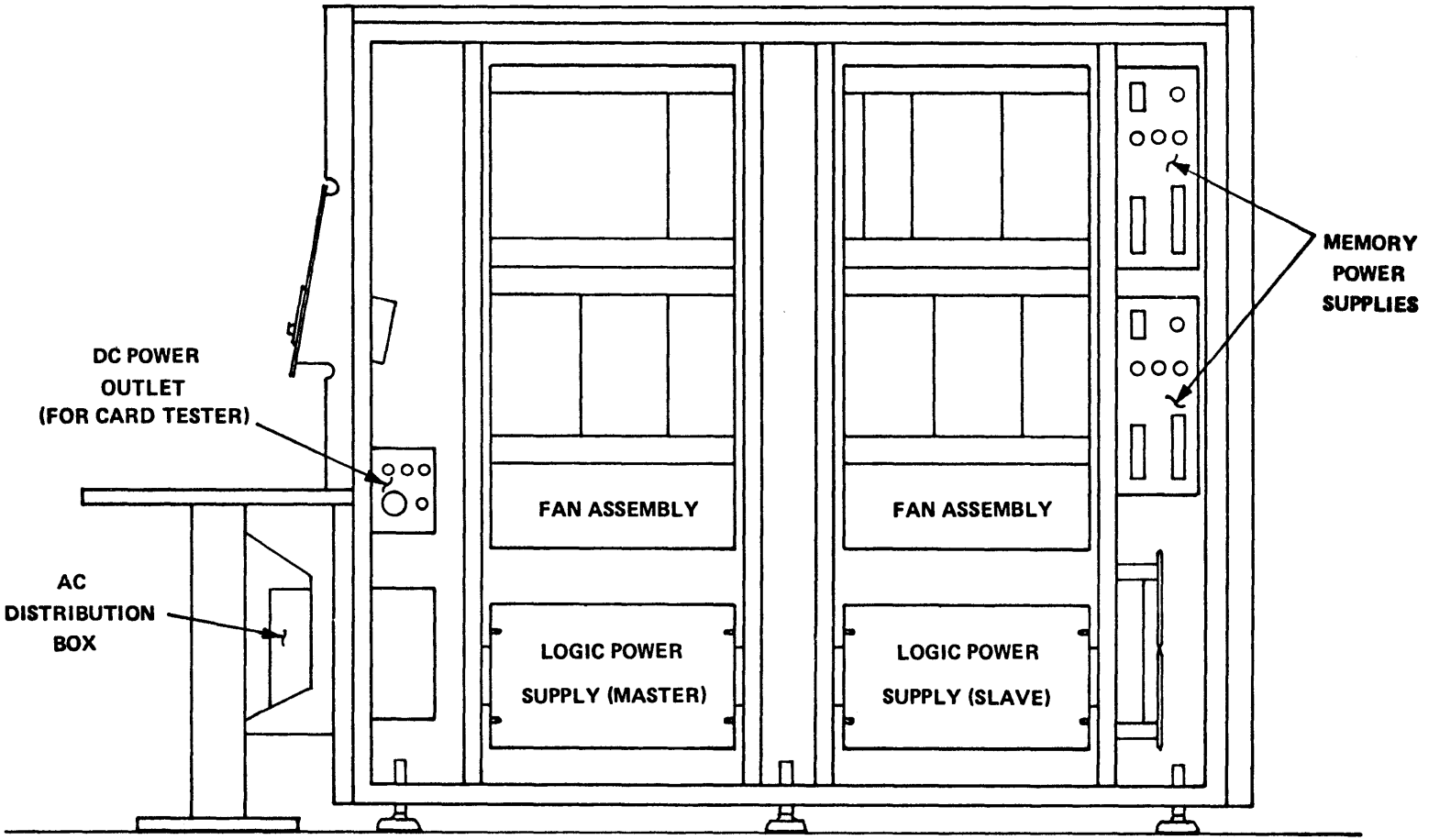


Figure 1-45. Central System Power Subassemblies

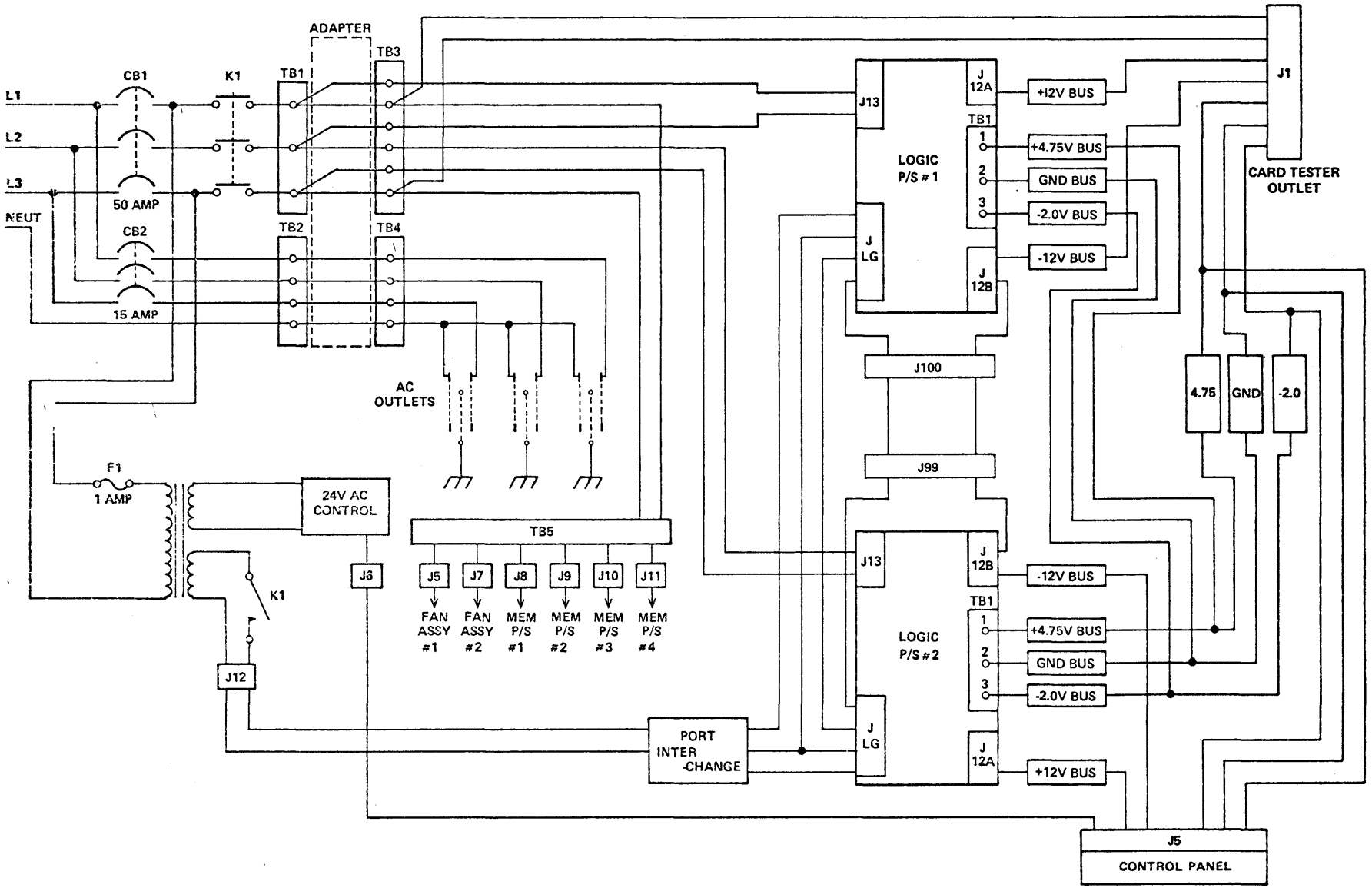


Figure 1-46. B 1720 Power Distribution

MEMORY POWER SUPPLIES

The memory power supplies convert the ac line voltage into the voltages required to operate the 1024-bit RAM memory chips (and the dual sense amplifiers which accompany them). Each supply provides the power necessary to the operation of one memory base unit, which can contain up to 64K bytes of data storage. The three output voltages are:

a. +19 volts for the Vss (source voltage) of the 1024-bit memory chips.

b. +23 volts for the Vbb (substrate voltage) of the 1024-bit memory chips.

c. -5 volts for the -Vref (negative reference voltage) of the dual sense amplifiers.

In addition to the above, the memory chips require -2 volts, and the dual sense amplifiers +4.75 volts. However, these voltages are provided by the logic power supplies and need not be duplicated in the memory supplies. Each of the three supply sections consists of a step-down transformer, full-wave rectifier, and capacitive filters. Also provided are voltage regulation and current limiting circuitry. Refer to section 2 for further details.

CENTRAL SYSTEM OPERATION

Operation of the B 1720 Series System is possible in either an on- or off-line manner. On-line operation implies that control of the system has been assumed by a program, with actions proceeding at the normal machine speed. Off-line operation, on the other hand, involves manual control by way of the console. Description of on-line operating procedures is beyond the scope of this manual, since they are fully documented in the applicable software publications. In particular, reference should be made to the B 1700 System Software Operational Guide, form number 1068731. However, off-line operation is provided primarily as a means of troubleshooting, and is, therefore, a necessary part of maintenance techniques. The following section describes the manner in which the console controls are to be used, and the machine capabilities which are available to the operator. Included are descriptions of each of the micro instructions, from which all operational control of the hardware emanates.

CONSOLE OPERATIONS

The console of the B 1720 Series System represents a physical interface with the micro decoding logic and main 24-bit exchange through which control of all system activities is exercised. Most of the controls create or modify a "wired" micro instruction to cause a desired action to occur. In most cases, these same actions can also be produced through execution of a normal software micro which is gated to the micro decoding logic by way of the M-register. The console switches therefore merely accomplish the same function manually, allowing for testing and demonstration of the hardware capabilities. In addition to the controls which provide a specific function, several are included for control of system operation. These include the START, CLEAR, and HALT switches, and the MODE control.

Selection of Operational Mode

Three operational modes are available for system operation: step, run, and tape. The step mode causes the hardware to fetch and execute one micro instruction each time the START button is pressed. The execution of this micro is normal in every way, except that the machine returns automatically to the halt state when it is completed. The step mode is used most often for troubleshooting.

The run mode is used for most normal systems operations. In it micro instructions are continuously fetched and executed until halted by the appearance of the halt micro or some other condition which requires that operations cease. During operation, control of the system is maintained by software. Note that the START button must always be used to initiate operation

in this or any other mode.

In the Tape mode, the cassette tape reader is used to transfer program data (micro instructions) into the Processor. These may be executed directly or stored in memory or registers for future use.

Register Selection

Selection of registers, in order that their contents may be viewed on the console lamps or the contents of the console switches may be loaded into them, is accomplished by way of the register group and REGISTER select switches. To select a register, the register group switch must first be rotated to the position corresponding to the group of four registers in which the desired one is located. When this has been done, the REGISTER select switch is then rotated to the number (0 through 3) which indicates that registers relative position within the group. This causes a hardware-forced 1C register move micro to be executed, with the source being the desired register and destination null, or the console lamps. It is important to note that this is possible only with the machine halted. Also, reading and writing in M-string memory and S-memory are selected in the same manner, although they are not registers.

Using the Console Lamps and Console Switches

The console lamps are provided to monitor the contents of the main exchange. As such they serve as both a visual indication of processor operation (when it is running), and allow the contents of any register or memory location to be examined when the processor is halted. The console switches allow manual selection of a 24-bit field configuration which can then be moved to any selected data storage location: registers, S-memory or M-string memory. As with examination of contents, this can only be done with the processor halted.

The significance of the various bit positions depends on the unit addressed. All registers are right-justified when displayed individually and will occupy the appropriate number of bit positions. Memory read/write data, on the other hand, is left-justified because the most-significant bit position represents the beginning address of the data field. In all cases, however, each individual console switch corresponds directly to the console lamp directly above it, and will, when raised, cause a 1 bit to be set in that position. Note, however, that 1 bits thus selected are not gated to the chosen register or memory location until this is specifically effected by initiating a load or write cycle. The binary weight of each of the 24 console switch/console lamp bit positions is shown in figure 1-47. Note that the binary weights shown are valid only as regards the 24-bit field taken as a whole. Quite often smaller fields within the 24 bits are assigned individual meanings with accompanying (smaller) binary weights. Refer to the register and micro descriptions for details on these specialized meanings.

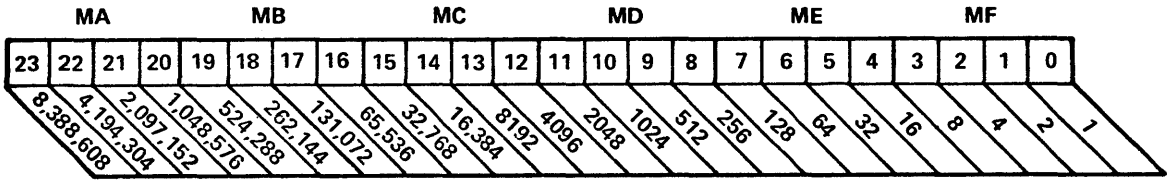


Figure 1-47. Console Switch/Lamp Binary Weights

Reading a Register's Contents

This involves simply selecting the desired register by way of the REGISTER select and register group switches, with the processor halted. A hardware-forced 1C register move micro is created in doing so, causing the contents of the selected register to be gated to the main exchange, and from there latched into the console lamp register from which it is continuously displayed. The act of selection causes this to occur, and no further action is necessary. Refer to the register descriptions for the specific meaning of the contents.

Loading a Register

To load a register, the selection procedure as described in register selection above must first be performed. When this has been done, the desired bit pattern should be set on the console switches. Pressing the LOAD pushbutton causes a hardware-forced 1C register move micro to be executed, gating the contents of the switches to the register. Note that the entire contents of the register are affected by the loading process. If it is desired to save any portion, these bits must be re-inserted by use of the console switches.

Reading in S-Memory

To read, or cause the data stored at a specific location in S-memory to be displayed on the console lamps, the bit address desired must first be loaded into the FA (field address) register. The binary weight of this code is illustrated in figure 1-48. After this preparatory step, select READ on the REGISTER select and register group switches and press the READ/WRITE pushbutton. Doing so initiates a memory cycle, with the stored data being gated to null, and thus displayed on the console lamps. Note that performing a read does not destroy the data stored at the accessed location. Once a location has been accessed, the FA address can be incremented by 16 bits (one word) by depressing the INC pushbutton. This may be done repeatedly, and be monitored by selecting FA. However, once the register group switch has been moved from the READ position, it is necessary to again press READ/WRITE when returning to it.

MA				MB				MC				MD				ME				MF				
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	M REG BIT
X	X	X	64K	32K	16K	8K	4K	2K	1K	512	256	128	64	32	16	8	4	2	1	X	X	X	X	FA BINARY WEIGHT (WORDS)
																			16	8	4	2	1	FL BINARY WEIGHT (BITS)

NOTES: 1 MAXIMUM FL VALUE IS 24

2 FA ADDRESSES ABOVE 512 ARE AS FOLLOWS

1K = 1024
2K = 2048
4K = 4096
8K = 8192
16K = 16384
32K = 32768
64K = 65536

3 X = BIT ADDRESS. MAY BE USED IF DESIRED.

Figure 1-48. FA and FL Values

Writing in S-Memory

To write, or cause the bit configuration selected on the console switches to be gated to S-memory, the desired bit address must first be loaded into the FA register. This procedure is identical to that performed before a read, and requires that the console switches be used to enter the address code first. Once this has been done, the bit configuration to be written in memory is selected on the switches. Note that all 24 switches are active, thus requiring that the desired state be selected for each bit position. Therefore, if it is desired to retain a certain portion of the existing stored data, these bits must be reset on the switches. The data gated from the console switches displaces the previous contents of that address in memory. The INC button operates in the same manner as for a read. However, it is necessary to press READ/WRITE for each write operation.

Reading in M-String Memory

Reading in M-string memory is similar to reading the contents of any register, except that the desired address must first be placed in the A- (address) register. Refer to figure 1-49. Select MSM after loading A. As with S-memory, the address may be incremented by use of the INC button.

MA				MB				MC				MD				ME				MF				
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	M REG BIT
X	X	X	X	X	X	X	4K	2K	1K	512	256	128	64	32	16	8	4	2	1	X	X	X	X	A ADDRESS

NOTES:
1. X = NO SIGNIFICANCE, ALL A VALUES ARE WORD ADDRESSES, SINCE BIT ADDRESS ABILITY IS NOT PROVIDED IN MSM.
2. VALUES IN EXCESS OF 4K (4096) ARE NOT POSSIBLE, SINCE THIS IS THE MAXIMUM SIZE OF MSM.
3. NO FIELD LENGTH INFORMATION IS REQUIRED, SINCE ALL MICROS ARE CONSIDERED TO BE 16 BITS LONG.

Figure 1-49. A-Register Values

Writing in M-String Memory

Writing in M-string memory is similar to writing in S-memory. The desired address is placed in the A-register, then MSM is selected. Set the desired bit configuration on the console switches, and press LOAD to initiate the memory cycle. Note that only the right-most 16 console switches are active. The address may be incremented by use of the INC button.

Cassette Operation

The cassette is used for input of test routines and short "bootstrap" programs. It is usable in both the tape and run modes of the processor. In the tape mode, micros are executed directly from it; in run mode, the cassette generally serves as a source for loading micros in M-string. Cassette operation is always begun in the tape mode. To start, insert the desired tape and close the door. Make sure that the proper side of the tape is facing outwards. This is usually indicated by the label. Tapes which may be run from either end are

marked BOTH SIDES. Before running the tape, check to ensure that it is in the starting position. This is indicated by the BOT (beginning of tape) lamp, which should be lit. If it is not lit, press REW and wait until the tape stops and BOT illuminates. When ready, press CLEAR, then START. Tape motion should begin. The directions given in the documentation accompanying the tape should be followed from this point. However, the usual procedure is to wait until the processor halts (as indicated by extinguishing the RUN lamp and ceasing of tape motion), then select the run mode and again press START (do not press CLEAR). Cassette operation from this point is controlled by the execution of micros within the program being used. Most programs utilize the cassette for only a short period of time, after which it may be rewound and the tape removed. The program documentation should be consulted for further information. Rewind must be performed manually by the operator, and does not interfere with processor operation in any way.

MICRO OPERATORS

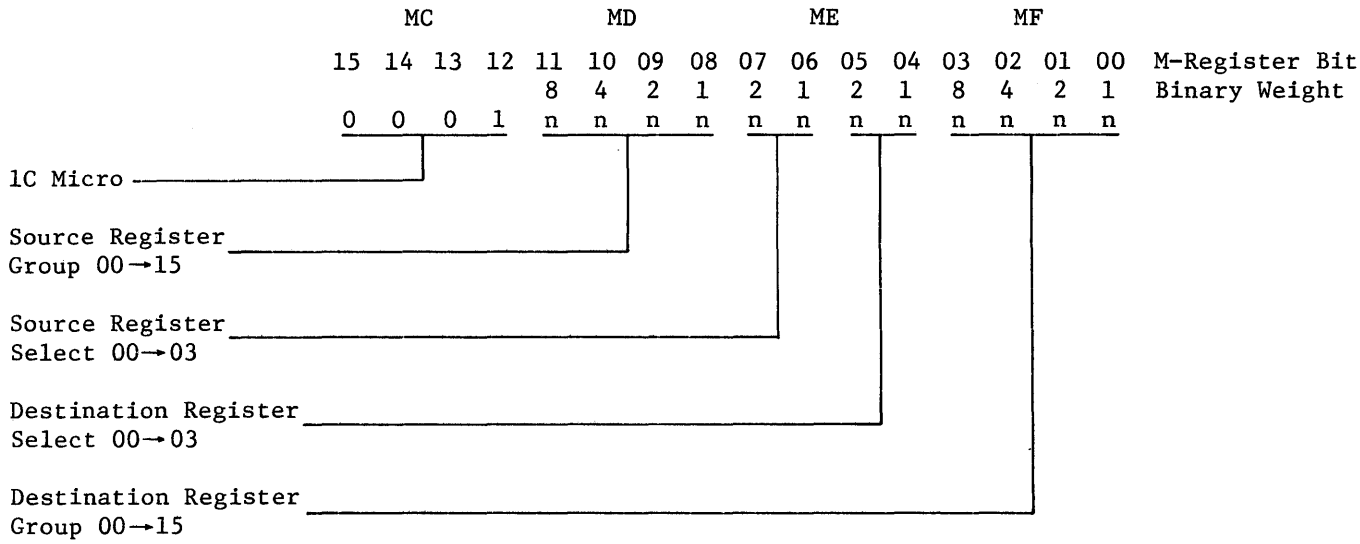
The B 1720 Series Micro Processor has the capability of producing approximately 60 different micro instructions. It was necessary to develop only 33 of the possible 60 to produce the necessary data processing functions. The 33 micros which have been implemented are listed in figure 1-50. The micros are referred to by a name which describes their function and a number indicating the binary weight in the specific subregister of M where the micro begins. For example, the 1C register move micro has a binary weight of 1 in the MC register. This binary weight in the most-significant 4-bit field is the distinguishing characteristic by which micros may be identified in the absence of other indications. Note that although all micros comprise a 16-bit field, the most significant portion is the left-most 4-bit subgroup which contains a binary weight other than zero. The remaining bits (if any) to the right of this 4-bit identifier contain coded register selection data, variants, or literals which modify the function of the basic micro. These are shown in figure 1-50, but for a more detailed explanation, refer to the individual micro operator descriptions which follow (figures 1-51 through 1-82). It is important to note that the use of micro operators is a significant aid to troubleshooting and repair work. The execution of selected micros allows suspected faulty circuits to be exercised with their normal operational functions while being monitored with external test equipment. Test micros may be executed singly (in the step mode) by addressing the M-register from the console and loading the desired micro with the 24 console switches. The micro executes when START is pressed, and any desired number of repetitions may be obtained by alternately pressing LOAD and START. To obtain repeated execution of selected micros at machine speed (in the run mode), a small program consisting of the desired micros may be manually written into MSM. This micro string should conclude with a "move literal (8C or 9C) to A" micro which causes the hardware to return to the beginning address at the completion of the string, thus forming a loop. Any desired number of micros may be included in such a string. Note that any of several micro types may be used to form a loop at the end of a string. These include the following:

1C: move null to A if the string begins at address zero.

8C/9C: move beginning address of string to A.

13C: decrement A to beginning address of string.

Of these, the 8C or 9C is probably the easiest to use.



Move the contents of the source register to the destination register. If the move is from a smaller register to a larger one, the data is right-justified with left (most significant) zero bits supplied. Moves from larger registers to smaller ones are also right-justified, but with the excess most-significant bits truncated.

The contents of the source register are unchanged unless it is also the destination register. The basic execute time is two clocks to which is added one additional clock if the source is BCD, and two additional clocks if the destination is the MAR(A) register. If the U-register is the source, the time takes many clocks. The following are exceptions:

a. CPU and CMND are excluded as source registers,

b. When M is used as a destination register, the operation is changed to a bit OR which modifies the next micro-operation. It does not modify the instruction as stored in the memory.

c. BICN, FLCN, XYCN and XYST are excluded as destination registers.

d. All registers and pseudo registers in column select=3 are excluded as destination registers except CMND, DATA, and NULL.

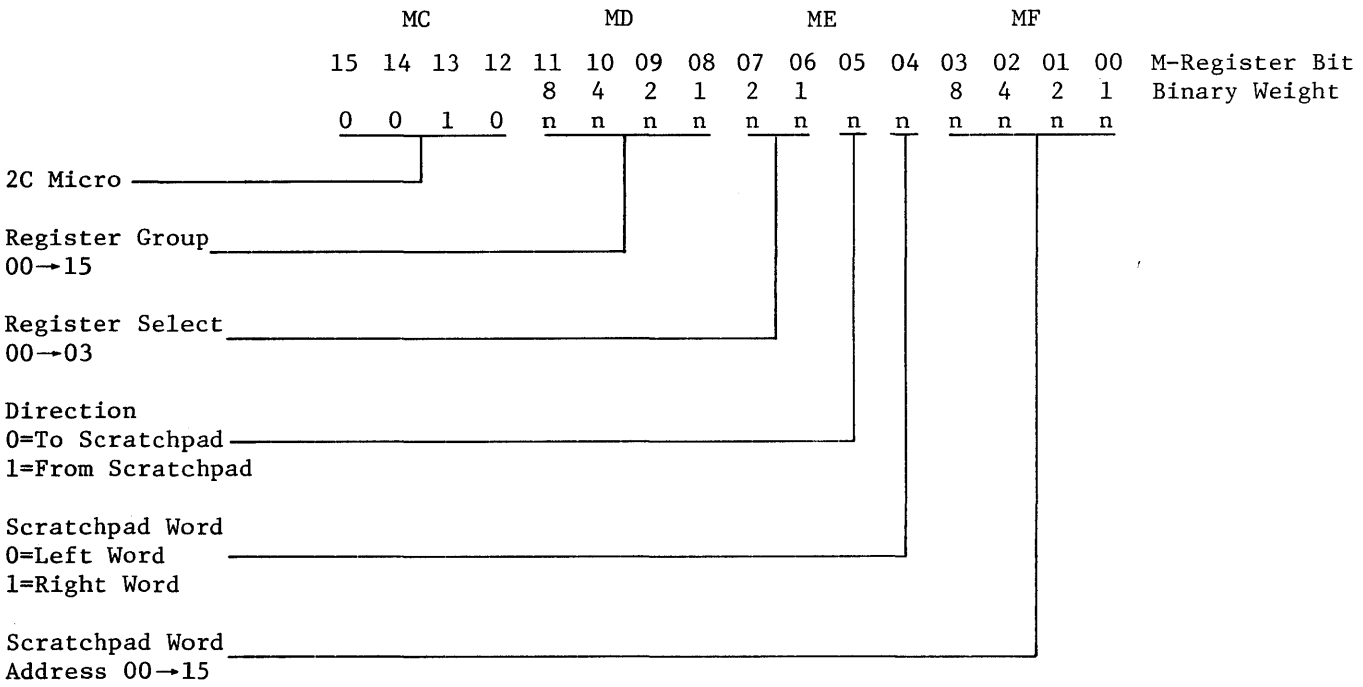
e. When DATA is designated as a source, CMND and DATA are prohibited as destinations.

f. U is excluded as a source register in step and tape mode. It is permitted as source in run mode. However, when U is used as a source the TAS, A- and M-registers are excluded as destination registers.

g. When CMND or DATA is designated as a destination, SUM and DIFF are prohibited as sources.

h. When A, M, CP or DATA is designated as a source, all 4-bit registers are prohibited as destinations.

Figure 1-51. 1C Register Move



Move the contents of the register (scratchpad) to the scratchpad (register). If the move is between registers of unequal lengths, the data is right-justified with left (most significant) zero bits supplied or with data truncated from the left, whichever is appropriate.

The contents of the source register are unchanged. The basic execute time is two clocks, to which is added one additional clock if the source is BCD and two additional clocks if the destination is the MAR(A) register. The following are exceptions:

a. U, CPU and CMND are excluded as source registers.

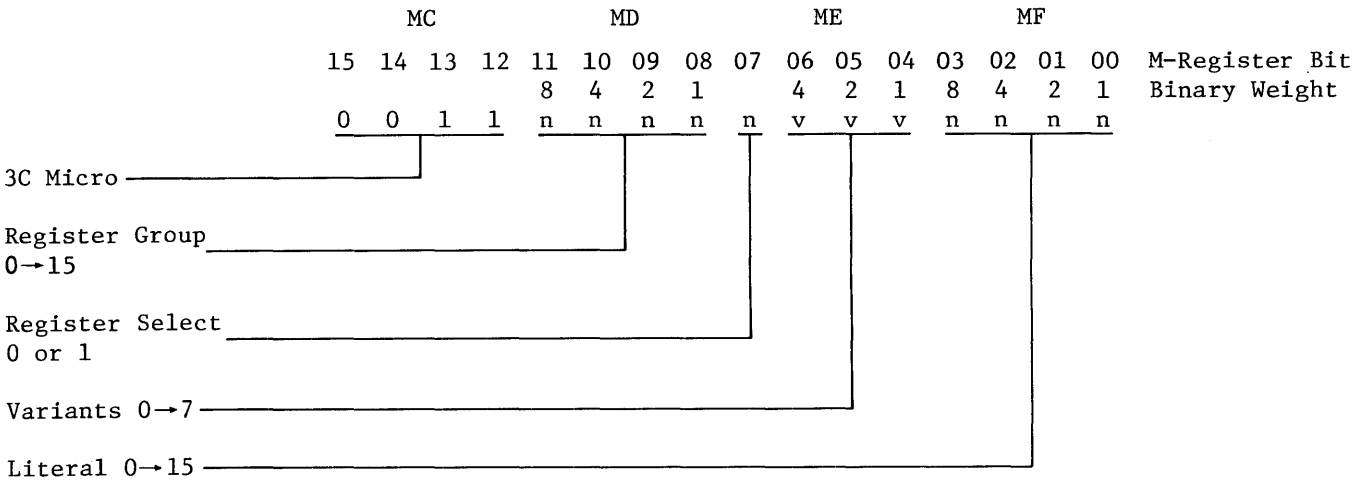
b. When M is used as a destination register, the operation is changed to a bit OR which modifies the next micro-operation. It does not modify the instruction as stored in the memory.

c. BICN, FLCN, XYCN, and XYST are excluded as destination registers.

d. All register and pseudo registers in column select=3 are excluded as destination registers except CMND and DATA.

e. M as a source would result in a transfer of 24 zeroes.

Figure 1-52. 2C Scratchpad Move



Perform the manipulate operation as specified by the variants on the addressed 4-bit register.

Variant	Name	Action
000=0	SET	Reg:=Lit
001=1	AND	Reg:=Reg*Lit
010=2	OR	Reg:=Reg+Lit
011=3	EOR	Reg:=Reg+Lit
100=4	INC	Reg:=Reg plus Lit (disregard overflow)
101=5	INC T	Reg:=Reg plus Lit (skip on overflow) T=test
110=6	DEC	Reg:=Reg-Lit (disregard underflow)
111=7	DEC T	Reg:=Reg-Lit (skip on underflow) T=test

V VARIANT DESCRIPTIONS

0 Set the register to the value of the literal.

1 Set the register to the logical AND of the register and literal.

2 Set the register to the logical OR of the register and literal.

3 Set the register to the logical Exclusive OR of the register and literal.

4 Set the register to the binary sum modulo 16 of the register and literal.

5 Set the register to the binary sum modulo 16 of the register and literal and skip the next M-instruction if a carry is produced.

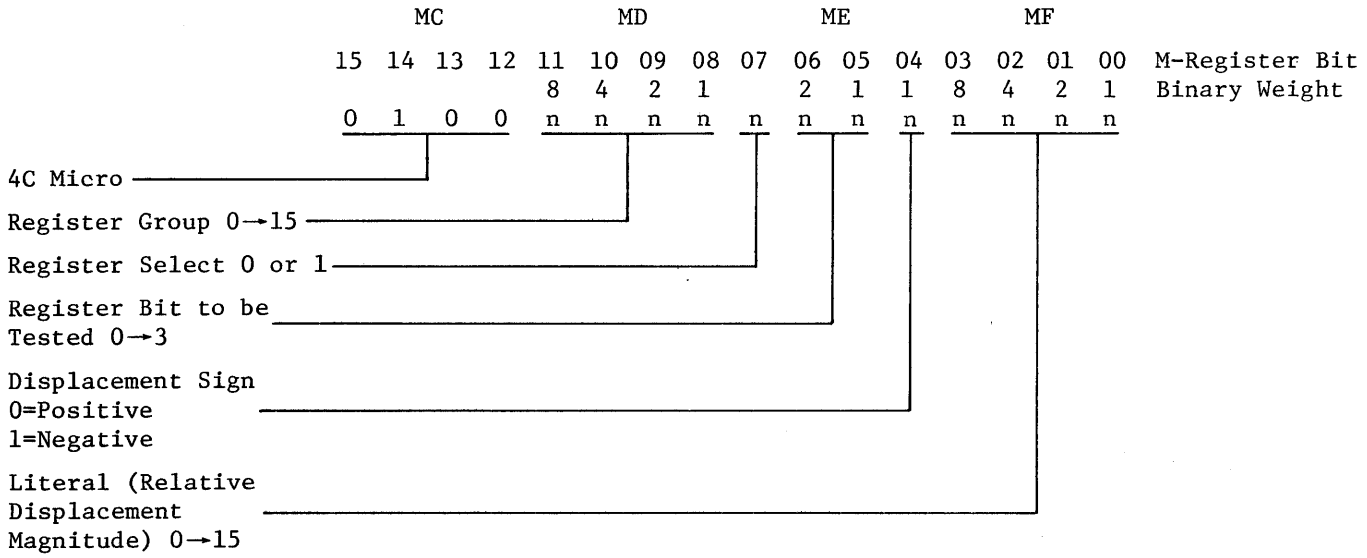
6 Set the register to the binary difference modulo 16 of the register and literal.

7 Set the register to the binary difference modulo 16 of the register and literal and skip the next M-instruction if a borrow is produced.

The basic execute time is two clocks to which is added two additional clocks if the branch is taken. The following is an exception:

a. BICN, FLCN, XYCN, XYST, and CPU are excluded as source/destination registers.

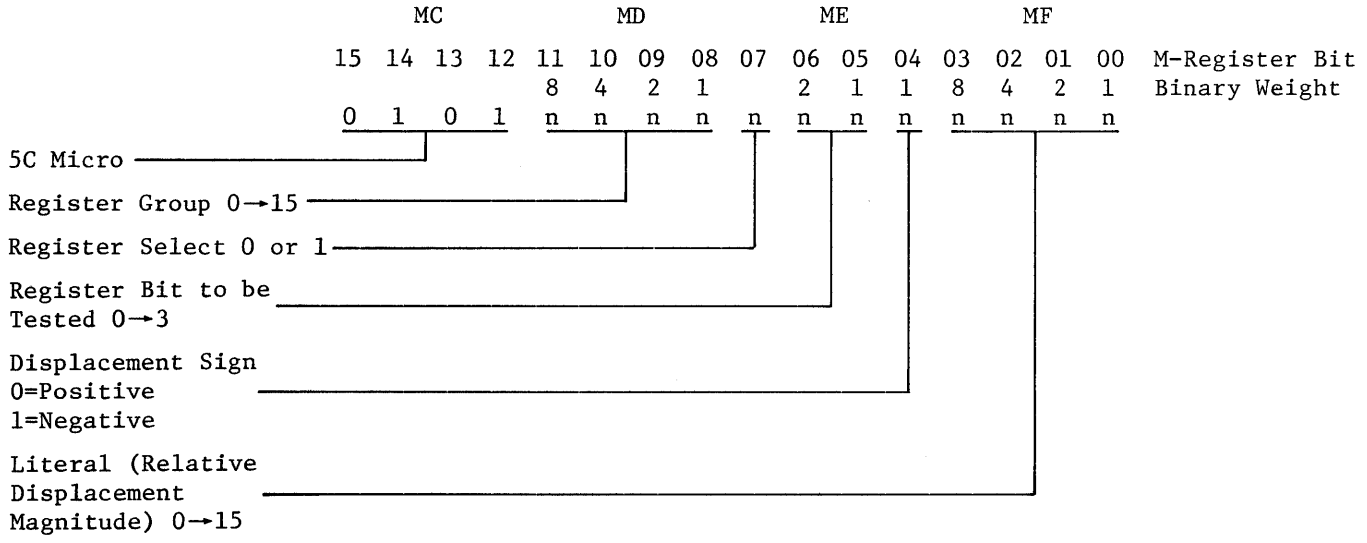
Figure 1-53. 3C 4-Bit Manipulate



Test the designated bit within the specified register and branch relative to the next instruction by the signed displacement value if the bit is zero. If the bit is one, a displacement value of zero is assumed and control passes to the next in-line M-instruction. A displacement value indicates the number of 16-bit words from the next in-line instruction.

The basic execute time is two clocks to which is added an additional clock if the source is BCD (BICN with CPU=01) and two additional clocks if the branch is taken.

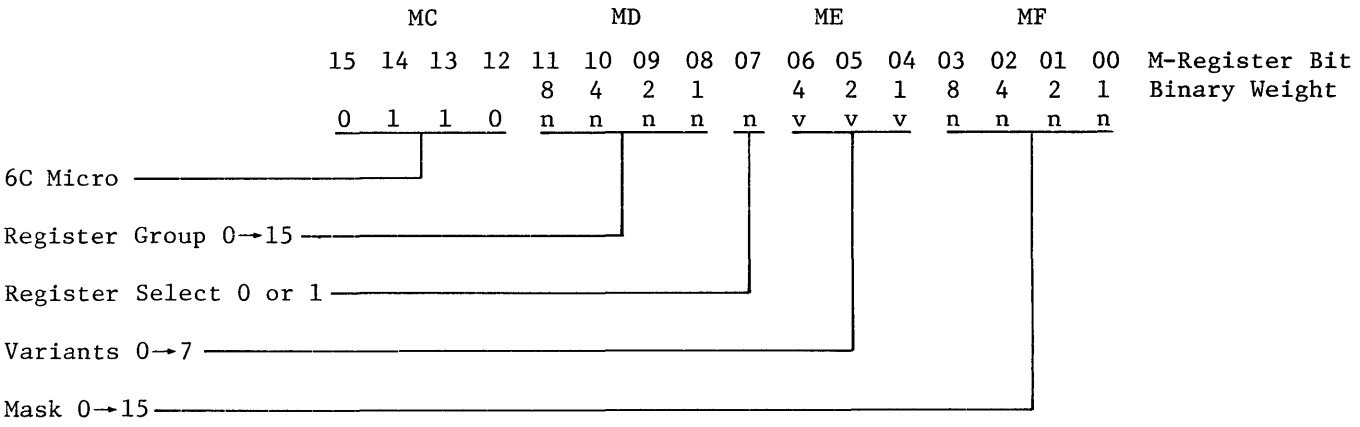
Figure 1-54. 4C Bit Test Relative Branch False



Test the designated bit within the specified register and branch relative to the next instruction by the signed displacement value if the bit is one. If the bit is zero, a displacement value of zero is assumed and control passes to the next in-line M-instruction. A displacement value indicates the number of 16-bit words from the next in-line instruction.

The basic execute time is two clocks to which is added one additional clock if the source is BCD (BICN with CPU=01) and two additional clocks if the branch is taken.

Figure 1-55. 5C Bit Test Relative Branch True



Test only the bits in the register that are referenced by the "1" bits in the mask (ignoring all others)and then perform the action as specified below. The following is an exception:

a. If V=2 or V=6, compare all bits for an equal condition.

VARIANTS

V=0 If any of the referenced bits is a "1", skip the next M-instruction.

1 If all of the referenced bits are "1", skip the next M-instruction.

2 If the register is equal to the mask, skip the next M-instruction.

3 Same as V=1, but also clear the referenced bits to zero without affecting the non-referenced bits.

4 If any of the referenced bits is a "1", do not skip the next M-instruction.

5 If all of the referenced bits are "1", do not skip the next M-instruction.

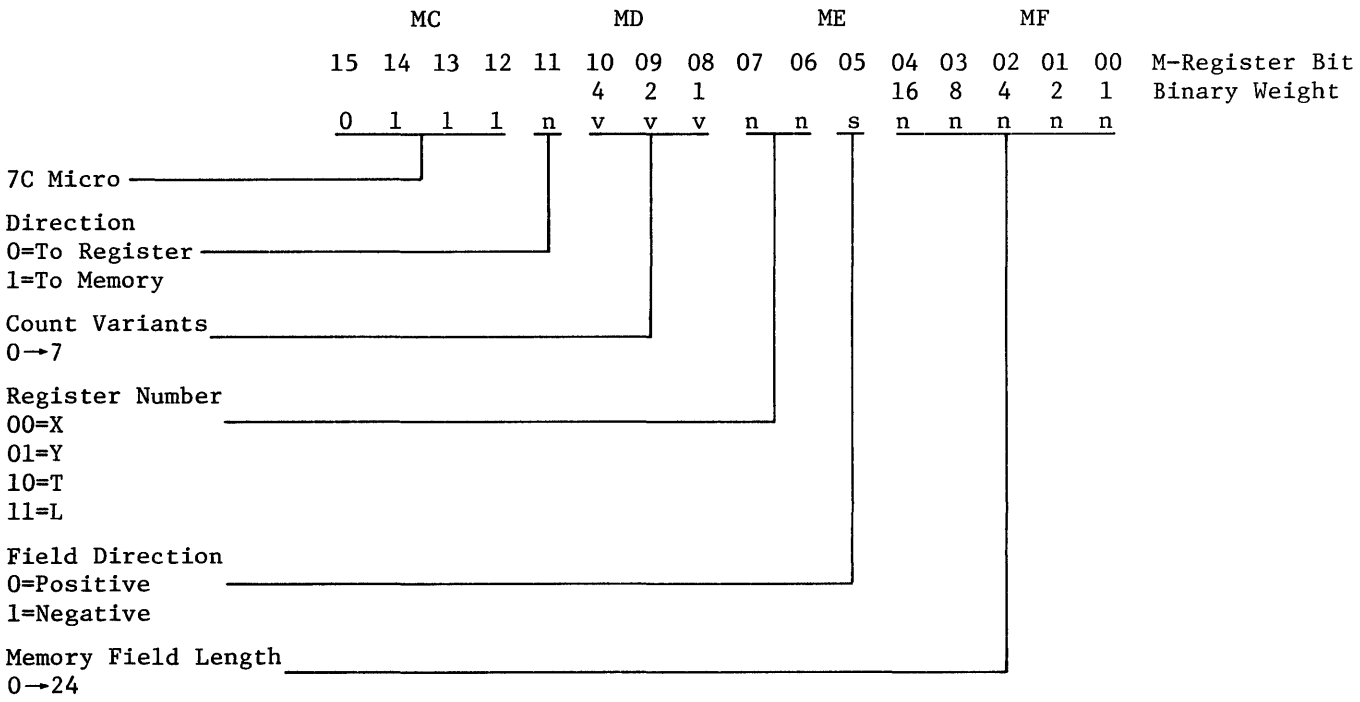
6 If the register is equal to the mask, do not skip the next instruction.

7 Same as V=4 but also clear the referenced bits to zero without affecting the non-referenced bits.

The basic execute time is two clocks to which is added one additional clock if the source is BCD (BICN with CPU=01) and two additional clocks if the branch is taken. Note that if the mask equals 0000 the ANY result is false. The skip is not made for V=0 and is made for V=4. If the mask equals 0000, the ALL result is true. The skip is made for V=1 and V=3 and is not made for V=5 and V=7. The following are exceptions:

a. BICN, FLCN, XYCN, and XYST are excluded as operand registers when V=3 or when V=7.

Figure 1-56. 6C Skip When



Move the register's (memory's) contents to the memory (register). If the value of the memory field length is less than 24, the data from memory is right-justified with left (most significant) zero bits supplied, while the data from the register is truncated from the left.

The contents of the source are unchanged. Register FA contains the bit address of the memory field while the memory field direction sign and memory field length are given in the instruction.

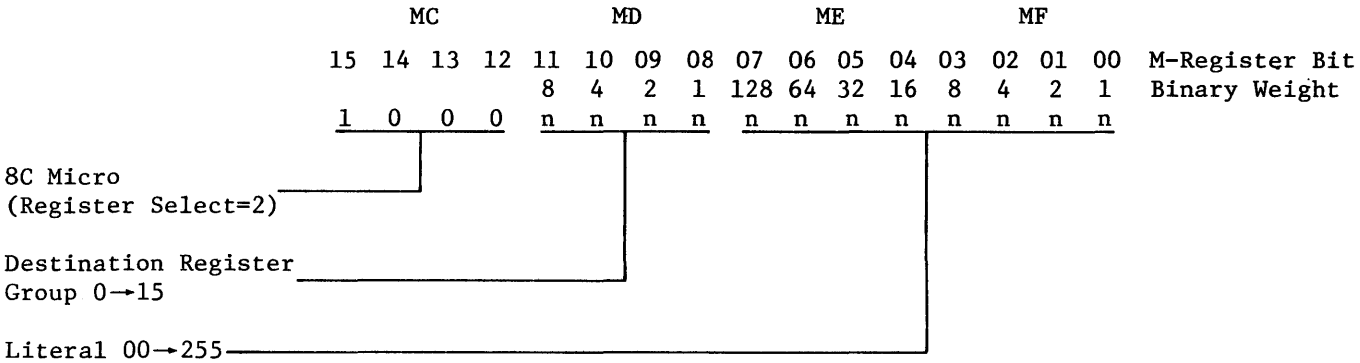
If the value of the memory field length as given in the instruction is zero, the value in CPL is used.

COUNT VARIANTS

V=000 No count
001 Count FA up
010 Count FL up
011 Count FA up and FL down
100 Count FA down and FL up
101 Count FA down
110 Count FL down
111 Count FA down and FL down

Execute time equals eight clocks.

Figure 1-57. 7C Read/Write Memory

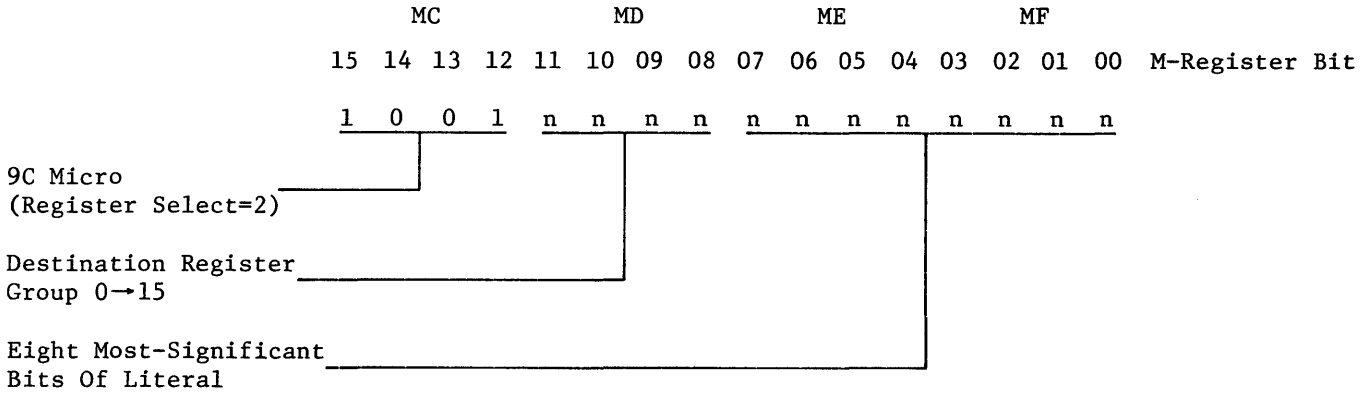


Move the 8-bit register literal, given in the instruction, to the destination register. If the move is to a register that is greater than 8 bits in length, the data is right-justified with left (most significant) zero bits supplied.

Only registers X, Y, T, L, MAR(A), BR, LR, FA, FB, FL, TAS, and CP can be specified. The register select is assumed to be 2.

The basic execute time is two clocks to which are added two additional clocks if the destination is the MAR(A) register.

Figure 1-58. 8C Move 8-Bit Literal

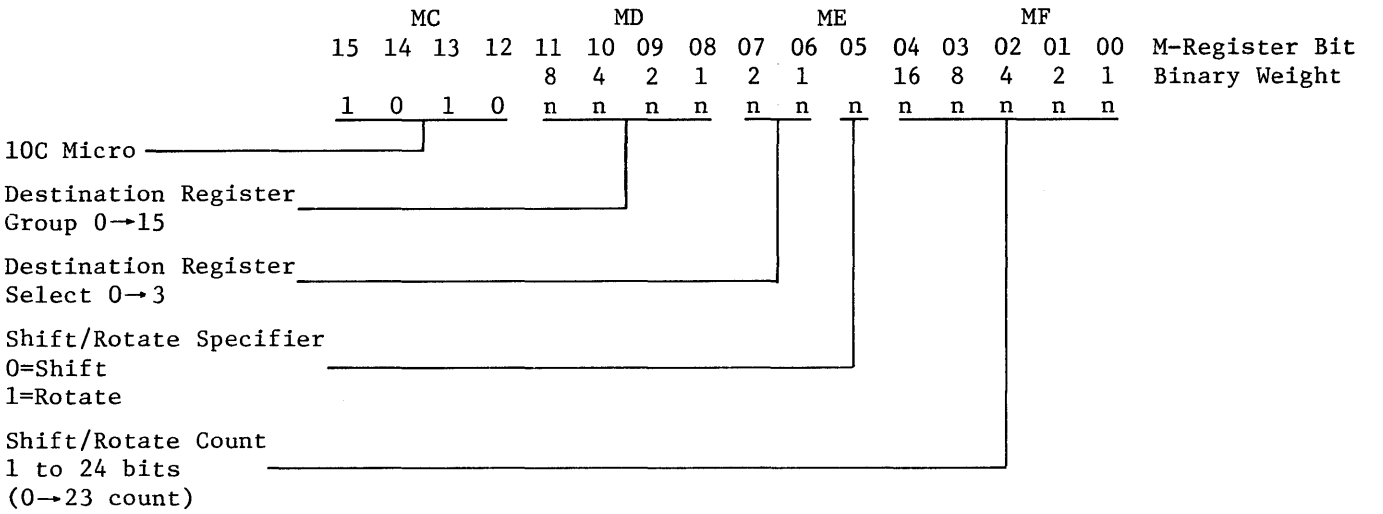


Move the 24-bit literal given in the instruction to the destination register. The 16 least-significant bits of the literal are stored in the position of the next in-line micro instruction. If the move is between registers of unequal lengths, the literal is truncated from the left.

Only registers X, Y, T, L, MAR(A), BR, LR, FA, FB, FL, and TAS can be specified. The register select number is assumed to be 2.

Execution time equals six clocks.

Figure 1-59. 9C Move 24-Bit Literal



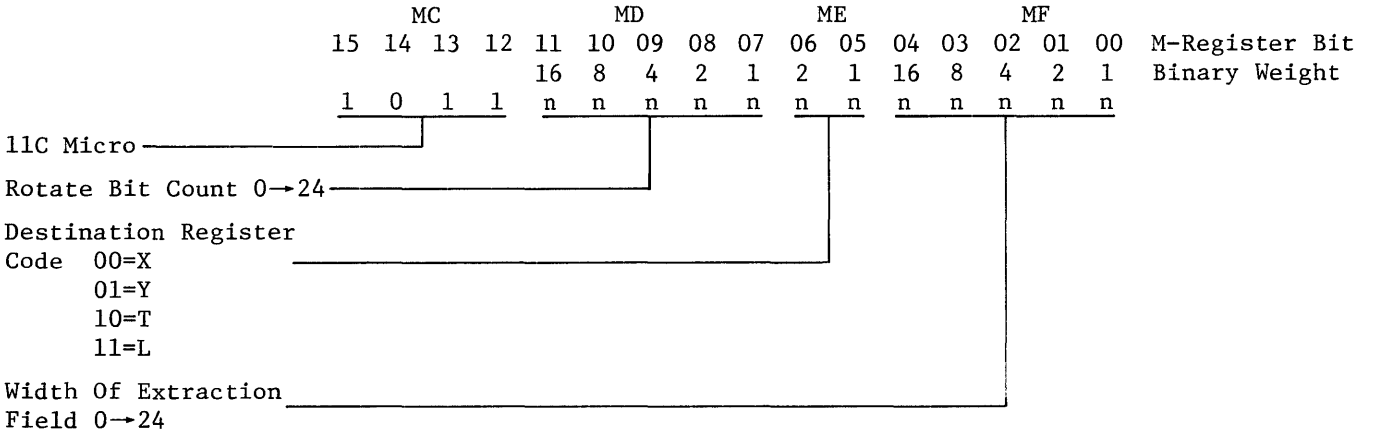
Shift (rotate) register T left by the number of bits specified and then move the 24 bit result to the destination register. If the move is between registers of unequal lengths, the data is right-justified with data truncated from the left.

The contents of the source register is unchanged unless it is also the destination register. Zero fill on the right and truncation on the left occurs for the shift operation. If the value of the shift/rotate count as given in the instruction is zero, the value given in CPL is used.

The basic execute time is three clocks to which are added two additional clocks if the destination is MAR(A). The following are exceptions:

a. When M is used as a destination register, the operation is changed to a bit OR which modifies the next micro-operation. It does not modify the instruction as stored in the memory.
b. BICN, FLCN, XYCN, and XYST are excluded as destination registers.

Figure 1-60. 10C Shift/Rotate T-Register Left

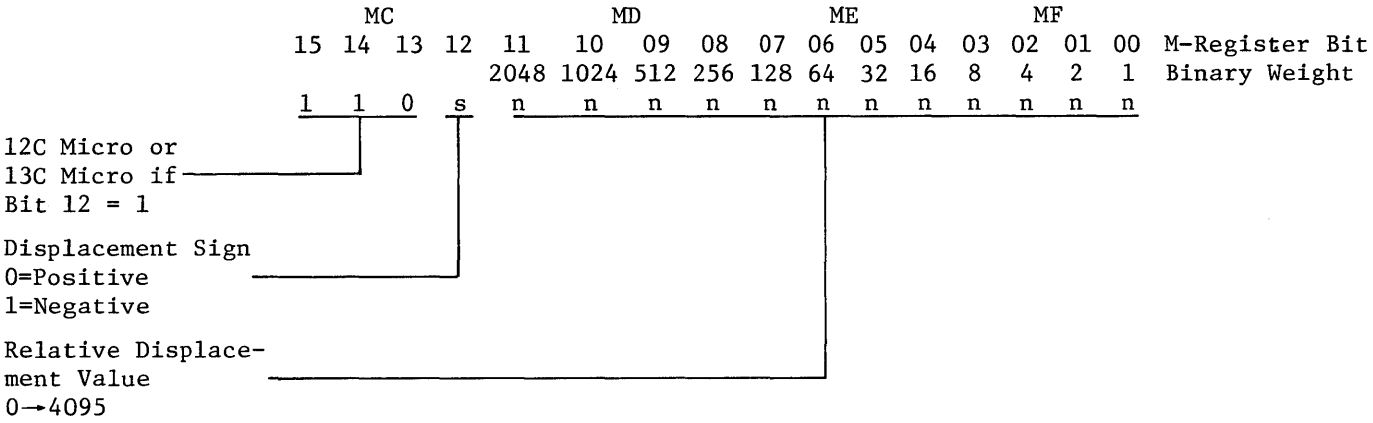


Rotate register T left by the number of bits specified and then extract the number of bits specified. Move the result to the destination register. If the extract bit count is less than 24, the data is right-justified with left (most significant) zero bits supplied.

The contents of the source register are unchanged unless it is also the destination register. A rotate value of 24 is equivalent to 0.

Execute time equals three clocks.

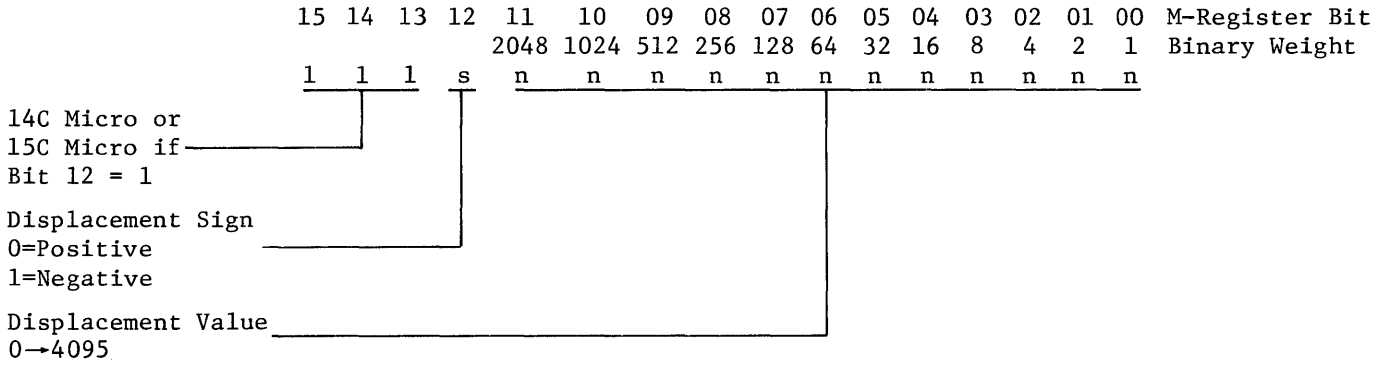
Figure 1-61. 11C Extract from T-Register



Fetch the next micro-instruction from the location obtained by adding the relative displacement value given in this micro instruction to the word address of the next in-line micro-instruction. A displacement value indicates the numbers of 16-bit words. Affects the contents of the A-register only.

Execution time equals four clocks.

Figure 1-62. 123C (12C, 13C) Branch Relative

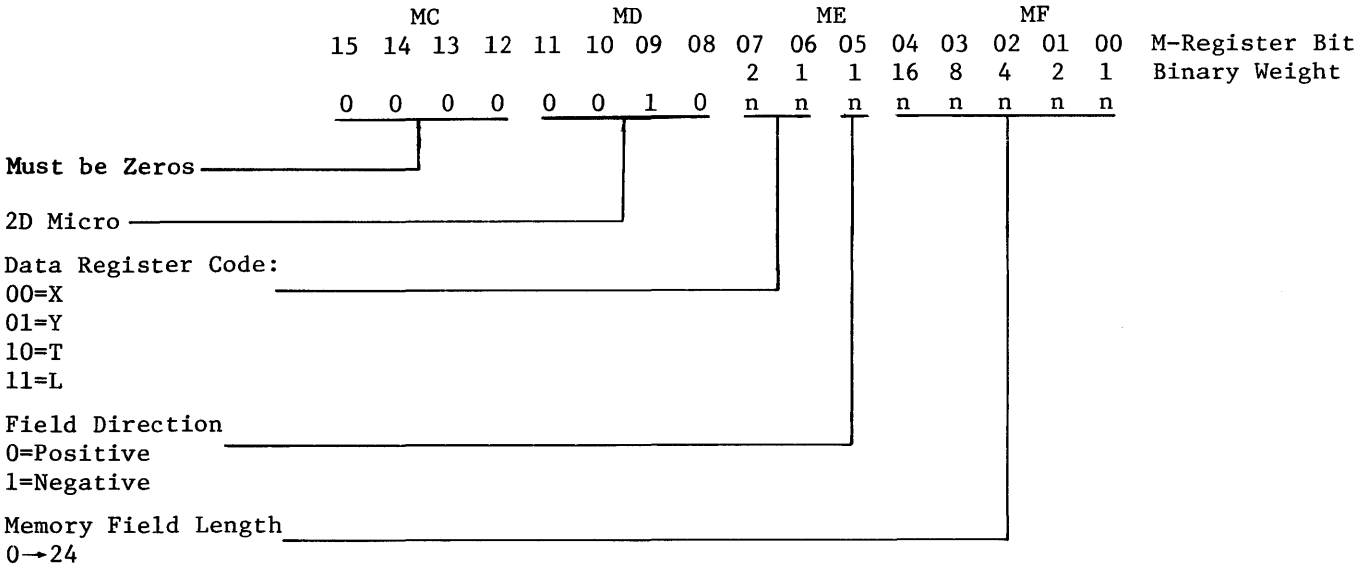


Push the address of the next in-line micro-instruction into the A-stack and then fetch the next micro-instruction from the location obtained by adding the signed relative displacement value given in this micro instruction to the word address of the next in-line micro-instruction. The displacement value indicates the number of 16-bit words. Affects the contents of the A-register only.

Note that exit is accomplished by employing the move register instruction with the TAS as the source register and as the destination register.

Execution time equals five clocks. This time includes the time to fetch the next micro-instruction.

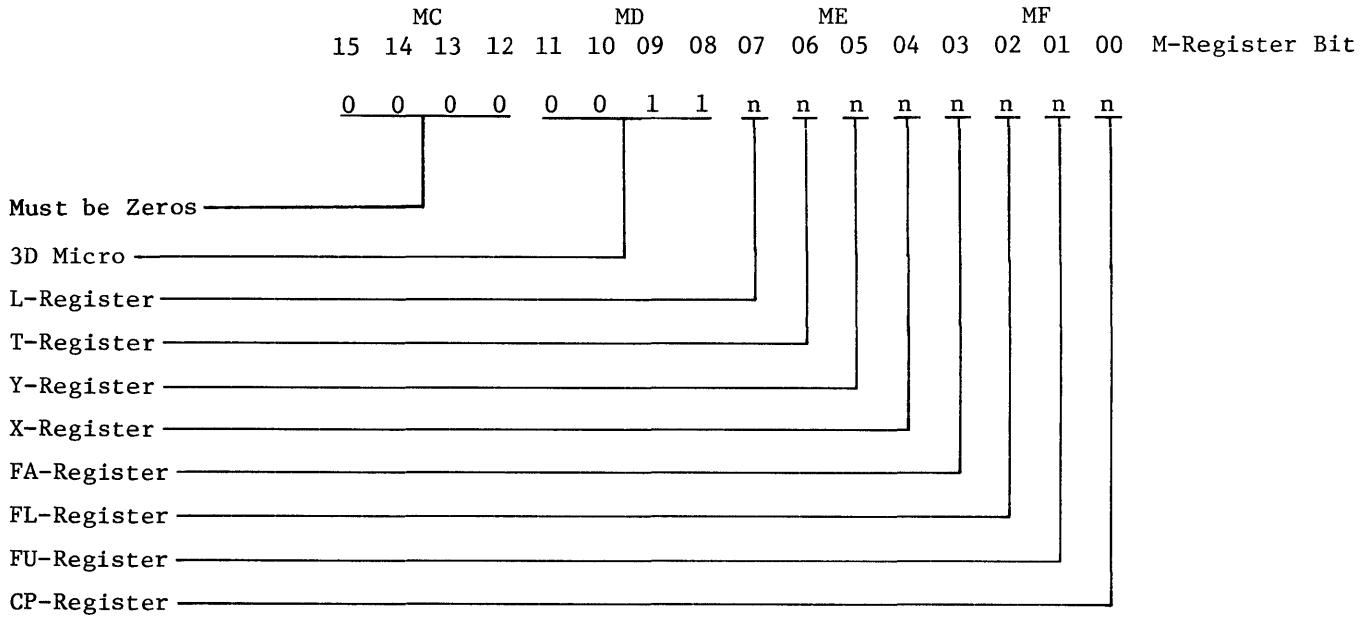
Figure 1-63. 145C (14C, 15C) Call



Swap data from main memory with the data in the specified register.

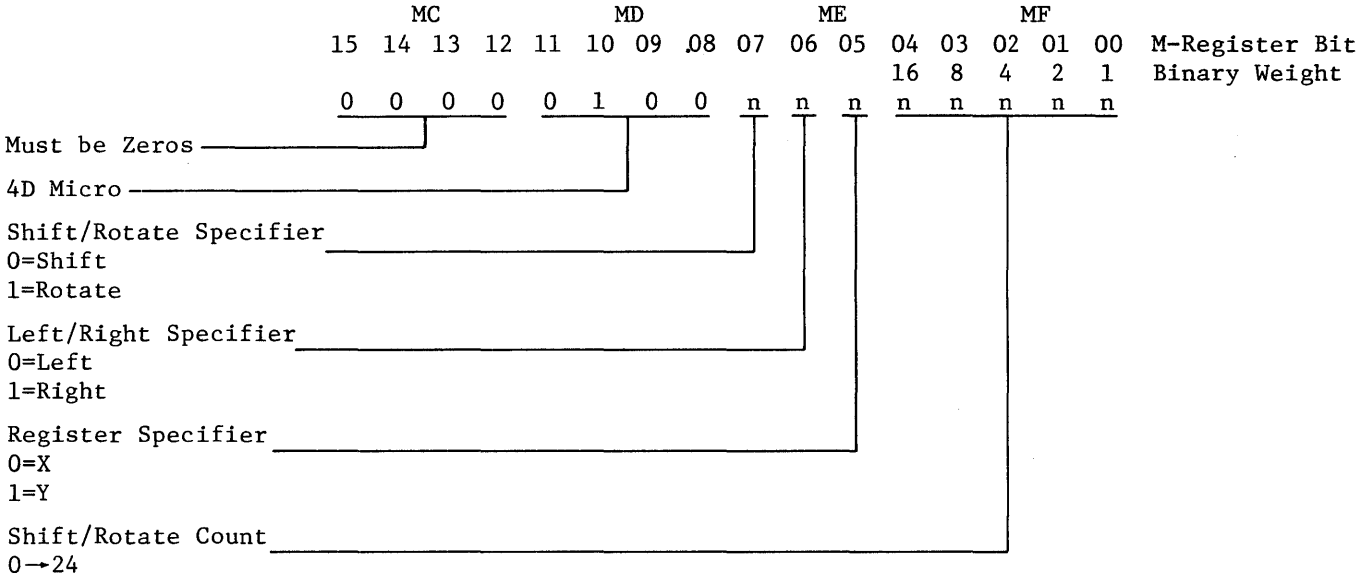
If the value of the memory field is less than 24, the data from memory is right-justified with left most significant zero-bits supplied while the data from the register is truncated from the left. Register FA contains the bit address of the memory field. If the value of the memory field length as given in the instruction is zero, the value in CPL is used

Figure 1-64. 2D Swap Memory



Clear the specified register unconditionally to zero if the respective bit is a one.

Figure 1-65. 3D Clear Registers

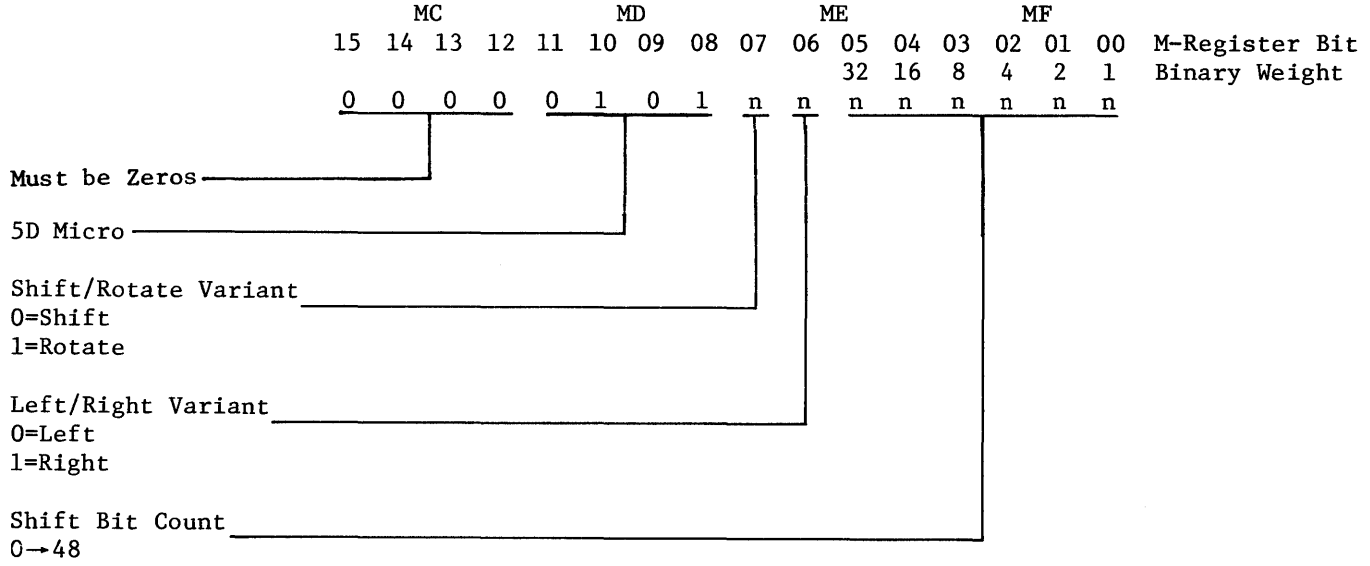


Shift (rotate) register X(Y) left (right) by the number of bits specified. Zero fill on the right and truncation on the left occurs for the left shift. Zero fill on the left and truncation on the right occurs for the right shift.

If the value of the shift/rotate count as given in the instruction is zero, the amount the operand is shifted (rotated) is zero.

Execution time is three clocks. It is the same for all counts

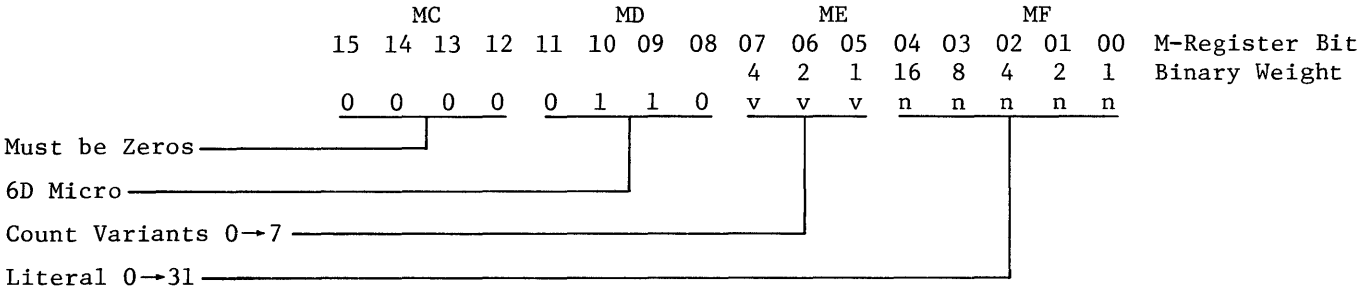
Figure 1-66. 4D Shift/Rotate X or Y



Shift/rotate registers X and Y left/right by the number of bits specified. The X-register is the leftmost (more-significant) half of the concatenated 48-bit XY-register.

Zero fill on the right and truncation on the left occurs for the left shift. Zero fill on the left and truncation on the right occurs for the right shift.

Figure 1-67. 5D Shift/Rotate X and Y



Increment (decrement) binarily the designated register(s) by the value of the literal contained in the instruction or by the value of CPL if the value of the literal is zero.

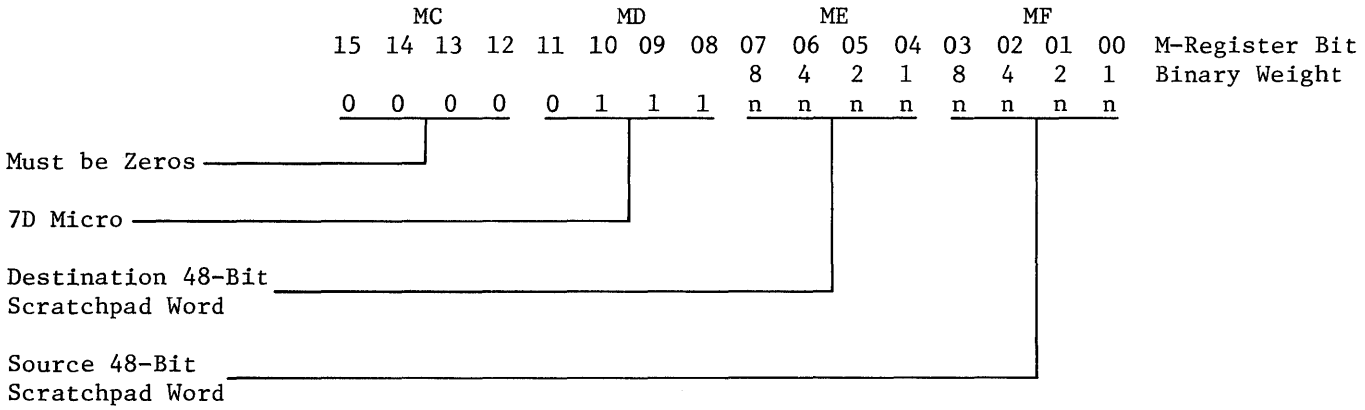
Neither overflow nor underflow of FA is detected. The value of FA may go through its maximum value or its minimum value and wrap around. Overflow of FL is also not detected. The value of FL may go through its maximum value and wrap around. Underflow of FL is detected and will not wrap around. The value of zero is left in FL.

COUNT VARIANTS

V=000 No count
001 Count FA up
010 Count FL up
011 Count FA up and FL down
100 Count FA down and FL up
101 Count FA down
110 Count FL down
111 Count FA down and FL down

Execution time equals four clocks.

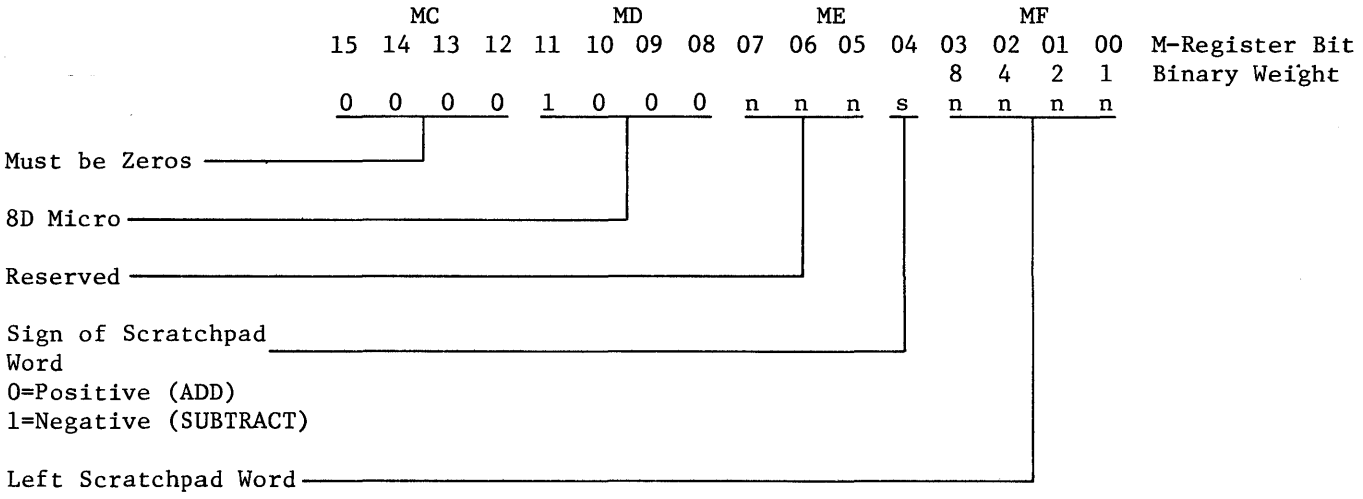
Figure 1-68. 6D Count FA/FL



Move the contents of the FA and FB registers to a holding register. Holding register is the scratchpad input latches. Move the contents of the left and right source scratchpad word to the FA and FB registers, respectively. Move the contents of the holding register to the left and right words of the destination scratchpad word.

Execution time equals 1 clock.

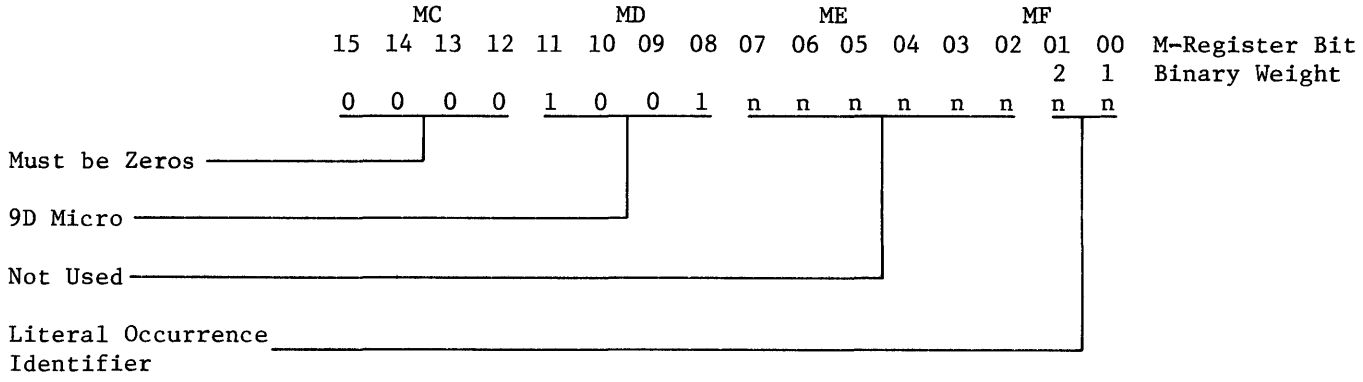
Figure 1-69. 7D Exchange Doublepad Word



Replace the contents of the FA register by the sum of the FA register and the specified scratchpad register or the difference between them (subtract operation).

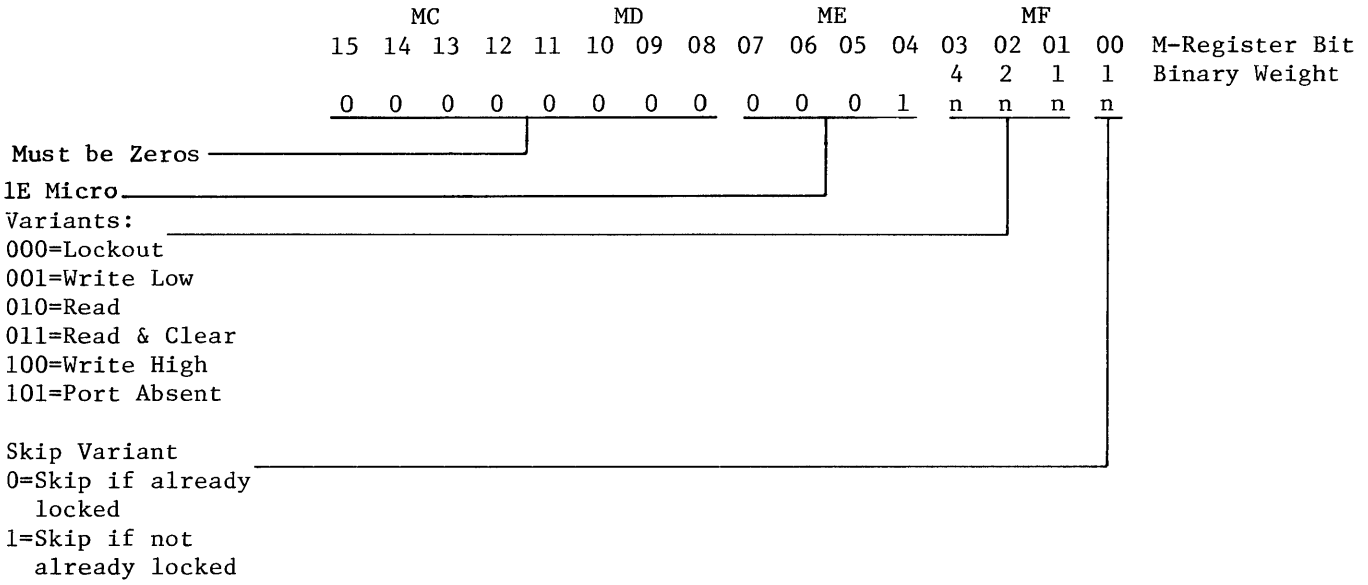
Execution time equals four clocks

Figure 1-70. 8D Scratchpad Relate FA



The monitor has no function programmatically and is treated as a no-op. However, execution of a 9D micro generates a special marker pulse which may be used to activate or synchronize external test equipment. Refer to section 4.

Figure 1-71. 9D Monitor



The dispatch is used to initiate port-to-port communications and to receive interrupt information from other ports. Since the dispatch system is shared by all ports, the processor must gain control of it by successfully completing a lockout prior to a dispatch write.

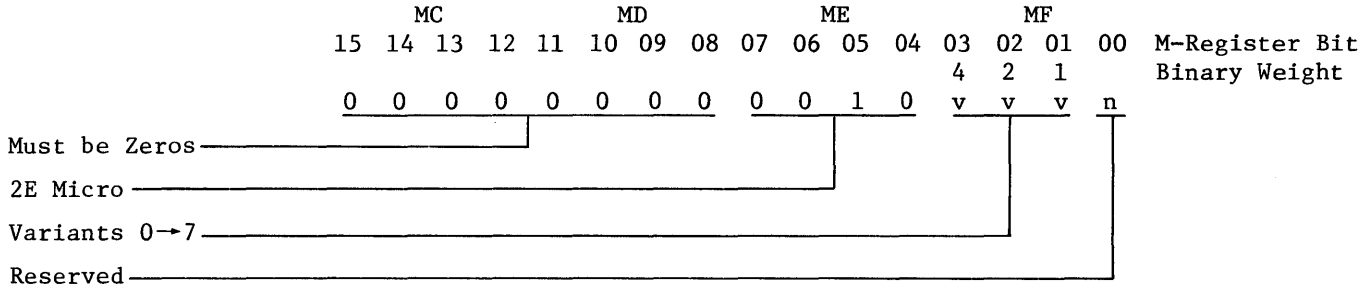
The skip variant allows skipping of the next 16-bit instruction based upon success or failure of the lockout attempt.

The write dispatch operation sets the lockout and interrupt flip-flops in the port interchange. It also stores the contents of the L-register into memory location 0 through 23 and the contents of the least-significant seven bits of the T-register (port and channel) into the port interchange dispatch register. In addition, it sets write high or resets write low, and the high interrupt flip-flop in the port interchange.

The read dispatch operation stores the contents of memory locations 0 through 23 into the L-register and the contents of the port interchange dispatch register into the least-significant seven bits of the T-register. The other 17 bits of the T-register are unaffected. The read and clear dispatch operation, in addition to performing the read dispatch operation, clears the lockout flip-flop, the two interrupt flip-flops, and the port device absent flip-flop in the port interchange. It does not clear any memory locations.

The port absent operation is executed by the processor when necessary to return a port absent level signal to another port indicating the absence of the designated channel.

Figure 1-72. 1E Dispatch



Perform the indicated operation on the tape cassette.

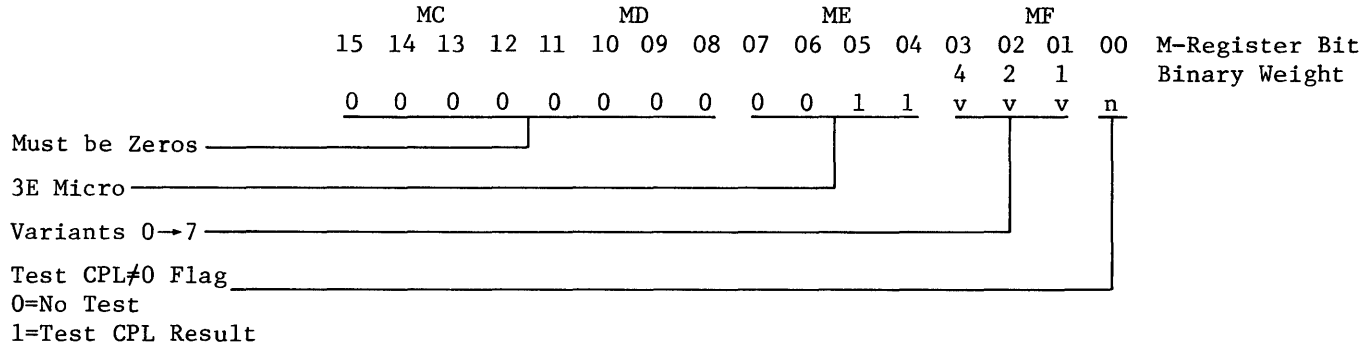
VARIANTS

V=0 Start tape
1 Stop tape (the processor also halts in tape mode.)
2 Stop tape if X≠Y
3→7 Undefined

All tape stop variants cause the tape to halt in the next available gap.

Execution time equals two clocks.

Figure 1-73. 2E Cassette Control



Set CPU to the value 1 if the value of FU is 4 and to 0 otherwise. The exception is: V=2, the value set into CPU is determined by SFU in lieu of FU.

Set the value of CPL to the value denoted or to the smallest of the values denoted in the following table.

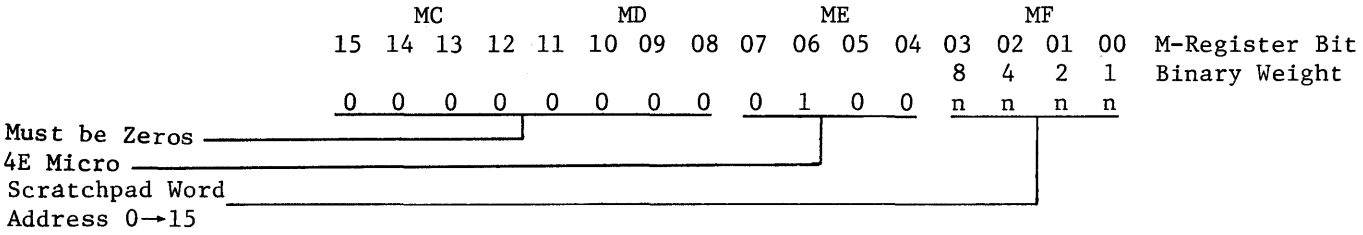
VARIANTS

V=0 FU
1 24 or FL
2 24 or SFL
3 24 or FL or SFL
4 CPL
5 24 or CPL or FL
6 CPL
7 Not Defined

If test flag equals 1 and final value of CPL is not zero, the next 16-bit micro instruction is skipped.

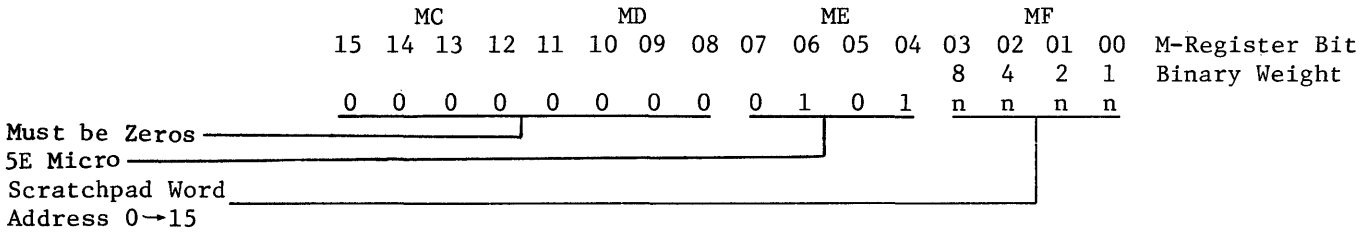
Execution time equals two clocks.

Figure 1-74. 3E Bias



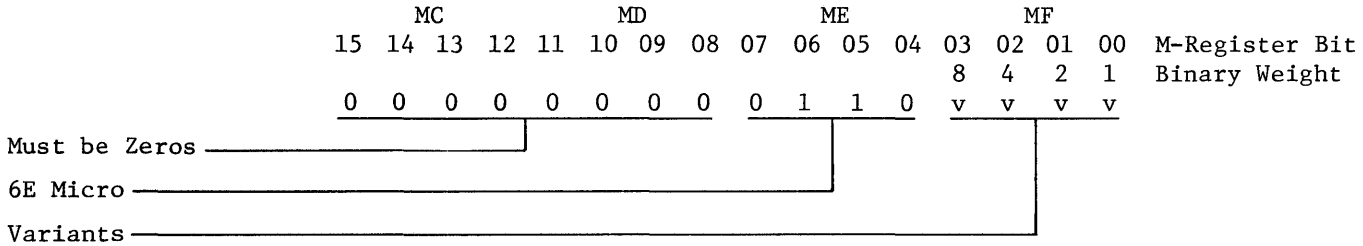
Move the FA and FB register's contents to the left and right words, respectively, of the designated scratchpad address.

Figure 1-75. 4E Store F Into Doublepad Word



Move the contents of the left and right words at the designated scratchpad address to the FA and FB register, respectively.

Figure 1-76. 5E Load F from DPW



Set the carry flip-flop as specified by the variants.

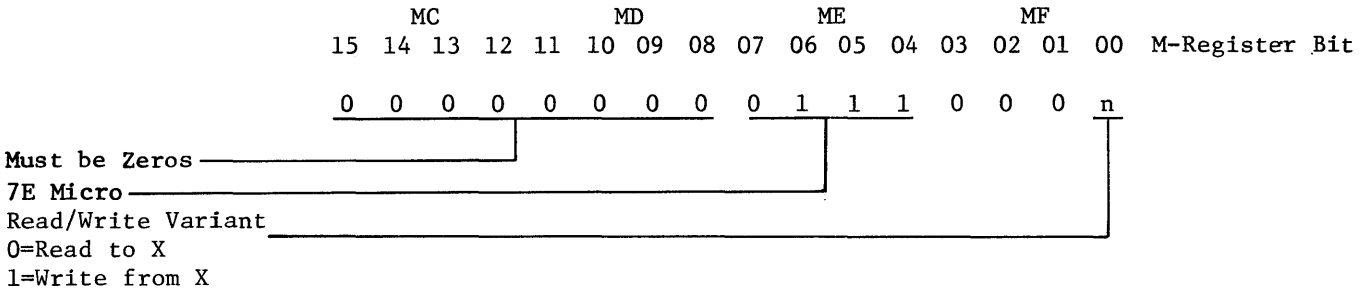
VARIANTS

V=1 Set CYF to 0
2 Set CYF to 1
4 Set CYF to CYL
8 Set CYF to CYD

Note that (X<Y) + (X = Y) *CYF = CYD

Execution time equals two clocks.

Figure 1-77. 6E Carry FF Manipulate



Write (R/W variant bit = 1)

Move the contents of the X-register to the MSM word specified by the address contained in the L-register. The data is truncated from the left.

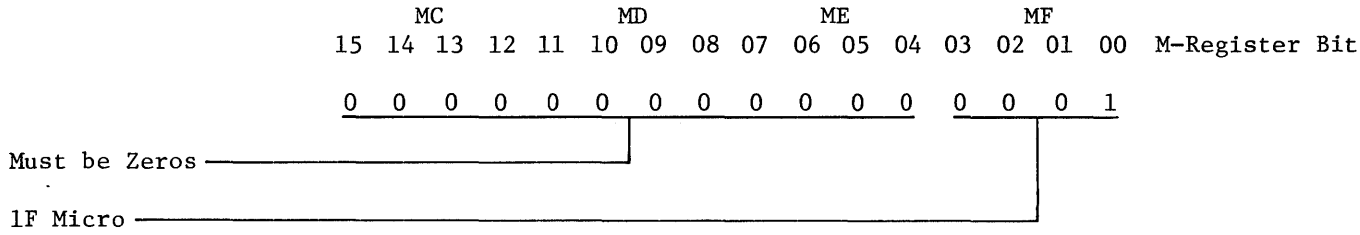
Read (R/W variant bit = 0)

Move the contents of the MSM word specified by the address contained in the L-register to the X-register. The data is right-justified with left-most significant zero bits supplied.

The lower four bits and the upper eight bits of the address in L are ignored.

The operation of this micro instruction causes the A-register to be moved to the TAS, and the L-register to be moved to A, before a read or write is performed.

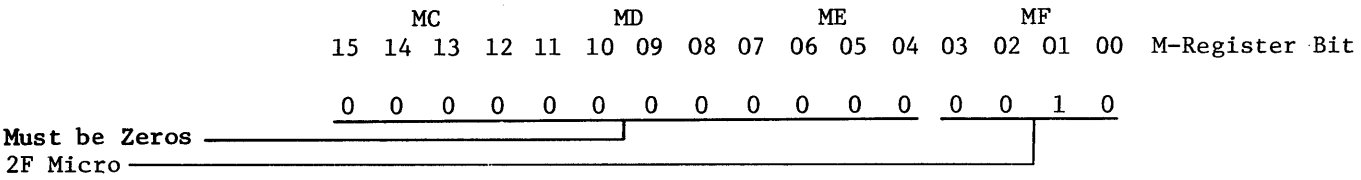
Figure 1-78. 7E Read/Write MSM



Stop the execution of micro instructions. The machine halts with the next micro in the M-register.

Execute time equals two clocks.

Figure 1-79. 1F Halt



The starting M-memory and starting main memory addresses are taken from the L-register and the FA register, respectively. The length of the data overlay in bits is taken from the FL register. The execution proceeds as follows:

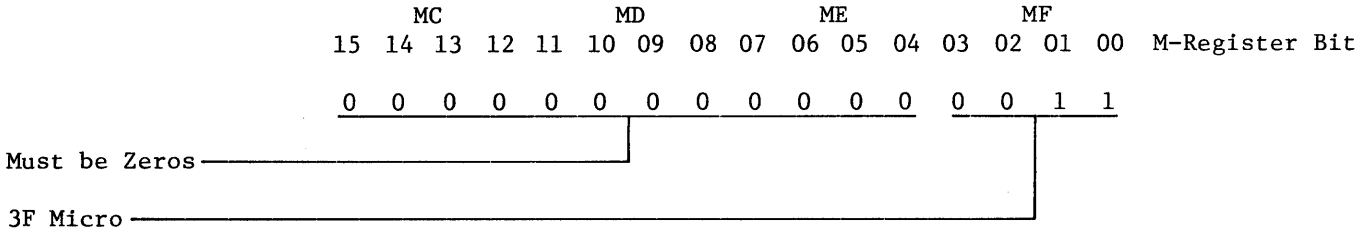
Move A to TAS

Move L to A

Cycle {
- Read 16 bits from main memory
- Store 16 bits in M-memory
- Decrement FL
- Increment FA and A
- If FL≠0, repeat cycle. If FL=0, go to halt.

Halt. Move TAS to A.

Figure 1-80. 2F Overlay M-Memory from Main Memory

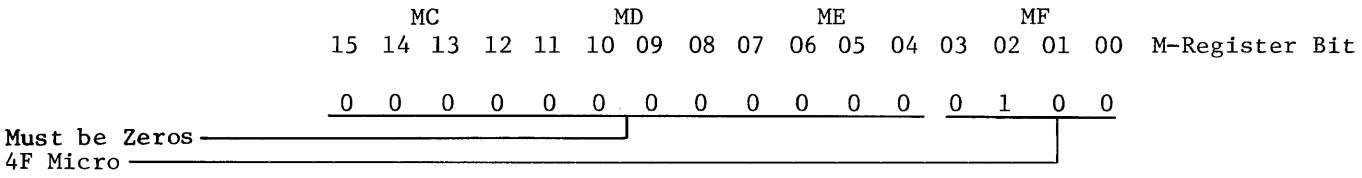


Shift the X-register left while counting FL down until FL=0, or until the bit in X referenced by CPL is a one. Zeroes are shifted into the rightmost end of X.

CPL = 1 references the rightmost bit of X while CPL = 24 references the leftmost bit of X.

Execute time equals six clocks per bit shifted plus two additional clocks if FL=0 or plus four additional clocks if MSBX=1.

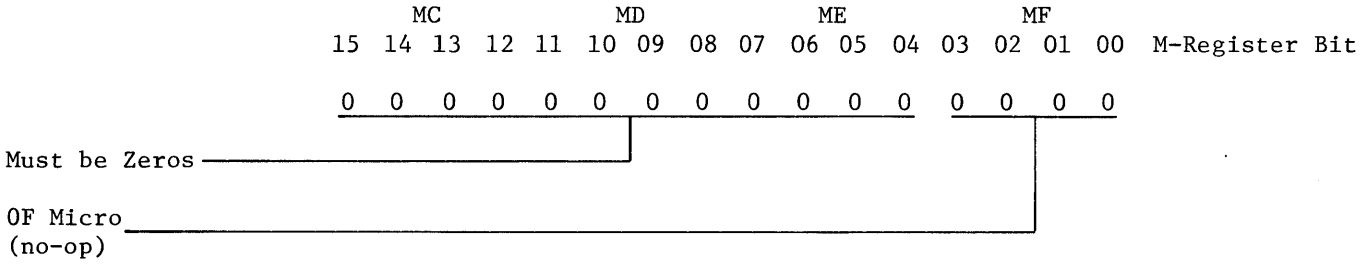
Figure 1-81. 3F Normalize X



Move the 24-bit value from the L-register to the MBR register. Move the least-significant four bits from the T-register to the TOPM-register.

Move the most-significant 20 bits from the T-register to the A-register, truncating the left-most six bits of the source.

Figure 1-82. 4F Bind



Skip to the next sequential instruction.

Execute time equals two clocks.

Figure 1-83. Zero NO Operation

SECTION 2

FUNCTIONAL DETAIL

INTRODUCTION

This section describes the individual circuits which make up the B 1720 Series Processor, port interchange and S-memory. The processor section includes the console control operational logic, system clock, current state logic, registers, 24-bit function box, 4-bit function box, M-string memory, port adapter/port device interface control logic, and I/O interface logic. Also included is a discussion of micro execution, in which each micro instruction is dealt with.

CONSOLE CONTROL OPERATIONAL LOGIC

The console controls represent an extension of the processor control logic which allow manual selection of micro operator functions. As such they serve as an operator interface with the processor, permitting overall control of system operation, monitoring of processor activity, and testing/troubleshooting facilities. The controls allow selection of micro operator functions because the majority of actions which may be initiated from the console duplicate the actions performed by micros. In addition, these actions are accomplished in the same manner, that being the insertion of appropriate voltage levels, either steady state or pulsed, at the proper decision making points of the control logic. An example of this is the operation of the register group and register select switches, which create binarily-coded control levels to exercise the register enabling logic within the processor. These control levels thus generated are identical to those produced by a 1C register move micro for accomplishing the same function. Of the console controls, which exercise logic circuitry, only the START, MODE, and REWIND switches cause functions not duplicated by software. The POWER off/on and CASSETTE ON/OFF switches control the application of power only.

CONSOLE LAMPS

The 24 console lamps represent a visual display of the contents of the main exchange. They are connected to it by way of a console lamp register and gating logic as shown in figure 2-1. The control of both the register and gating is by the freeze term, which causes the contents of the MEX to be latched into the register, and therefore, continuously displayed. In normal operations (processor running), the absence of the freeze term enables the gates and causes the register to operate in the D-set mode. Since the lamp register is enabled at each system clock pulse, the data on the MEX is transferred to the register. There is no storage involved in this mode. The freeze term is used only in the console current state (processor halted) when viewing the contents of an S-memory location. Appearance of freeze simultaneously disables the gates and changes the register operation to the bit-set mode. This causes the data present at the register's inputs to be latched into it and held for as long as the register remains in this mode.

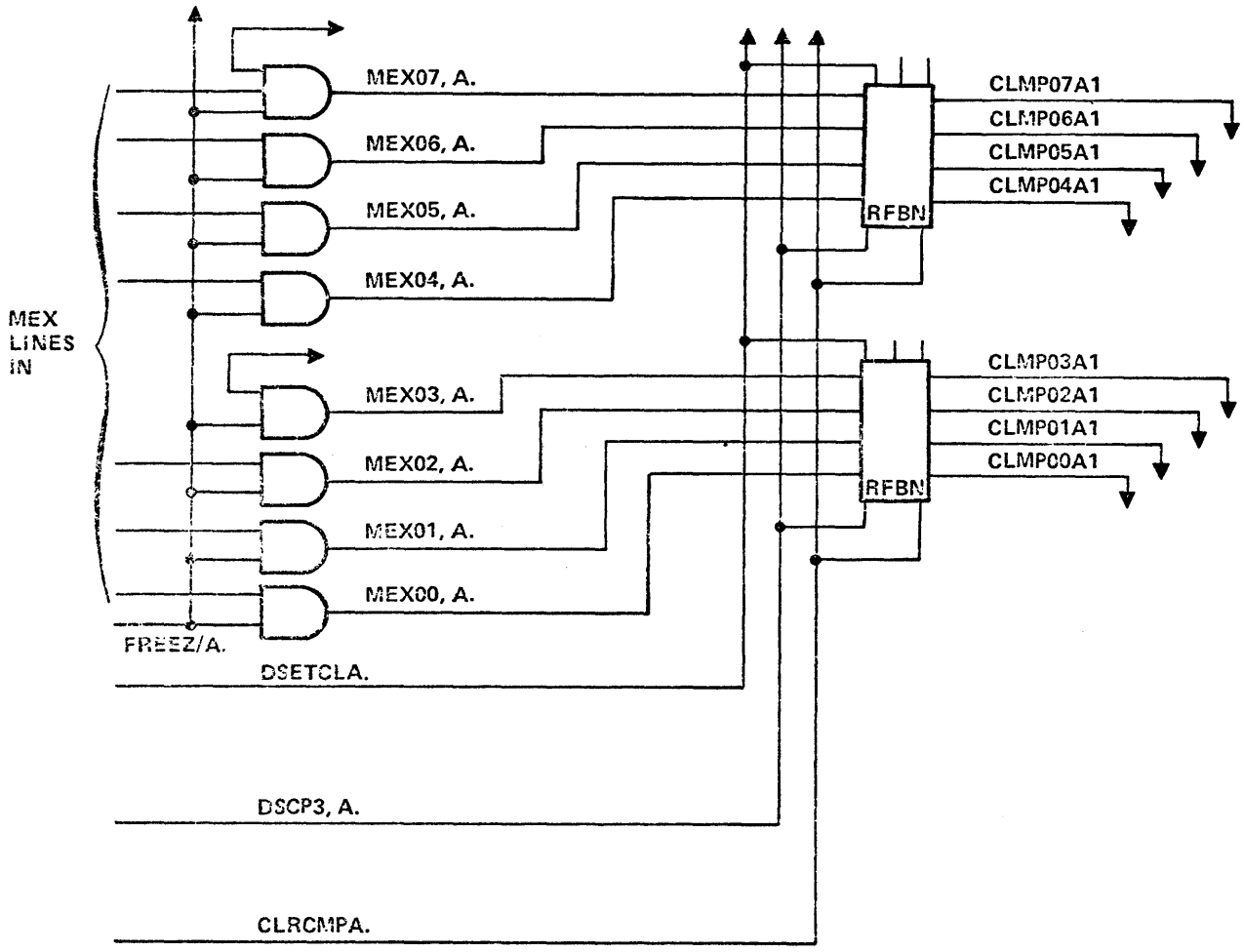


Figure 2-1. Console Lamp Register

CONSOLE SWITCHES

The 24 console switches serve as a manual entry point for 24 bits of data, the configuration of which is selectable by the operator. Each switch makes (up position) or breaks (down) contact to a +4.75 volt source, which represents a logical "high." The contents of the switches are gated directly to the MEX via 24 logic gates. These gates are all enabled simultaneously by the term MC ← CSWM., which is derived from LOAD or READ/WRITE. See logic schematics card M, page 2.

REGISTER GROUP AND REGISTER SELECT SWITCHES

The register group and REGISTER select switches cause creation of a 1C register move micro by exercising the load/display register logic (figure 2-2). The switch outputs are the binarily-weighted RGRPnT1 and RSELnT1 levels, which correspond to the coordinates of the desired register's location in the register table (table 1-1). These, in turn, control the RFANS which generate the variant portion of the micro and gate it to the MOP lines, which transport it to the micro decoding logic. The load/display register is active only in the console current state (when the processor is halted). It is capable of functioning in two

operational states for generation of a micro to read the contents of a selected register, or a micro for loading data from the console switches into the register. Note that whenever the processor is in the console current state, the term CONSCSPO is true, enabling the load/display register output buffers, and causing a constant true signal on MOP line 12. This bit in itself signifies the 1C micro, with the variants (MOP lines 00 through 11) being supplied by the register.

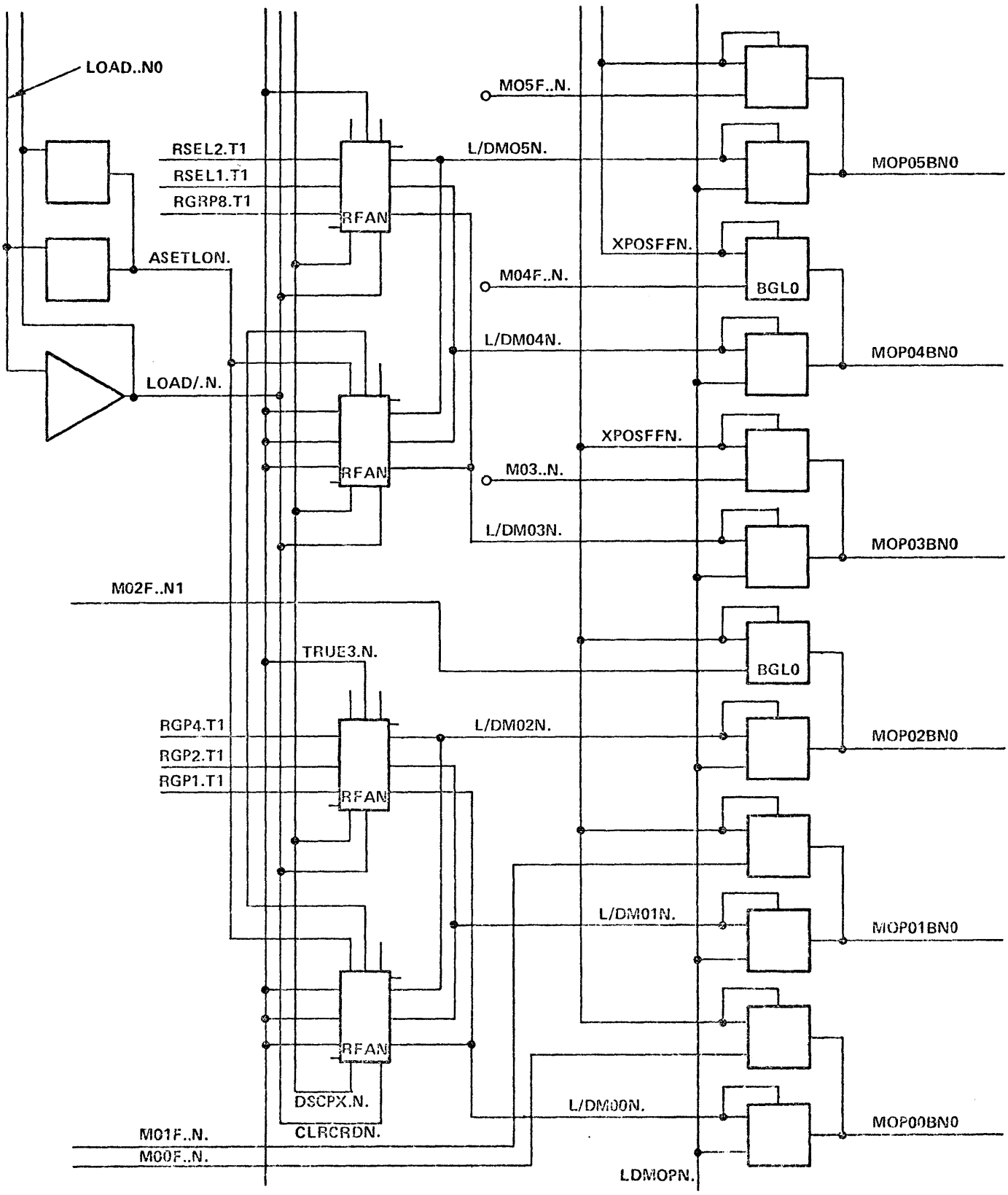


Figure 2-2. Load/Display Register (Bits 00 through 05 Shown)

The overall function of the load/display register becomes clear when the 1C register move micro is examined (figure 2-3). Briefly stated, that is to gate the register selection code from the console switches to the MOP lines corresponding to source (for a read) or sink (for a load or write), and to generate and gate the null selection code to the opposing set of MOP lines. As discussed in section 1, the console lamps represent null in a read condition, and the 24 console switches represent it in load or write. This function is accomplished by controlling the operating mode of the register chips. In the read state, which occurs continuously when the processor is halted, the level LOAD..NO is false, causing LOAD/.N to be true. RSETHIN and RSETLON are constantly true. LOAD/.N and RSETLON being true cause RFANs 6 and 8 to operate in the bit-set mode, meaning that at the next clock pulse all outputs are true. This condition causes a true level to be gated to MOP lines 00 through 05, which specifies NULL as sink. RFANs 1, 3, 5, and 7 always operate in the direct set mode, meaning that the register selection code appearing at their inputs also appears at their outputs. Note that 1-3 and 5-7 each constitute a pair on which the register selection code appears. The appearance of this code at the outputs of RFANs 5 and 7, which feed MOP lines 00 through 05 in parallel with RFANs 6 and 8, is of no significance because the "all true" outputs of the latter forces selection of null. In the case of RFANs 1 and 3, the register selection code is significant because LOAD..NO, being false with RSETHIN. true, causes RFANs 2 and 4, which feed MOP lines 06 through 11 in parallel with 1 and 3, to operate in the bit reset mode. The register selection code is thus exposed, and specifies the source.

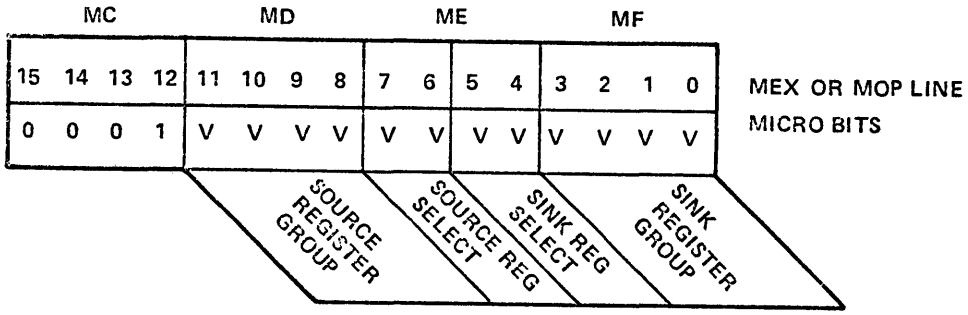


Figure 2-3. 1C Register Move Micro

When the console LOAD switch is pressed, the level LOAD..NO comes true for two clock pulses, causing LOAD/ .N. to go false. These levels put RFANs 6 and 8 in the bit reset mode and RFANs 2 and 4 in the bit set mode. With all outputs of RFANs 2 and 4 true, a true level is gated to MOP lines 06 through 11, specifying the source register as NULL. RFANs 6 and 8 being disabled exposes the register selection code at the outputs of RFANs 5 and 7, which are gated to MOP lines 00 through 05, specifying the sink register. Note that reading and writing in S-memory and M-string memory is handled in the same manner as for registers, but involves additional circuitry. An address must be specified, using FA for S-memory or A for MSM. In addition, the READ/WRITE pushbutton must be used when accessing S-memory. Console read and write timing are shown in figures 2-4 and 2-5, respectively.

MODE SWITCH

The MODE switch places the processor in one of the three possible operational modes: step, run, or tape. With the processor halted and the MODE switch in the STEP position, pressing the START button causes one micro to be executed, then a return to the halt state with the next micro in the M-register. If the MODE switch is in the RUN mode position, pressing START initiates continuous execution of micros. This continues until either the HALT button is pressed or an internal halt condition arises. In the TAPE mode position, the processor, upon pressing of the START button, begins cassette movement and executes micros from the tape. The machine is returned to the halt state by pressing the HALT button or from an internal halt condition.

Processor operational mode is determined from two control levels from the MODE switch: SNGMODT1 and RUNMODT1. (Refer to figure 2-6.) These generate the control levels STEP..P., RUN...P., and MTR...P. which exert overall control over the current state logic, finish logic, and other sections having major influence in processor operations. A detailed explanation of all the actions caused by the mode control signals is beyond the scope of this discussion. However, in all cases these actions follow the logical intent of the selected mode (as described above). Note that only STEP..P. and RUN...P. are the direct result of control levels from the console. MTR...P. (tape mode) is the inverse of RUN..P., and therefore the two can never be true at the same time. This restriction does not apply to STEP..P. which, when true, overrides either of the other two levels. Thus, if STEP..P. comes true when the processor is running, it forces a halt at the end of the micro currently executing. This method is used to halt the processor for the "A equals console switches" feature. In this case, the term STEP..P. is generated by the presence of A=CSW.MO which comes from a static compare of the value contained in A and the console switches, EXECCSP (execute current state) and NULREGP. (null register). This latter term is decoded from the register selection levels (from the REGISTER select and register group rotary switches) and indicates that null (the 24 console switches) has been selected. Note that all of the logic thus involved operates without the use of any clock signals.

The processor halt in response to the HALT button on the console is also brought about by generation of the STEP..P. term. Refer to the following HALT pushbutton description for details.

HALT PUSHBUTTON

Use of the HALT pushbutton causes the processor to cease operations at the completion of the micro currently executing in the M-register. The action brought about by pressing HALT is dependent on the operating mode which has been selected. In the run mode, the next micro is placed in M, and the MSM address (in A-register) is upcounted to indicate the second succeeding micro. In tape mode, the currently executing micro remains in M, and A is not upcounted. Either case prepares the processor to begin operations where it left off when START is again pressed.

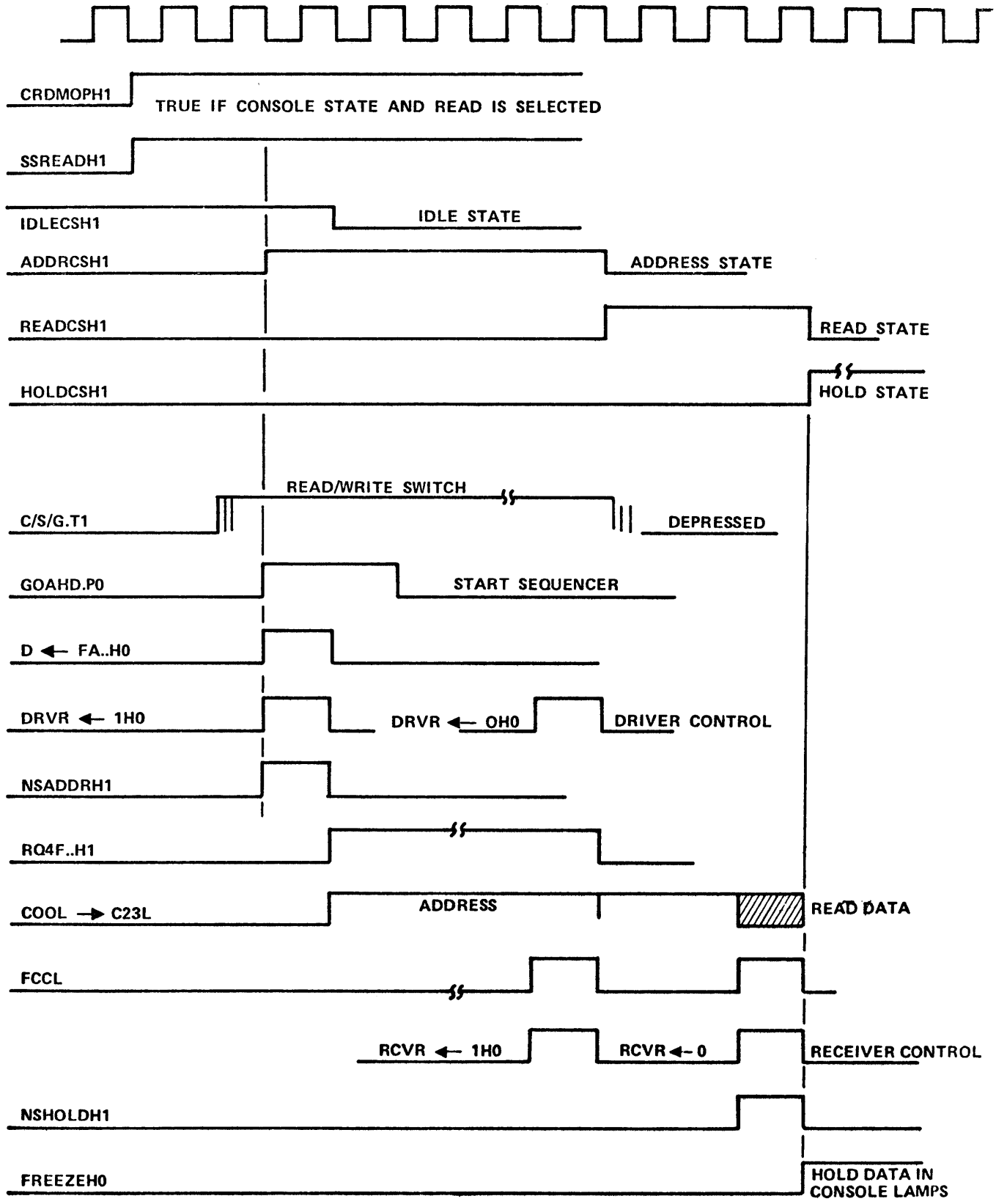


Figure 2-4. S-Memory Console Read Timing

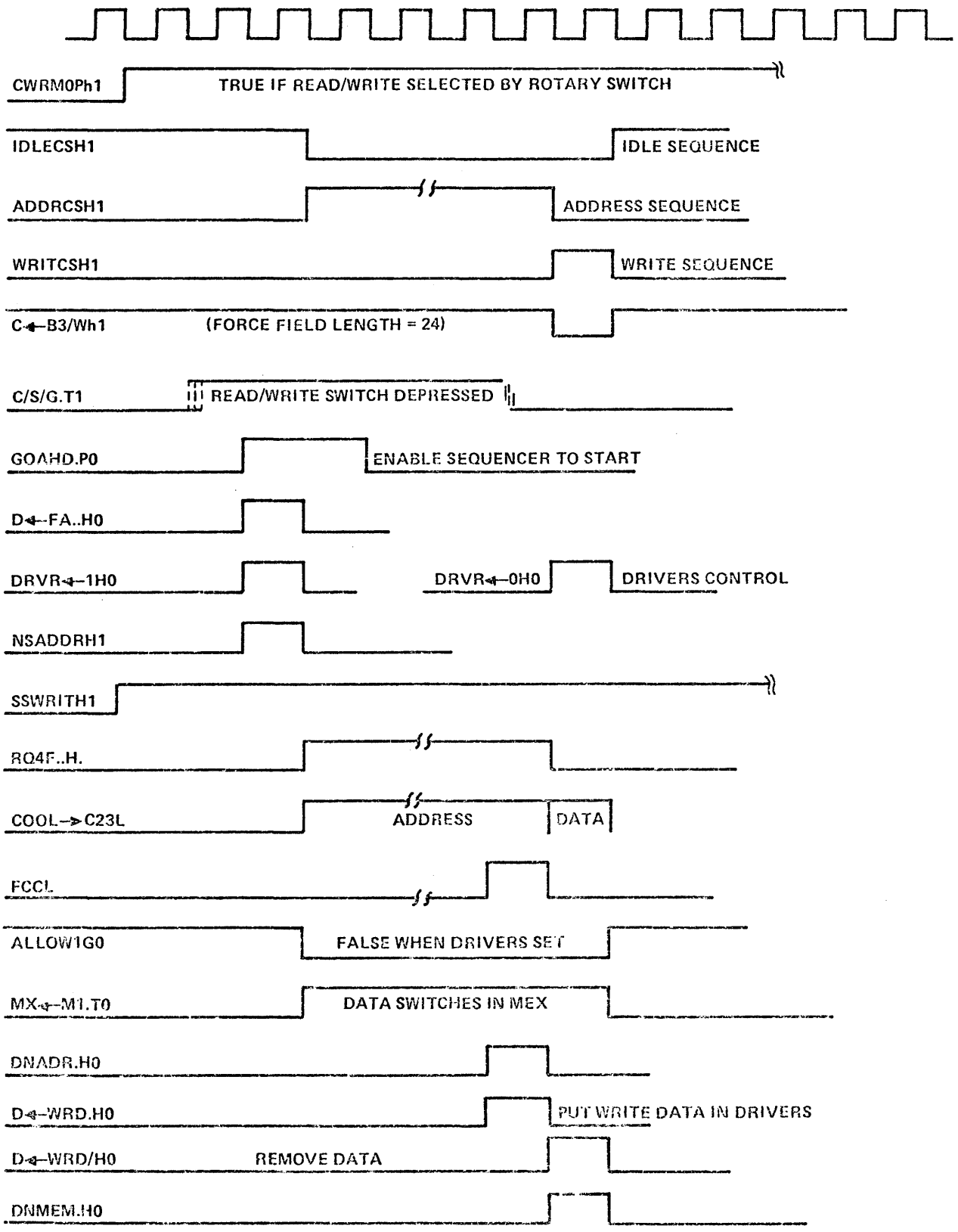


Figure 2-5. S-Memory Console Write Timing

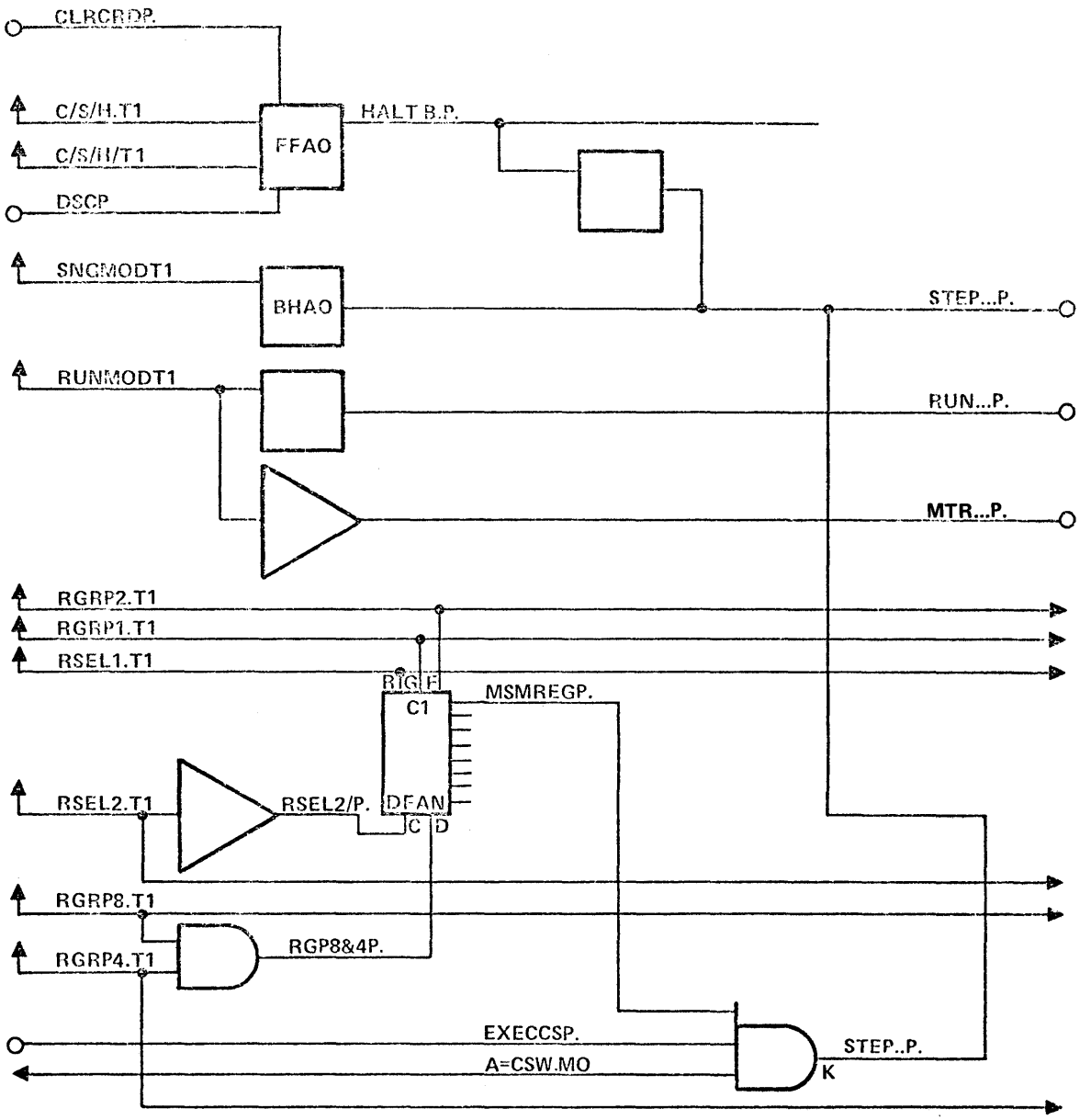


Figure 2-6. Mode Switch Operational Logic (Also HALT Pushbutton)

Pressing the HALT button generates the control level C/S/H.T1 and removes the level C/S.H/T1. Refer to figure 2-7. This is done by switch action. At the next system clock pulse, flip-flop AO is set, generating the HALTPB.P. term. This, in turn, generates STEP..P. by way of a buffer, causing the processor to return to the step mode at the conclusion of the executing micro. This function is the same as that caused by selection of step when the processor is running, or by "A equal console switches" when that option is used, except that HALTPB.P. is clock synchronized.

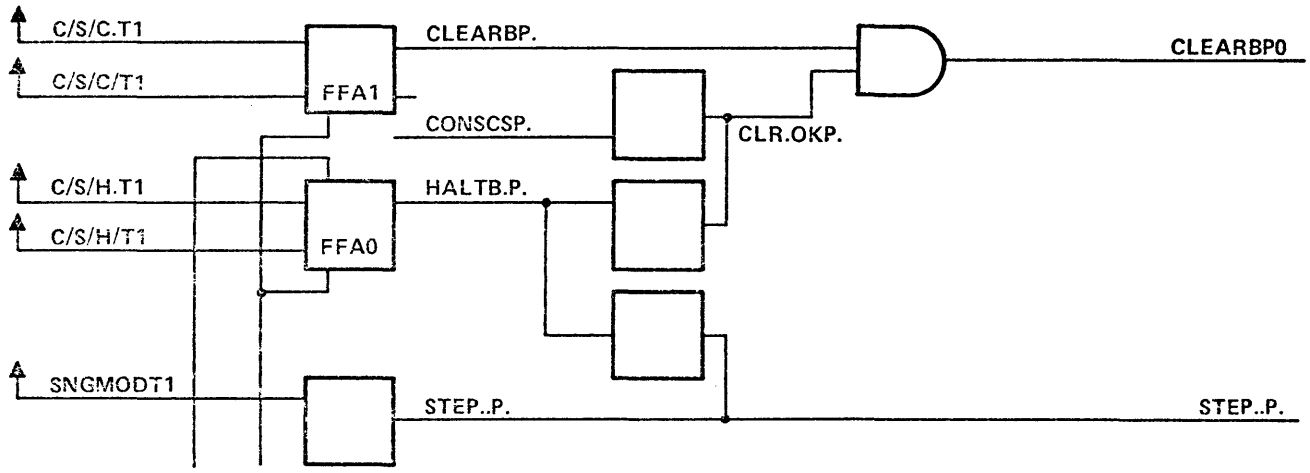


Figure 2-7. HALT Pushbutton Operational Logic (Including CLEAR Pushbutton Logic)

CLEAR PUSHBUTTON

The CLEAR button generates the CLEARBPO level which ultimately becomes the master system clear (MSCLR) signal that goes to the port interchange. This signal serves as a request for the clear operation. When the port interchange grants the clear request, the level CLRL is returned to the processor, where it becomes the GPCLRBHO (general processor clear) signal. This resets the system by clearing all registers and flip-flops in the processor, port interchange, memory logic, and I/O controls to a defined clear state. The CLEAR button is active only when the processor is halted (in the console current state). Note, however, that if the processor cannot be halted by ordinary means, a halt may be forced by simultaneously depressing the HALT and CLEAR buttons. This is possible because a special circuit was included to enable CLEAR when HALT is pressed, regardless of the current state existing in the processor.

Pressing CLEAR creates the level C/S/C.T1 (console/switch/clear) and removes the level C/S/C/T1 (console/switch/clear not), which are applied respectively to the set and reset inputs of flip-flop 1. (refer to figure 2-8). As with halt, these levels are produced directly by action of the CLEAR switch. With the set input high, the flip-flop output also goes high at the next clock pulse, generating the CLEARBP. (CLEAR button) signal. This signal remains present for as long as CLEAR is held pressed. CLEARBP. is ANDed with the CLR.OKP. (clear OK) signal to produce CLEARBPO which goes to the PAPDIC logic. There it assumes the mnemonic MSCLR..1 (master system clear), and is passed unchanged to the port interchange. The signal CLR.OKP. is true whenever the processor is in the console current state, or the HALT button is pressed (CONCSP. or HALTB.P. true). The clear signal returned from the port interchange is known as CLRL...1 (clear level). It is passed unchanged through the PAPDIC logic, and there assumes the mnemonic GPCLRBHO (general processor clear) which performs the clearing function throughout the processor.

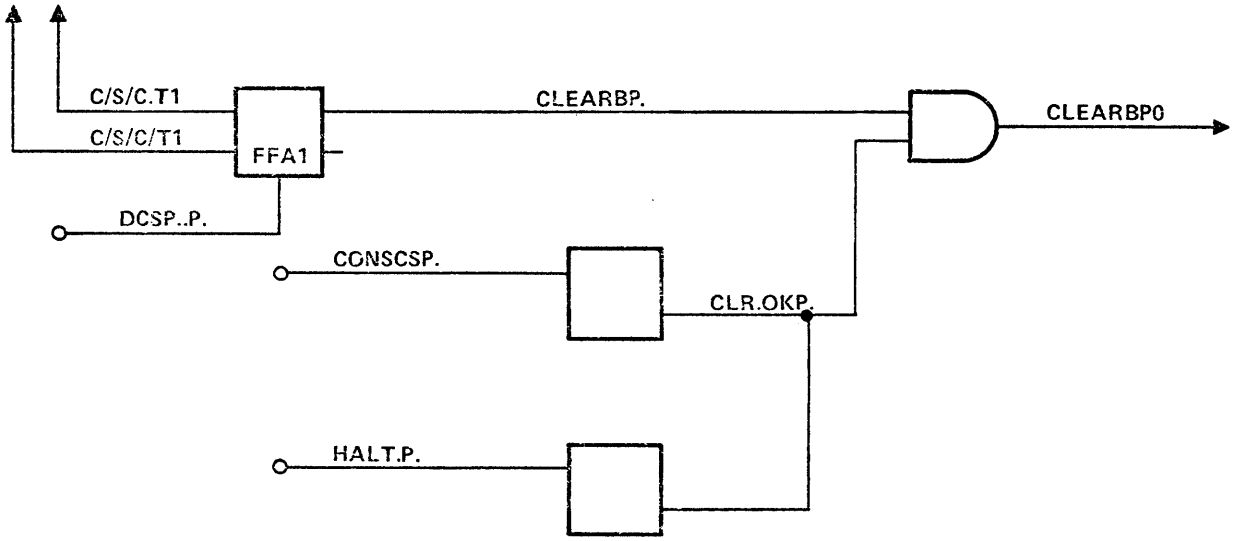


Figure 2-8. Clear Signal Generation

INTERRUPT SWITCH

The INTERRUPT switch sets bit 0 of the CC register. The presence of the interrupt bit is handled programmatically. In hardware terms, placing this switch up creates the level C/S/I.T1 (refer to figure 2-9). This causes the output of the flip-flop FFA1 H3 on card A to go high at the next clock pulse, generating the signal CCOF..A. (CC bit 0). Since C/S/I.T1 is applied to the J-input of the dual-mode flip-flop, CCOF..A. remains high until cleared or set to zero. This flip-flop, as can the others comprising the CC and CD registers, may be loaded from other external sources. This is accomplished by changing the mode to direct set and applying the data to the D inputs of each flip-flop.

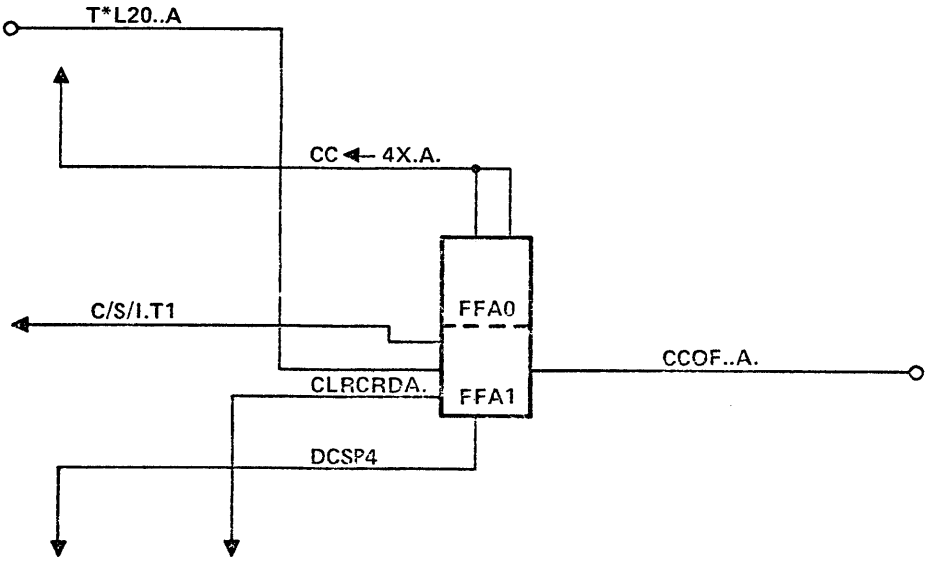


Figure 2-9. INTERRUPT Switch Operational Logic

INC (INCREMENT) PUSHBUTTON

Pressing the INC button causes the A or FA address to be upcounted by one word, or 16 bits (all micros are 16 bits long). The register incremented depends on whether MSM(A) or

S-memory (FA) is addressed by the console switches. This capability is provided as a convenience to the operator for use in examining/altering the contents of memory, and is available only in the console current state.

Operationally, pressing INC creates the level INCA..T1 and removes the level INCA/.T1 (refer to figure 2-10). These feed the J- and K-inputs, respectively, of flip-flop 2. This stage, along with flip-flop 2, is a one-shot pulse generator. With the J-input true and K-input false, flip-flop 2 changes state at the trailing edge of the next clock pulse, producing the level INCAF1P. This, in combination with INCAF2P. and CONCSP (console current state), produces CONINCP. (console incrementation). INCAF2P. is a continuous level which is absent only when FFA0 DO is set. This occurs at the next clock pulse, when its inputs INCAF1P. (J) and INCAFOP. (K) are true and false, respectively. INCAF2P, going false one clock pulse after INCAF1P., comes true also and causes CONINCP. to go false, thus restricting it to a one clock pulse length. The latter condition, with INCAF1P true and INCAF2P false, remains for as long as the INCA console switch is pressed. When it is released, the converse operation occurs, with INCAF1P. going false, and one clock later INCAF2P going true. Since the two are never true at the same time during resetting, CONINCP. is not generated. Both flip-flops are reset when the signal CLRCRDP is received by changing the operational mode from JK to D-Set.

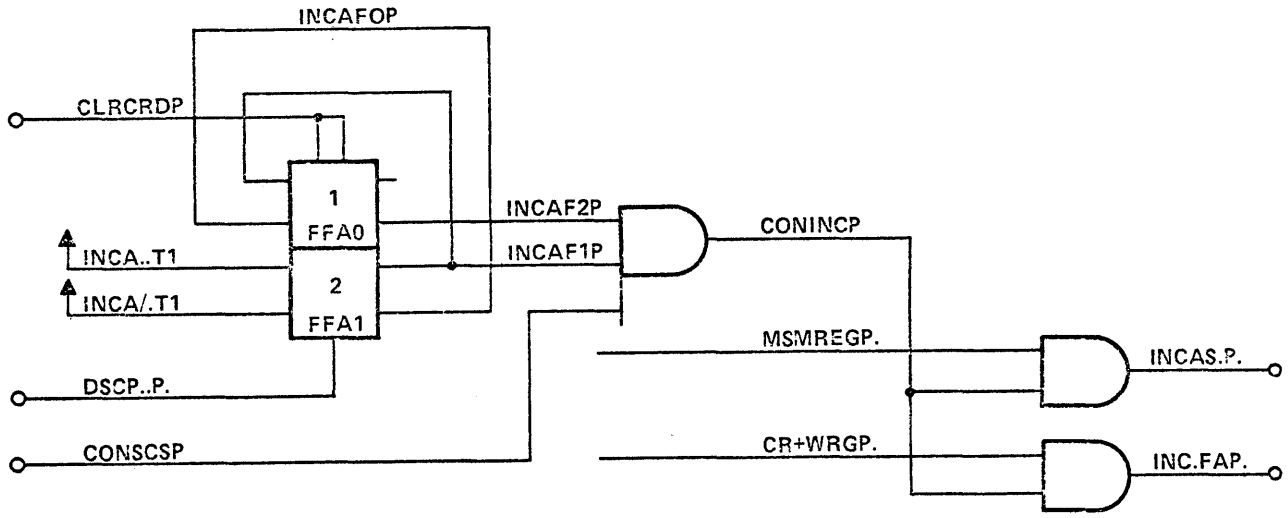


Figure 2-10. INCREMENT-A Switch Operational Logic

The signal CONINCP. is ANDed with CR+WRGP (console read or write register group) and INCAS.P (increment A-register straight) to produce the signals MSMREGP (MSM register group) and INC.FAP. (increment FA register), respectively. The signals CR+WRGP and MSMREGP are decoded from the register group and register select signals.

START PUSHBUTTON

Pressing the START button initiates processor operation by forcing the current state generation logic to go to start current state. The execute current state follows automatically, but thereafter control is dependent on the program executing and the operational mode selected. Start is active in the console current state only.

The start operational logic is a one-shot pulse generator which is identical to the device used for INCA. Therefore, it is not described in detail. As with INCA, the control levels C/S/S.T1 and C/S/S/T1 from the START switch are false and true, respectively, in the resting

state. Pressing START reverses the state of both, and this causes the output signals STRTF1B and STRTF2P. to both be true for one clock pulse only (both change state, but on succeeding clock pulses). This combination generates STARTBP. (START button), which is ANDed with CONSCSP. (console current state) to produce NSSTRTP. (next state start). (Refer to figure 2-11.) This latter signal moves the current state generation logic out of the console current state.

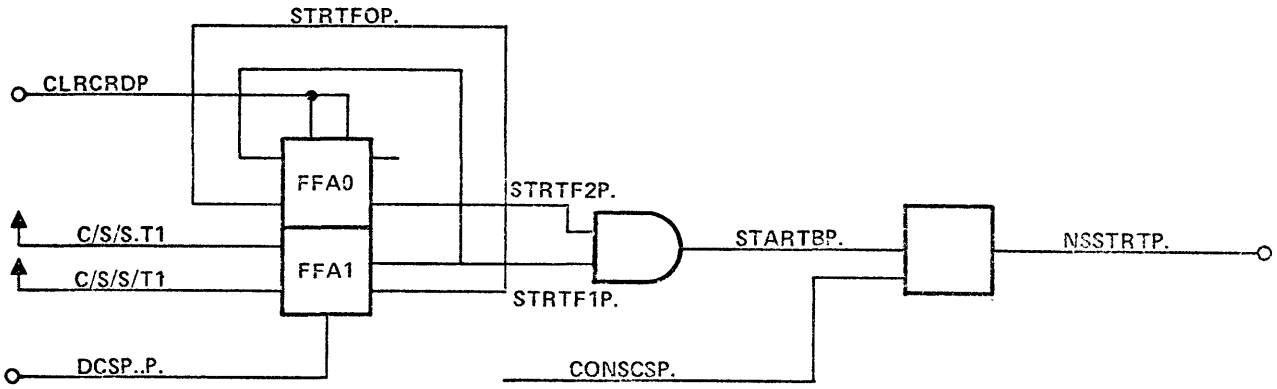


Figure 2-11. START Switch Operational Logic

READ/WRITE PUSHBUTTON

The READ/WRITE button initiates a console read or write operation in S-memory. This is accomplished by enabling the "pseudo micro" generated by the console register group and register select switches. The S-memory cycle is initiated by pressing READ/WRITE, which signals the PAPDIC logic to proceed with the operation.

The READ/WRITE button operational logic is a one-shot pulse generator identical to those used for Start and INCA. Refer to the discussion of Start for a detailed description of how this device operates. In the resting state, the control levels C/S/G.T1 and C/S/G/G1. from the READ/WRITE switch are false and true, respectively. This results in the flip-flop output GOHDF1P being false and output GOHDF2P being true, and therefore, the output of the following AND gate (GOAHD.PO) is false. When READ/WRITE is pressed, GOHDF1P comes true and remains so. One clock pulse later GOHDF2P goes false. This causes the AND gate output, GOAHD.PO (go ahead) to come true for one clock pulse. When READ/WRITE is released, GOHDF1P and GOHDF2P revert to the resting condition (false and true respectively) on subsequent clock pulses. Refer to figure 2-12.

LOAD PUSHBUTTON

Pressing the LOAD button produces a two-clock-long enabling signal which is used in the load/display register. There it causes the load pseudo micro to be produced. Refer to the register group/register select switch discussion for details on this operation. LOAD is active only in the console current state.

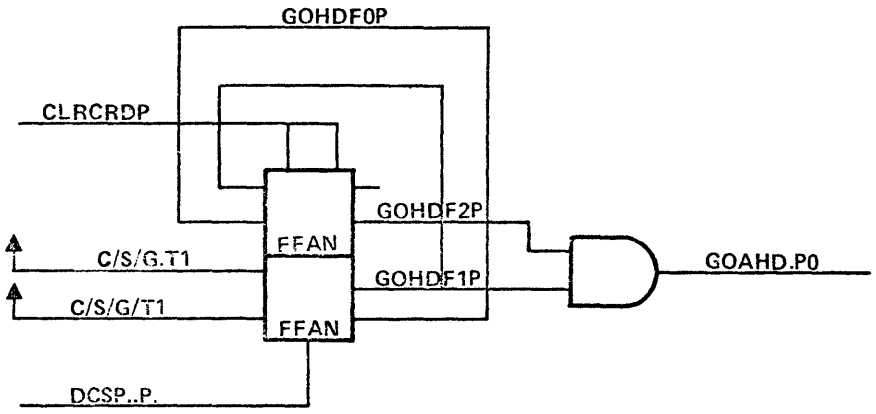


Figure 2-12. READ/WRITE Pushbutton Operational Logic

The load operational logic is a one-shot pulse generator adapted to create a two-clock long output signal. In the resting state, the levels C/S/L.T1 and C/S/L/T1 from the LOAD switch are false and true, respectively. C/S/L.T1 being false prevents generation of the signal LD.ON.N., and therefore, flip-flop 2 cannot be set, with the result being that LOAD..NO remains false. When LOAD is pressed C/S/L.T1 and C/S/L/T1 change state to true and false, respectively. With C/S/L.T1, CONSCSPO (console current state) and LDBF1/N. (reset output of flip-flop 1) true, the signal LD.ON.N. is generated. This sets flip-flop 2 at the next clock pulse, generating LOAD..NO. Refer to figure 2-13. Note that in addition to being the output, LOAD..NO. feeds the set input of flip-flop 1. This causes 1 to set at the clock pulse following the appearance of LOAD..NO, generating LDBF1.N., and causing LDBF1/N. to go false. And gate 3 is therefore immediately disabled, forcing LD.ON.N. false, allowing LDBF1.N. to reset flip-flop 2 at the next clock pulse. LOAD..NO is thus forced false after two clock pulses. Resetting flip-flop 2 generates LDBF2/N which prepares AND gate 4 for completion of the sequence. This occurs when LOAD is released; causing C/S/L.T1 and C/S/L/T1 to revert to false and true, respectively. C/S/L/T enables gate 4, generating LD.OFFN., causing flip-flop 1 to reset at the next clock pulse. The entire circuit is thus returned to the resting condition.

RUN INDICATOR

The RUN indicator, when lit, indicates that the processor is in the run state and executing micros. The micros being executed may come from MSM, S-memory, or the cassette tape drive. Run is derived from the inverse of CONSCSP (console current state). In other words, run is true whenever the processor is not in the console current state (halted). See schematic card P, sheet 8.

STATE INDICATOR

The STATE indicator, when lit, indicates that bit 3 of the CC register is set. This function is handled entirely by software (a micro must be executed to set or reset the bit).

PARITY INDICATOR

The PARITY indicator, when illuminated, indicates that a parity error has been detected during one of two conditions: a) non-recoverable parity error on a cassette tape read, or b) a parity error on an S-memory micro fetch. In either case, detection of a parity error causes the processor to halt, with the PARITY lamp serving as an indication of why this unexpected termination of operations occurred. It is important to note that detection of a

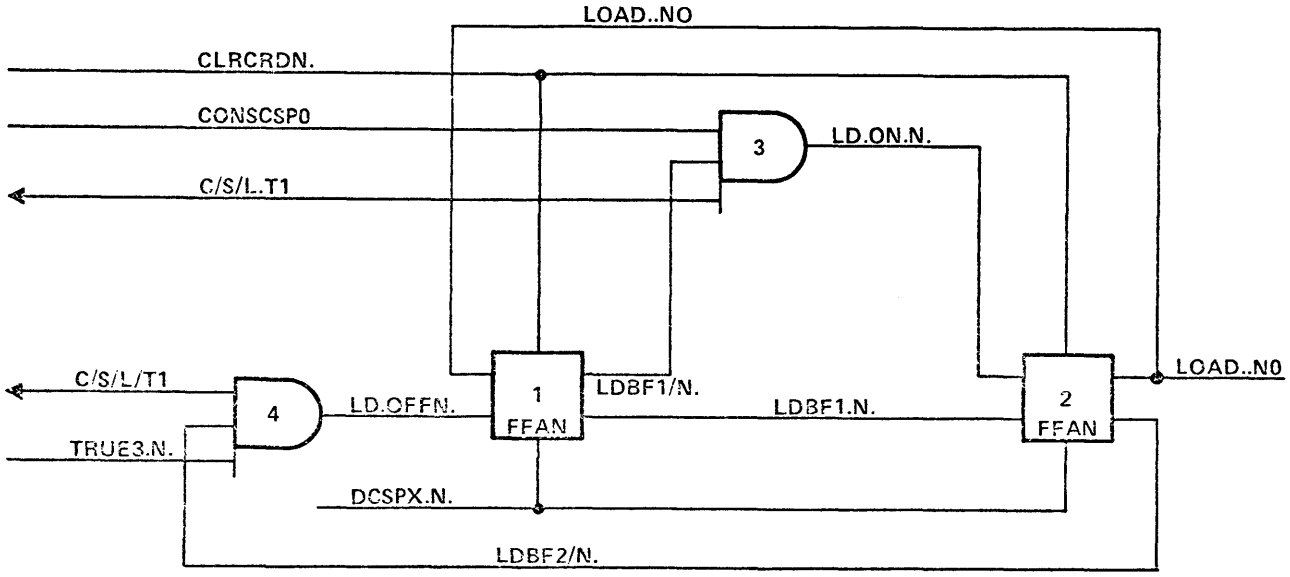


Figure 2-13. LOAD Switch Operational Logic

parity error during any S-memory cycle (other than a micro fetch) does not cause a halt. Such a parity error is reported by setting bit 3 of the CD register, and is handled as determined by software. Also, detection of parity errors on micro fetches from M-string memory is not possible because parity generation and storage are not incorporated in its design.

The general parity error signal, PARERRTO., can be generated either by logic in the PAPDIC circuitry, or by the cassette control logic. PARERRTO is generated by PAPDIC when reported by the port interchange, (PARERRH.), provided that a fetch cycle is occurring (FMPMOPH1). From the cassette control, PARERRTO is generated at sequence count zero time (SC=0..L1) when the error flip-flop is set (ERRORFL.). This can be caused by several conditions. The appearance of PARERRTO immediately generates STEP..P., which forces the current state generation logic to return to the console current state (halt) at the end of the fetch operation. At the next clock pulse following the appearance of PARERRTO, flip-flop A is set, generating PE3...L1 (parity error). The PE3...L1 signal, in turn, causes generation of TPPE..L1, illuminating the PARITY lamp. Flip-flop A remains set until reset by the general processor clear (GPCLR.L1) signal. Refer to figure 2-14.

CASSETTE ON/OFF SWITCH

CASSETTE ON/OFF is the power ON/OFF switch for the cassette drive.

BOT INDICATOR

The beginning of tape (BOT) indicator is lit when the cassette in the tape drive is at the beginning (or end) of the tape. This condition is detected by a photocell which is activated by a light shining through the clear leader at the ends of the tape. The signal from the drive is TPRL/..1, which is inverted to produce BOT...L1. This signal drives the lamp directly.

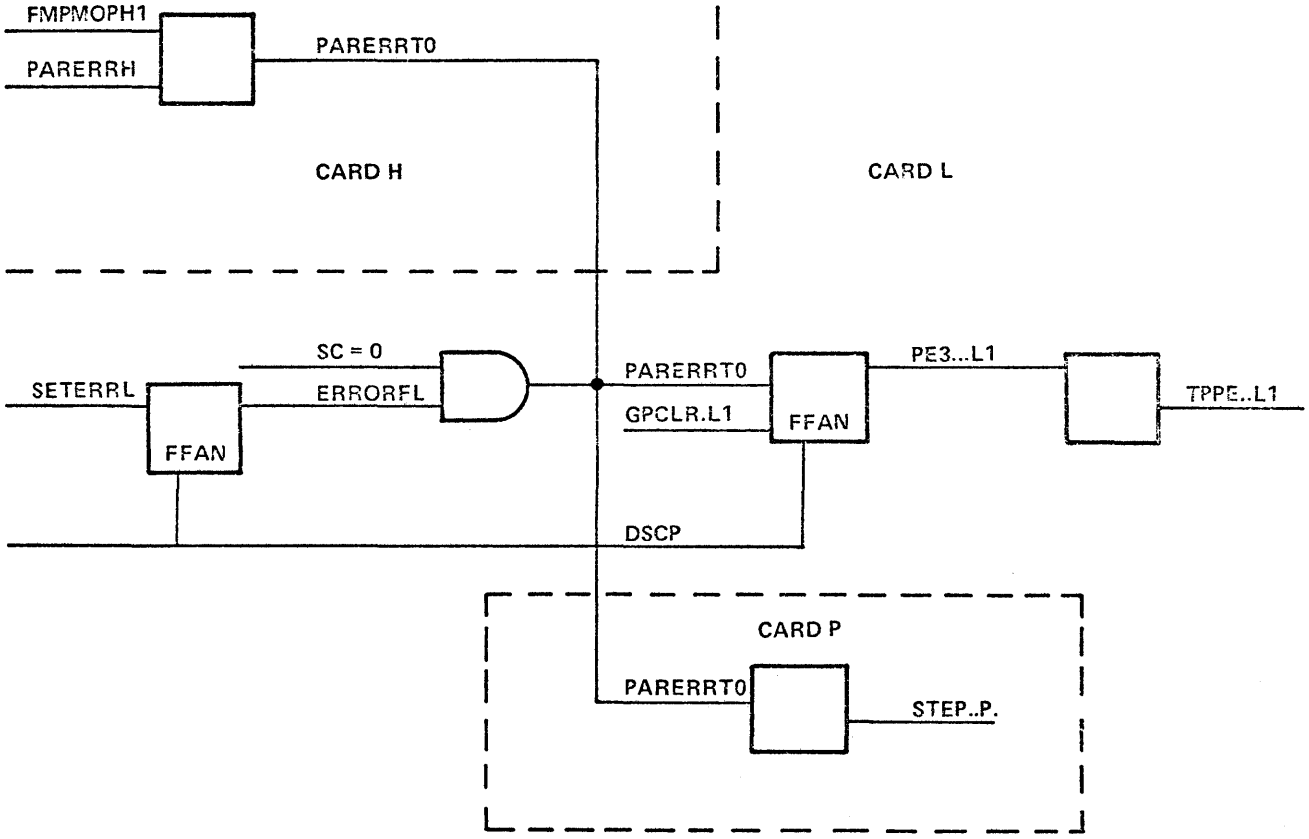


Figure 2-14. PARITY Lamp Operational Logic

REWIND PUSHBUTTON

The REWIND pushbutton when pressed, causes the tape in the cassette drive to return to the beginning of tape via a high speed rewind. The switch itself provides an enabling signal (RWD...T1) which activates the rewind control logic as shown on figure 2-15. RWD...T1 is ANDed with the reset output of the forward drive flip-flop, which means that rewind is allowed whenever the tape is not being driven in the forward direction. The output TWRP..L. (tape rewind pulse) is inverted and fed directly to the cassette drive. This is an asynchronous control level which remains in the active state for as long as the REW button is held pressed.

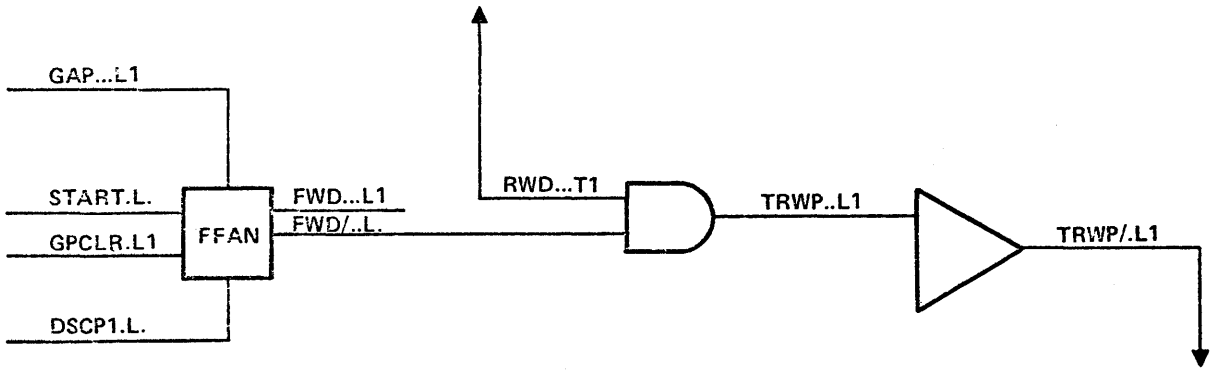


Figure 2-15. REWIND Switch Operational Logic

SYSTEM CLOCK

The B 1720 Series System Clock consists primarily of a 6-megahertz crystal controlled oscillator followed by adjustable delay lines and a buffered distribution system. Three outputs are produced which are used throughout the system to synchronize logic activity. These are the early clock, system clock, and adjustable clock which bear a timing relationship to each other the same as the order in which they are listed. In addition to selectable delay times, features of the system include provisions for use of an external oscillator, and the capability of single pulse operation on approximately half of the system and early clock outputs. An overall schematic diagram of the clock system is shown in figure 2-16.

CONTINUOUS CLOCK OUTPUTS

Provided the INTERNAL/EXTERNAL switch is set on INTERNAL, the clock oscillator output is fed to delay line F2 by way of buffer 2. Note that placing the switch to EXTERNAL enables buffer 1, activating the external input, and simultaneously disabling buffer 2 by way of inverter 3. In DL5N "A", the source clock signal is delayed a total of 10 nanoseconds (as determined) by jumper chip "B", and is supplied as the early clock to the clock drivers by way of buffer 4. Additional outputs from the delay line are supplied to jumper chip "C", where a variety of delayed clocks may be chosen to drive gate 5, the output of which is known as the adjustable clock. This signal is used exclusively by S-memory, and is usually delayed by an additional five nanoseconds. Note that in addition to the outputs of A, the further delayed outputs of delay line D are available at C. These provide a wide range of selectable delays from the system clock. A fixed output from D is used to provide the clock signal (CRTCLKO) for the single pulse circuit. This signal is delayed by 15 nanoseconds from the system clock. The clock drivers which can be used in a single pulse mode are controlled by this circuit, which serves only to enable them. Therefore, in the normal mode, the system clock output from the single pulsed clock drivers is identical to that from the other system clock outputs. In normal, the single pulsed outputs are enabled by a true level from buffer 6. The remaining system clock outputs are always enabled by a constant true from inverter 7. Refer to figure 2-17 for an illustration of continuous clock timing.

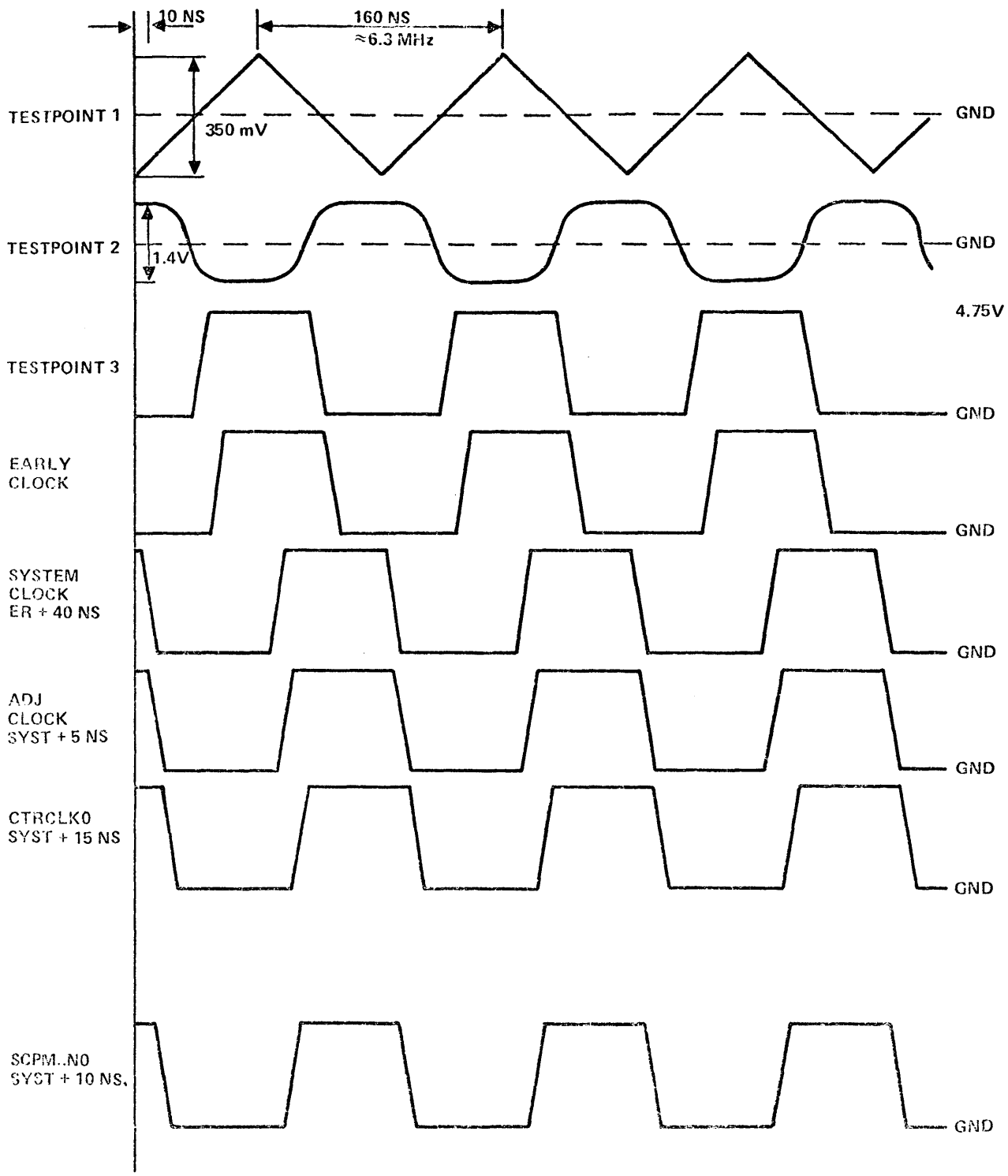


Figure 2-17. Continuous Clock Timing-Chart

SINGLE PULSE CLOCK

In the single pulse mode, four of the early clock drivers and six of the system clock drivers can be caused to (simultaneously) pass a single clock pulse. This is accomplished manually by pressing the SINGLE PULSE pushbutton. The single pulse circuit consists of a group of flip-flops which control the enable outputs from buffers 6 and 8. In the normal state, FFAN's F, G and H are reset. With neither input to inverter 9 true, the set output of FFAN F (which operates in the D-set mode) is true. This enables buffer 6, generating the normal enable signal for the single-pulsed clock drivers. Selecting the single pulse mode disables this signal by switching both the output of inverter 9 and the set output of F false. The same result may be achieved by applying a true to the SYNC-STOP banana plug (SYNS...0.), which is the other input to the inverter.

Pressing the SINGLE PULSE pushbutton sets FFAN E at the next CNTRCLKO pulse, providing a true output from buffer F1-1. With FFAN E set, the next CNTRCLKO pulse sets FFAN G, causing its reset output to revert to false. This disables buffer 8. Therefore, the 10 single-pulse clock drivers were enabled for one cycle of CNTRCLKO, which is just sufficient time to allow a single system clock pulse to pass to their outputs. The timing relationship of this sequence is illustrated in figure 2-18. Releasing the pushbutton resets FFAN E at the next CNTRCLKO pulse, which in turns resets G at the following clock, restoring the circuit to resting condition. It is also possible to use an external Single Pulse switch. Connecting the external switch presets FFAN H, disabling the local single pulse switch by removing the true signal (INSIC.0) from it.

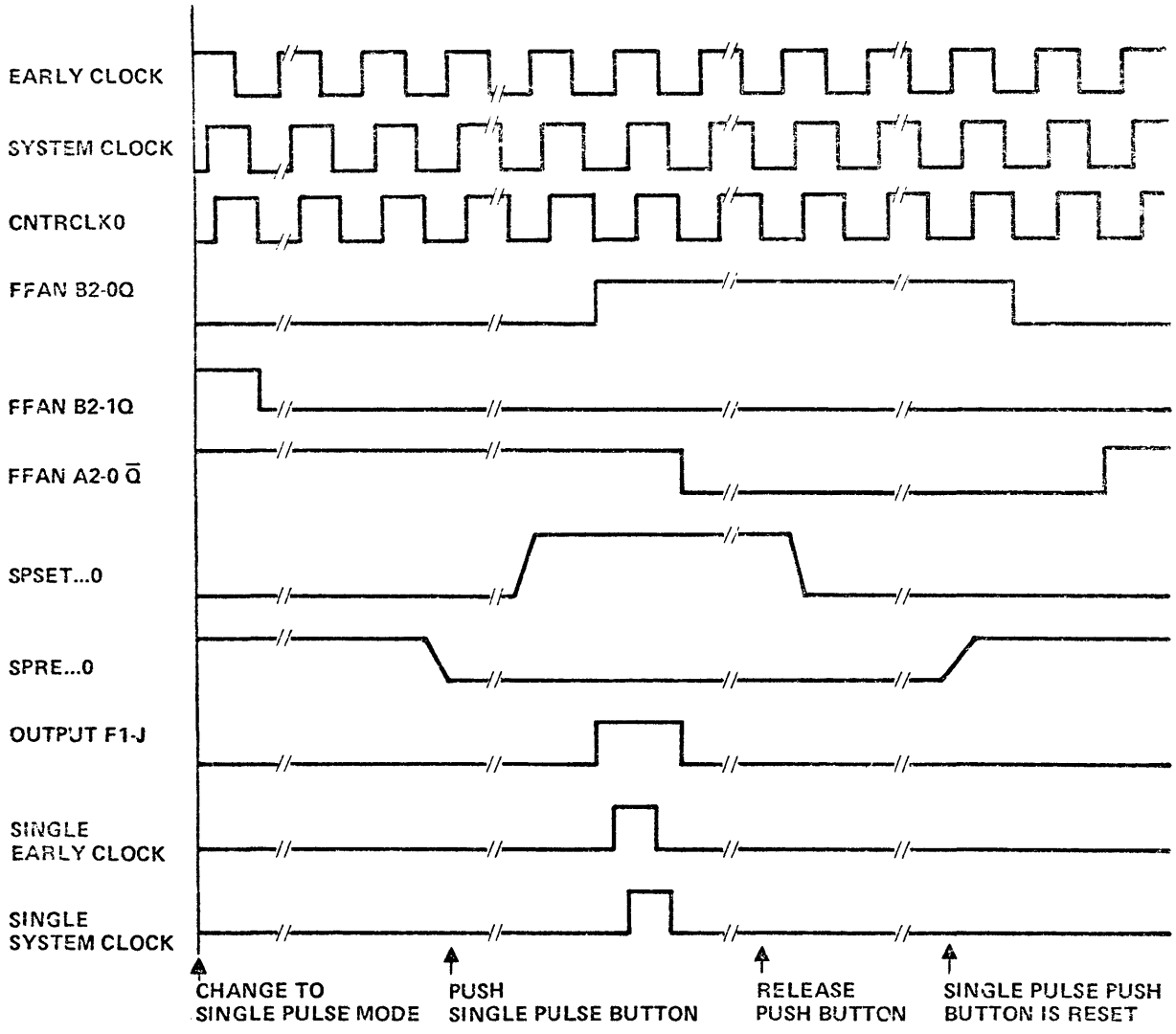


Figure 2-18. Single Pulse Clock Timing Chart

CLOCK DISTRIBUTION

The various clocks are brought via coax cables to the different parts of the central system. (Refer to figure 2-19.) The adjustable clock goes to main memory where it is used to develop all the necessary memory timing signals. The early clocks go to the I/O distribution card, the port interchange control card C, and the processor card R, where they are further delayed and used as special clocks within the associated logic. System clocks go to the processor cards B, H and, R as well as to the port interchange control cards A and B where they are delayed another 10 nanoseconds until they are finally derived as SCPM..KO outputs and brought to the backplane for distribution to all the other logic cards and the M-string memory cards.

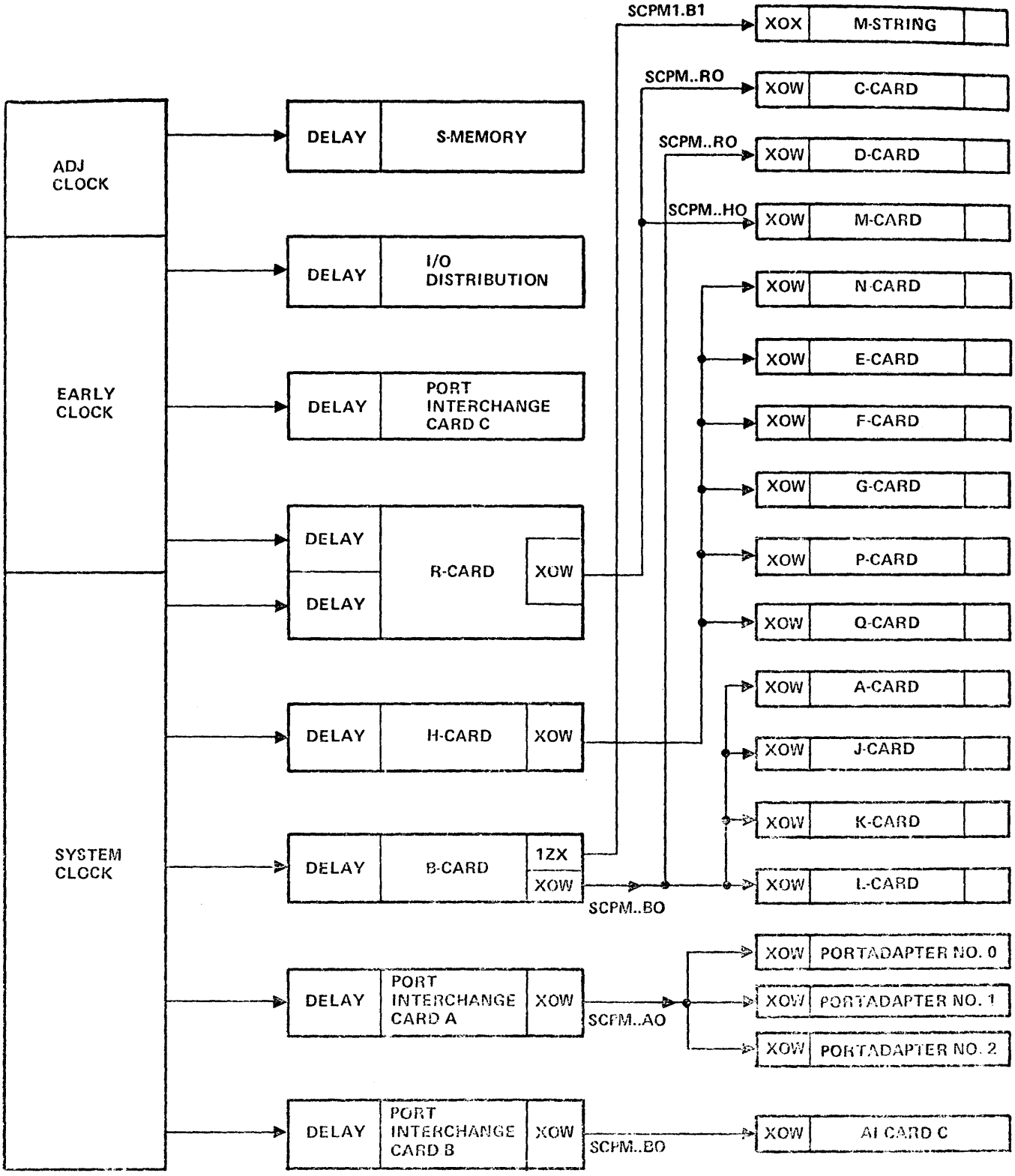


Figure 2-19. Clock Distribution Block Diagram

CURRENT STATE LOGIC

The current state logic is the most significant portion of the processor, because it exerts the major controlling influence over the operational configuration which is allowed to exist in the various sections of the machine. All processor operations involve logic activity in one or more of the current states, and the machine is in one of the five defined states at all times. For this reason the B 1720 Series is known as a "finite state machine."

CURRENT STATES

The current states are each activated by a single control level which serves to enable the appropriate circuitry for performing the defined functions of the state. These control levels, by which the current states are known, may be defined as follows:

CONSCSP. - console current state. True only when the processor is halted. The load display functions of the console are enabled when CONSCSP. is true.

STRTCSP. - start current state. True for one clock pulse when the console START button is pushed. This is a transient state which conditions the processor logic to enter the execute current state, which follows. The cassette tape drive is started in this state when tape mode has been selected.

EXECCSP. - execute current state. The normal operating state of the machine wherein micros are gated from MSM to the M-register and executed. Micros gated to M from other sources (as a result of the micro being executed) may also be executed in EXECCSP.

FTCHCSP. - fetch current state. This state is entered when it is necessary to fetch a micro from some source other than M-memory (S-memory or the U-register). Fetch is unique in that it involves actions which duplicate certain micro functions without actually executing one. FTCHCS is, in a sense, a sub-state of EXECCSP, because it may only be entered from or exited to that state.

STOPCSP. - stop current state. A one clock length transient state which conditions the processor logic to enter the console current state, which follows. The cassette drive is halted in STOPCSP, if running

CURRENT STATE FUNCTIONAL DETAIL

The current state of the machine is stored in three flip-flops, BMAC1F (24), BMAC2F (23), and BMAC4F (22), which are basic micro and address control flip-flops, all operating in D-set mode when CHBGCSP1 (change basic control state) is true. Refer to figures 2-20 and 2-21.

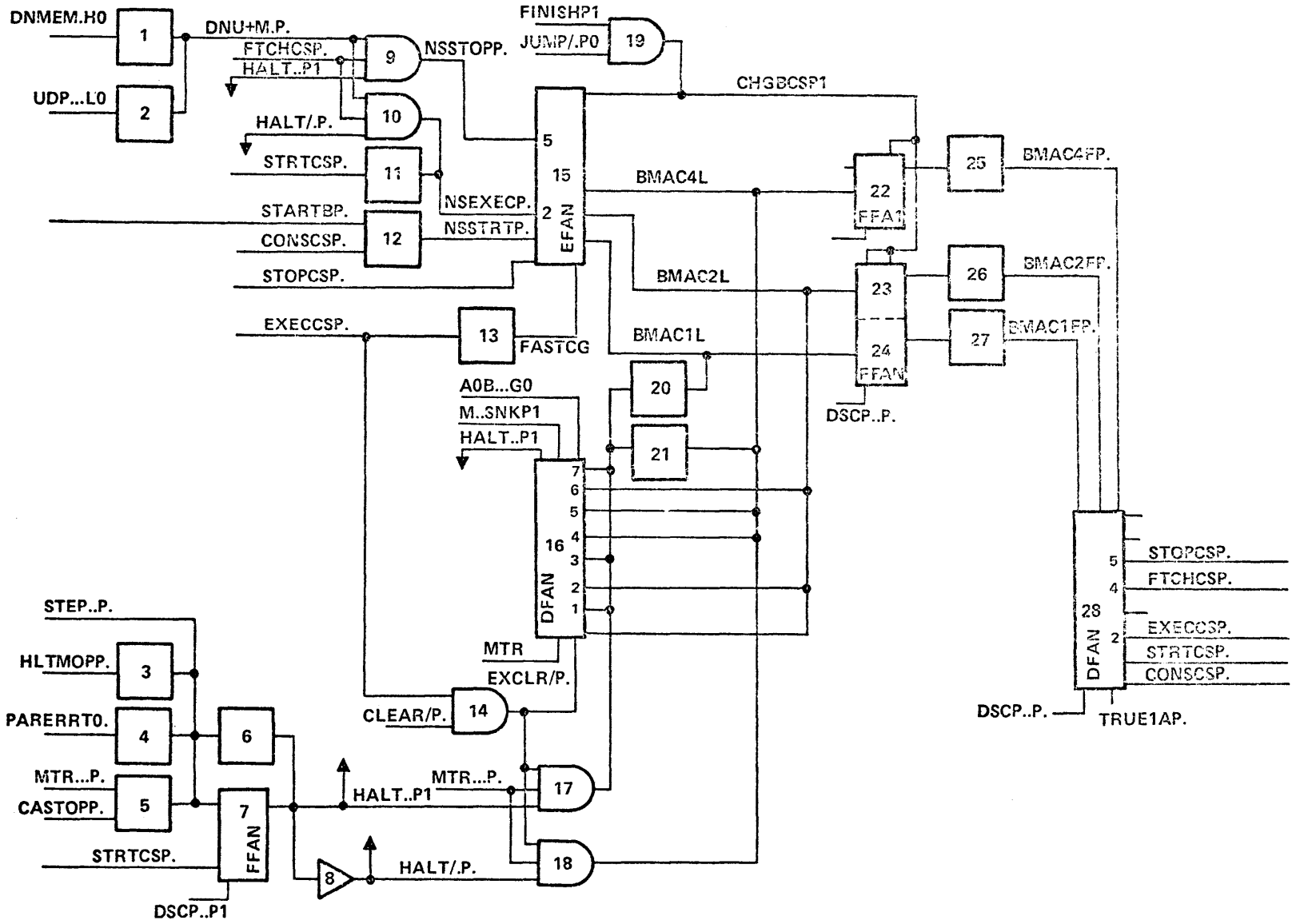


Figure 2-20. Basic Micro and Address Control Logic

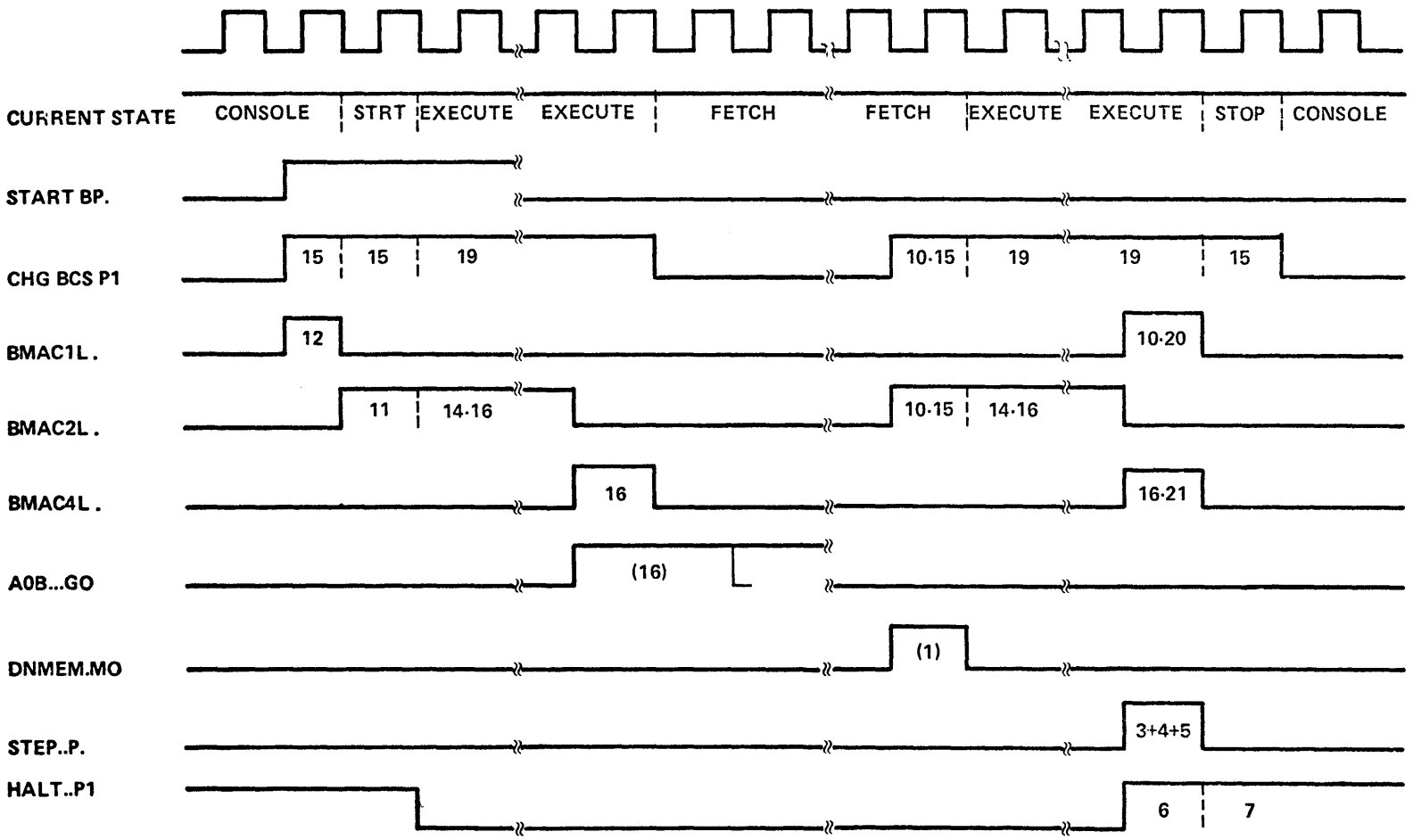


Figure 2-21. Current State Timing

After a power on, a clear, or a halt, the processor is in CONSCS with flip-flops 22, 23, and 24 all being reset, thus enabling the zero output of DFAN 28. At this time EFAN 15 is enabled since the buffer 13 output is false. Depression of the START button enables buffer 12, producing NSSTRT (next state start) true. With EFAN 16 enabled and NSSTRT true CHGBCS and BMAC1L. (basic micro and address control 1 level) are produced. The next DSCP sets flip-flop 24 which is decoded through DFAN 28 to give STRTCS.

STRTCS is ANDed with MTR (tape mode) to give CSTARTPO and DSETM, so that if the processor is in MTR mode the cassette drive is started and zeros are clocked into M-register.

STRTCS also goes to the reset input of the HALTF.F. (7) and to gate 11 to produce NSEXEC (next state execute), which is encoded through EFAN 15 to produce CHGBCS and BMAC2L. The next clock after entering STRTCS sets BMAC2F (23) (to decode EXECCS through DFAN 28), resets HALTF.F. (7), and (if in tape mode, MTR), starts the cassette drive, and loads zeros into M-register. See card P, pages 1 and 2.

With EXECCS now true the output from buffer 13 is true, EFAN 15 is disabled, and the output of buffer 14 is true which enables DFAN16 (if MTR is false). Provided there are no exception conditions AOB (A-out-of-bounds), M..SNK (M-register sink) or halt, the zero output of DFAN 16 is true which maintains a true on the D-input of flip-flop 23. Each time FINISHP1 is true (at the end of execution of each micro instruction) and JUMP/ is true (indicating that the next sequential micro is to be executed), CHGBCS is true from gate 19 putting flip-flops 22, 23 and, 24 in D-set mode. 23 remains set and 22 and 24 reset. The processor remains in EXECCS executing micro instructions and testing each time FINISH and JUMP/ are true for an exception condition.

Micro instructions are normally fetched from M-string memory in one system clock time while the previous micro instruction is being executed. If the A-register address is greater than TOPM then AOB is true (A-out-of-bounds) and the next micro to be executed must be fetched from S-memory. AOB true enables output #4 of DFAN16, producing the level BMAC4L. With FINISH true at the completion of execution of the current micro in the M-register, BMAC4F (22) sets. BMAC2F resets, and BMAC1F (24) remains reset. This allows FTCHCS to be decoded from DFAN28, EFAN15 to be enabled, and DFAN16 to be disabled. A fetch from S-memory takes place, and at the end of memory cycle DNMEM.HO (done memory) comes true and through gates 1 and 10 causes NSEXEC to become true (next state execute). As EFAN15 is now enabled the next DSCP sets BMAC2F (23) and resets BMAC4F (22) to decode EXECCS and allow execution of the micro instruction which has just been fetched from S-memory.

If the processor is in the tape mode, then DFAN16 is not enabled when entering EXECCS from STRTCS. With EXECCS true and MTR true, gate 18 is enabled, which forces BMAC4L true. Zeros were gated into M-register as the processor went from STRTCS to EXECCS. A one-clock no-op is executed, FINISH and JUMP/ are true so that BMAC4F (22) is set and BMAC2F is reset on the next clock pulse after entering EXECCS in MTR mode. FTCHCS is now true. The processor now waits while a micro instruction is read from the cassette tape and assembled in U-register. When a complete micro has been read into U-register, UDP...LO (U-data present) comes true. The micro instruction is gated through the MEX and into the M-register and, simultaneously, UDP...LO true gives NSEXEC true from gates 2 and 10. The next DSCP sets BMAC2F (23) and resets BMAC4F (22) to decode EXECCS from DFAN16. The micro read from tape, and now in M-register, is executed. At the completion of execution FINISH is true, again putting flip-flops 22, 23, and, 24 in D-set mode. Gate 18 forces BMAC4L true and BMAC4F sets and BMAC2F resets. The processor has now returned to FTCHCS and waits for the next micro instruction to be read into the U-register.

HALT LOGIC

The signal HALT..P1 may come true from several sources:

HLTMOPP.	=	Halt micro decode.
PARERRTO	=	Parity error.
MTR*CASTOPP.	=	MTR mode and cassette stop micro.
SNGMOD	=	Single step mode.
HALTB.P.	=	HALT pushbutton.
A=CSW*NULLREG*EXECCSP	=	A-register address equals console switches and null register and execute current state

Any of the above conditions brings STEP..P. true (figure 2-22) which, through gate 6, gives HALT..P1 true and puts a true on the set input of HALTF.F. (7). In run mode and EXECCS, gates 17 and 18 are disabled and DFAN16 is enabled. HALT..P1 true is decoded by the DFAN16 to make the number 1 output true. Through buffers 20 and 21, BMAC1L and BMAC4L are true. After execution of the current micro, FINISH and JUMP/ at gate 19 allow BMAC1F (24) and BMAC4F (22) to be set and BMAC2F (23) to reset. STOPCS is decoded from DFAN28. Stop current state forces CSTPAGPO true (card P, page 1) to stop the cassette drive at the next gap. EFAN15 is now enabled and, with STOPCS on the zero input, CHGBCS is true and all the BMACnL signals are false. The next DSCP resets BMAC1F, BMAC2F, and BMAC4F to put the processor in the console current state and enable the load/display functions. The processor can only return to EXECCS by depression of the START push button (via gate 12).

If the processor had been in FTCHCS when HALT..P1 came true then the fetch would continue until DNU+M came true. Through gate 9, NSSTOP comes true and the processor goes from FTCHCS to STOPCS to CONSCS, with the next in-line micro to be executed in the M-register. In MTR mode DFAN16 is disabled and is bypassed by gate 17 to force a similar function as when in the run mode.

If, in the run mode, EXECCS, FINISH, and HALT..P1 are true but JUMP/ is false, this would imply that a BRANCH or SKIP micro is currently being executed and the jump is to be taken. In this case one extra clock pulse is required to allow the new micro instruction at the "branch to" address to be fetched and the "branch to" address in A-register to be incremented by one to address the next in-line micro. JUMP/ is true at the extra clock pulse and the halt logic takes the processor to the console state as before with the next micro to be executed in M-register and A-register addressing the next micro to be fetched.

HALT (CONSOLE SWITCHES EQUAL A-REGISTER)

When null register is selected by the console dial settings and the processor is in run mode (and executing micro instructions), the processor will halt whenever the contents of A-register match the console switch setting (starting with the fifth switch position from the right and ignoring the left-most six positions). Card M (page 6) contains three CFAN's where a comparison is made: compare console switches CSW04 through CSW17 to A-register bits A00F through A13F.

When an equal comparison is found, A=CSW.M0 is true. This signal goes to card P (page 8) where it is ANDed with EXECCSP. and NULREGP. to make STEP..P. true and bring the processor to a halt. A-register/console switch alignment is shown in figure 2-23.

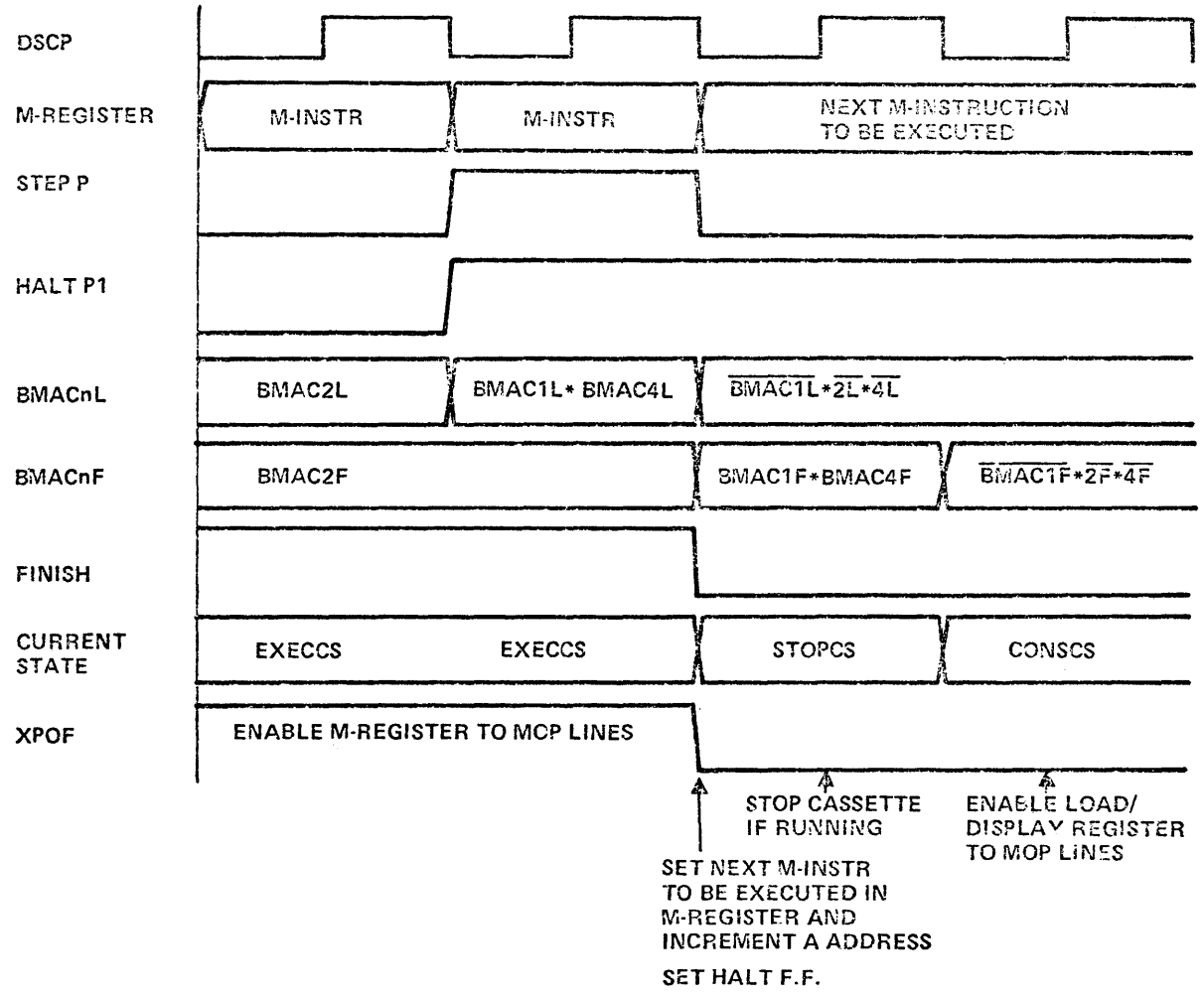


Figure 2-22. Halt Logic Timing

A-REGISTER

The A-register is the address register for M-string memory. It is 14 bits wide and consists of flip-flops. A's input comes primarily from the main exchange, from which it receives bits 04 through 17 only. This displacement is due to the fact that MSM is addressable in one word (16-bit) increments only, which is the binary weight of MEX bit 4. Refer to figure 2-23. The other inputs to A are from the increment logic and the relative displacement (branch) logic.

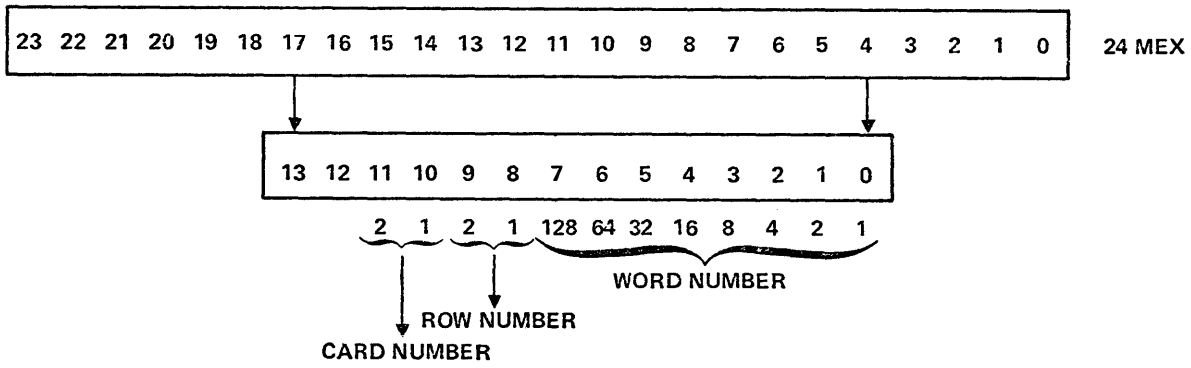


Figure 2-23. MSM Addresses in A-Register

A-REGISTER FUNCTIONAL DETAIL

Transferring the contents of the MEX to A is accomplished by the level A←MX..P0, which when true enables the input gates MEX04BT0 through MEX17BT0. Refer to figure 2-24. The levels DSETA0M. through DSETA3M. must be true to put the flip-flops in the D-set mode. In this mode, the MEX has control of these flip-flops. This occurs at the next clock pulse DSCP..M. The outputs of the flip-flops are continuously available through buffers to the MSM addressing logic. The flip-flop outputs are labeled according to the significance of each bit in the addressing of MSM. MSWA00M0 through MSWA07M0 indicate M-string word addresses 0 through 255 within the 256-bit storage chips which are used. Note that these bit positions have duplicate outputs labeled MSWA10M0 through MSWA17M0. The two sets of buffers contain identical addresses, and are required only because the physical addressing hardware on each MSM card is divided into two sections to reduce the load on the buffers. Outputs MSWA00M0 through MSWA07M0 feed the first two rows of M-string memory storage chips, and MSWA10M0 through MSWA17M0 feed the last two rows. The outputs MSR00M0 and MSR01M0 indicate M-string (chip) row addresses, and are used to enable the appropriate chip row. They are likewise duplicated by MSR10/M0 and MSR11/M0 which are used in the same manner (duplicate outputs) as the word addresses. The two remaining bits are MSCA0.M0 and MSCA1.M0 which indicate the M-string card address, permitting selection of one of the four (maximum) MSM storage cards.

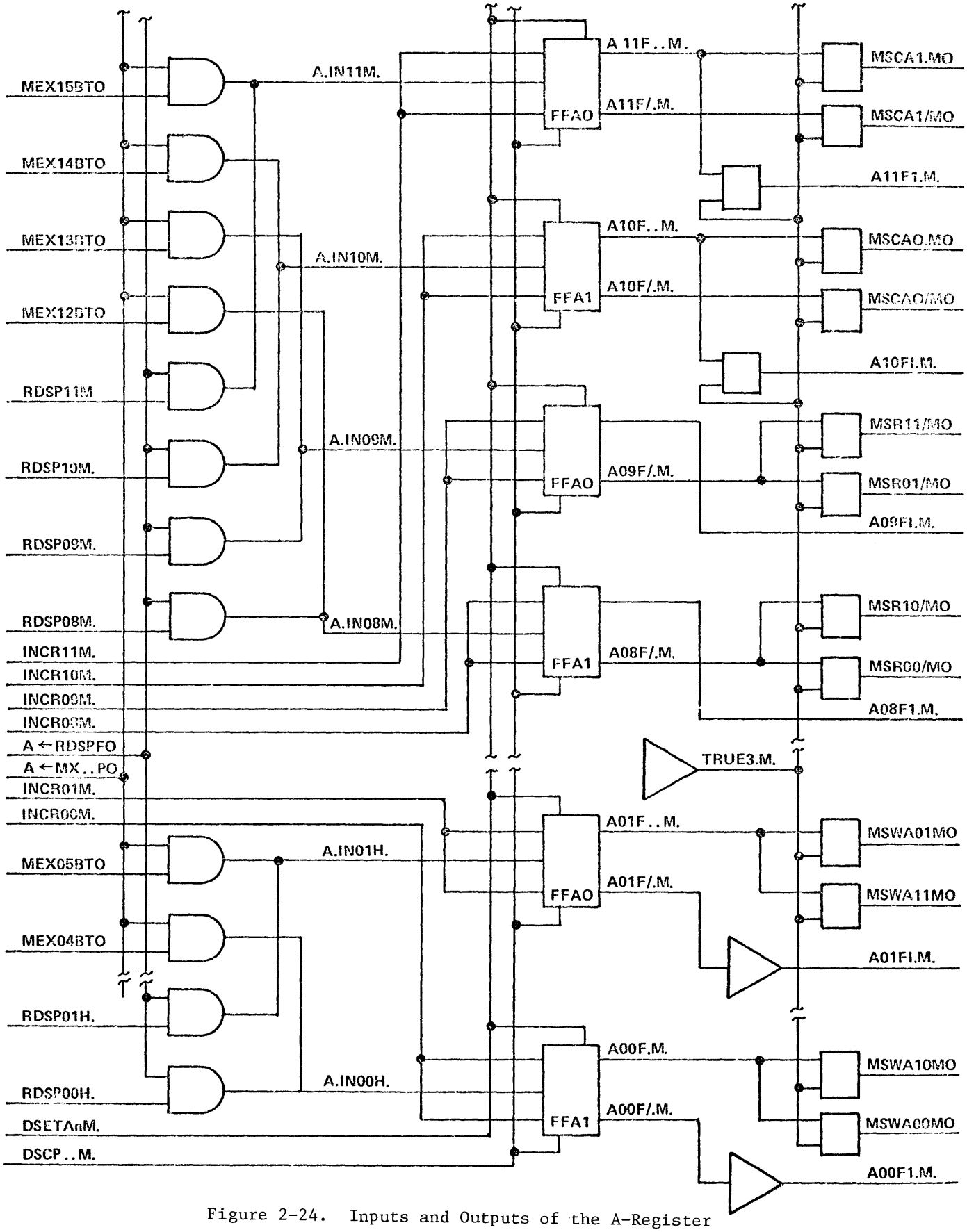


Figure 2-24. Inputs and Outputs of the A-Register

BRANCH LOGIC

The contents of the A-register may be changed by adding or subtracting the contents of MOP lines 0 through 3 (for a 4C or 5C micro) or MOP lines 0 through 11 (for a 123C or 145C micro). Refer to figure 2-25. In these cases the MOP lines mentioned contain the branch relative displacement magnitude. In addition to its outputs to the MSM addressing logic, the A-register produces several address bit outputs, among which is the group that feeds the relative displacement adder. These lines are known as A00F1.M. through A13F1.M. and constitute half the adder's inputs. The other half are the MOP lines 00 through 11 which contain the literal. Completing the loop back to the A-register are the AFAN adder chip SUM outputs RDSP00M through RDSP11M.

The operational mode of the adder is controlled by the signal RDMODEF0, which is true or false depending on the desired direction of the branch. RDMODEF0 being false causes the entire circuit to function as an adder, while true selects the subtract mode. Note that the inputs to the adder from MOP lines 04 through 11 are gated, and enabled by the signal 123C..F1. This signal is true only when a 123C or 145C micro is being executed, entailing the entrance of a 12-bit literal. Likewise, the level A←RDSPF0 (relative displacement to A) must be true to allow the outputs of the adder to enter the A-register.

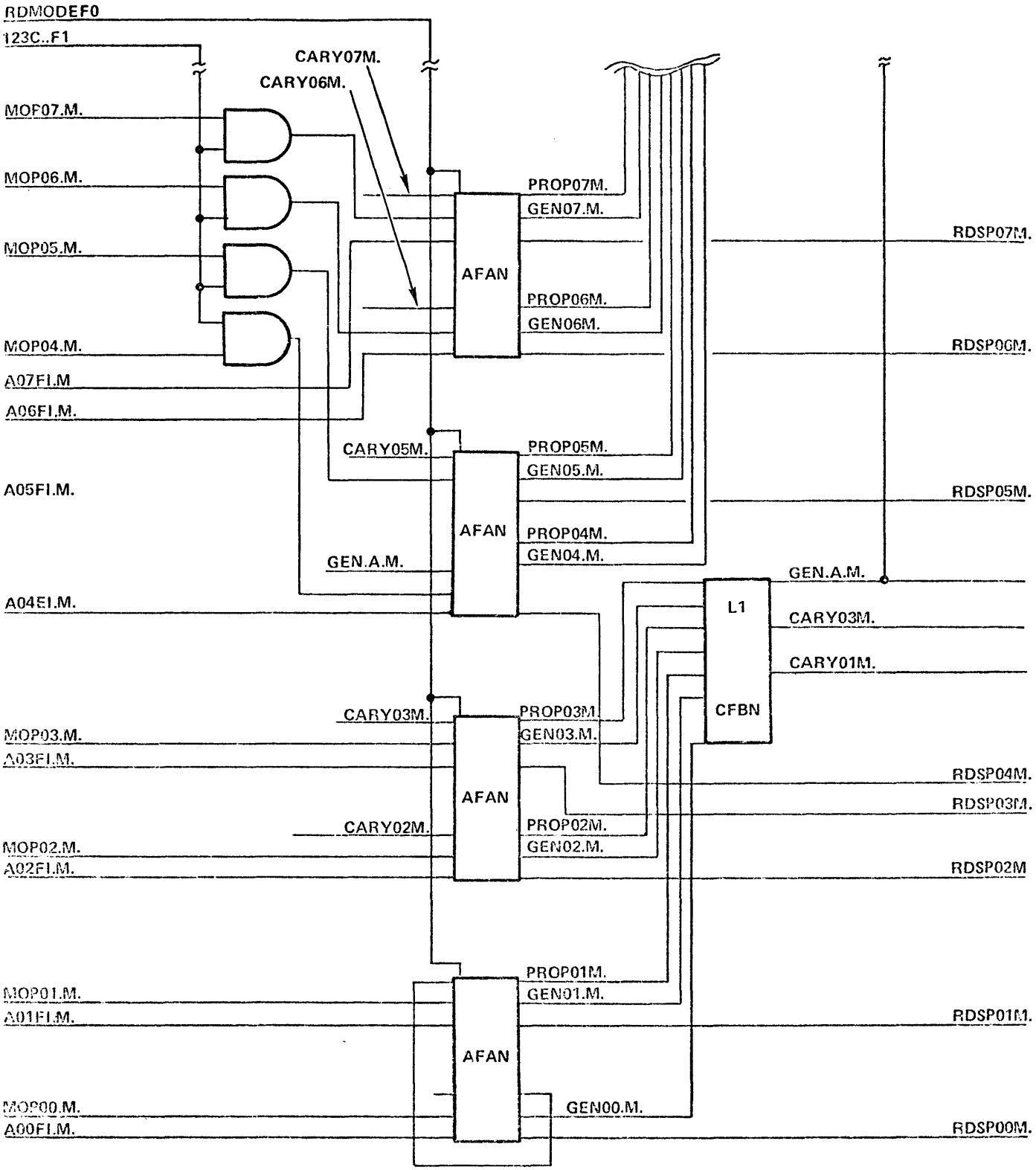


Figure 2-25. Relative Displacement Adder (Branch Logic)

INCREMENT A-LOGIC

The A-register may also be incremented by one (note that a one-address increment of A equals 16 bits in MSM). This is done by the increment A-counter illustrated in figure 2-26. The counter is preset to add a binary 1 to the contents of the A-register, and performs no other function. It uses as inputs the levels A00F1.M. through A12F1.M., which are derived from those outputs of the A-register flip-flops which are not used for addressing. For bits 00 through 07 these are the reset outputs of the flip-flop (inverted), while bits 08 through 12 use the set outputs. The incrementation function is automatic, being only the result of an ANDing process, and is continuously available. The counter output is enabled by the level INCA..P0, generating the levels INCR00M. through INCR13M. (increment address register). Since the levels DSETAnM are false except when A is being loaded from an external source, the flip-flops are normally operating in the J-K mode. Therefore, the appearance of INCR00M through INCR13M, which feed the J- and K- inputs in parallel causes the register to be upcounted by complementing (flip-flops change state when a 1-bit is present at both inputs). The result is not monitored and can wrap around.

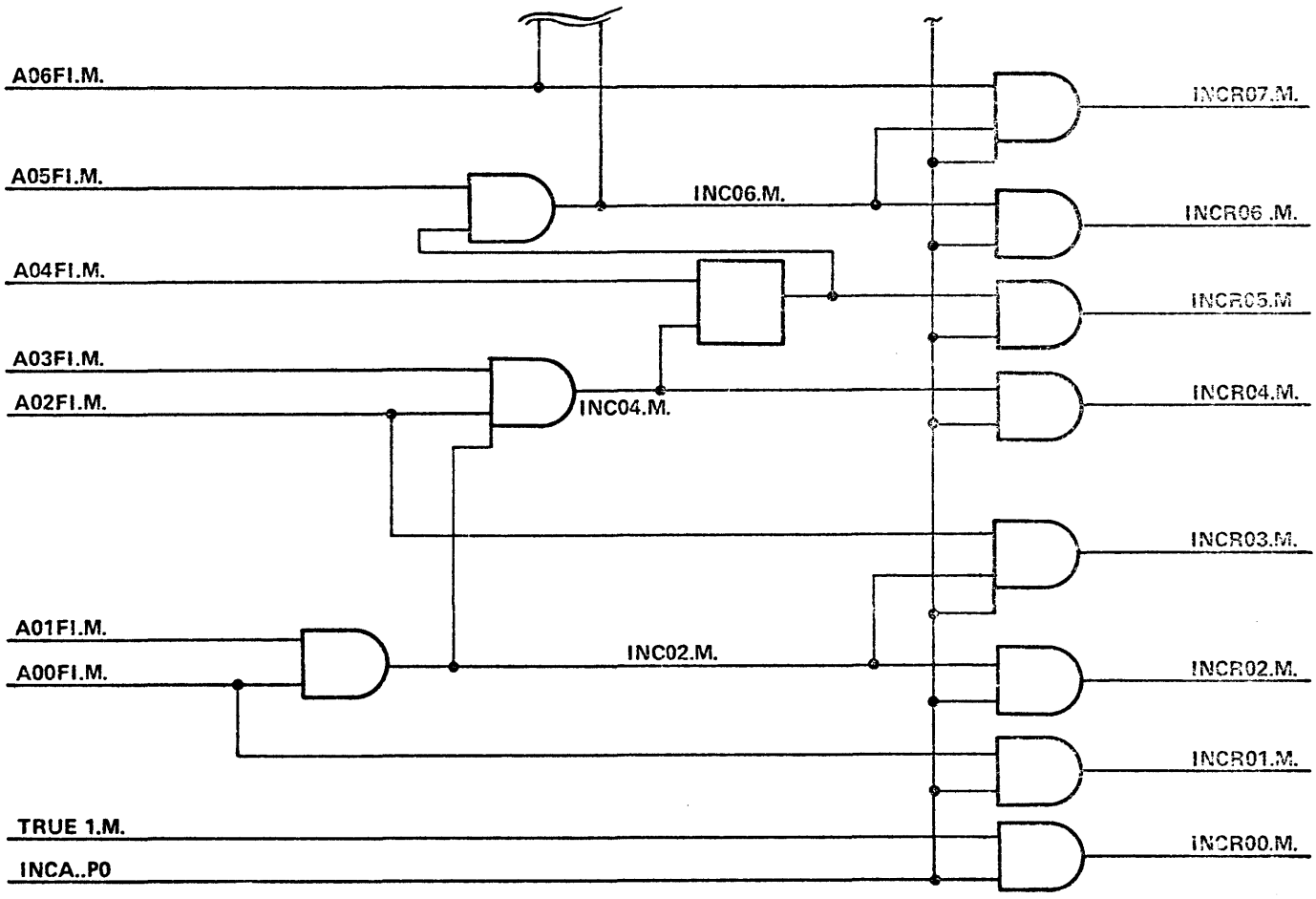


Figure 2-26. A-Register Increment Logic

MEMORY BASE REGISTER

The memory base register (MBR) is a conventional storage register utilizing RFBN chips, and is used exclusively to store a binary value for reference purposes. Refer to figures 2-27 and 2-28. This value is automatically added to the contents of the A-register (when the upper limit of M-string memory is reached). The result (A+MBR) serves as the S-memory address where the continuation of the micro string in MSM may be found. Associated with MBR is an S-fetch address adder which performs the actual addition of the contents of the two registers. This logic is active only when the level AOB..GO is true, indicating that the top of MSM has been reached. The overall purpose of the MBR-TOPM-AOB logic is to allow for the use of micro programs larger than the capacity of MSM. Access to MBR is from the 24-bit main exchange or the 4-bit result bus (for the four least-significant bits only). Output is to the S-fetch address adder. Note that MBR bits 00 through 04 bypass the adder, as they refer to a bit address and are unaffected by manipulation of A.

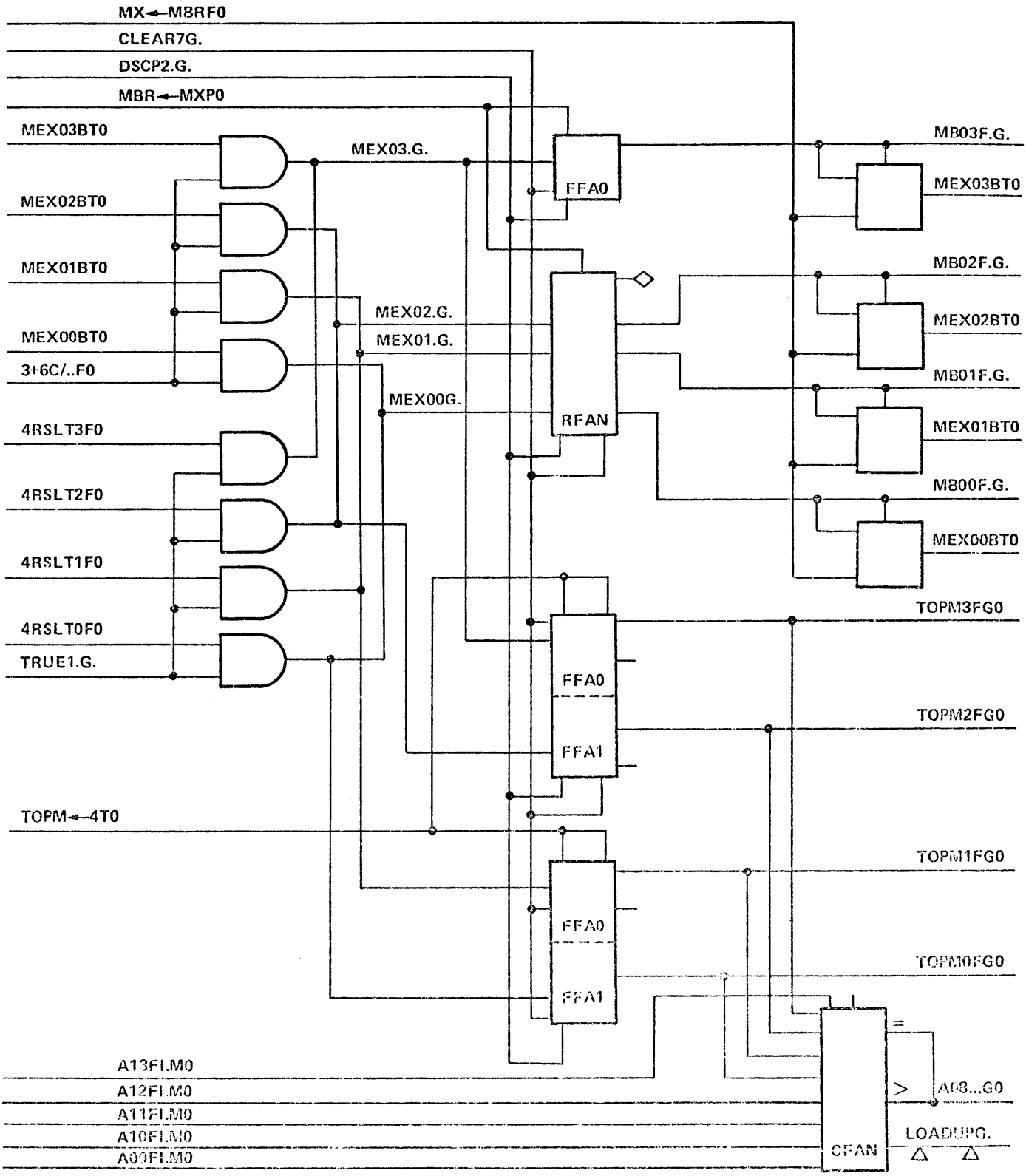


Figure 2-27. 4 LSB of MBR, TOPM Register and AOB

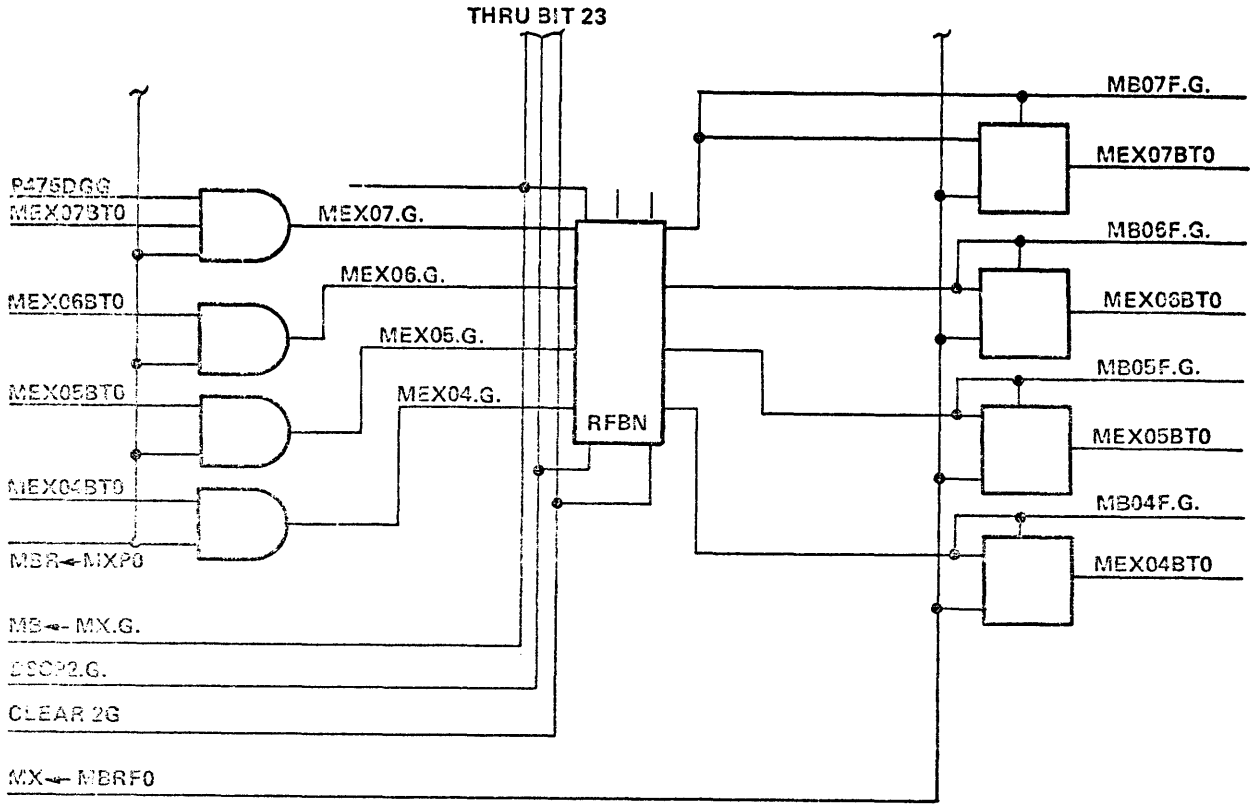


Figure 2-28. MBR Register Bits 04 Through 23

TOPM AND AOB

TOPM is a 4-bit flip-flop register which, like MBR, is used to store a reference binary value. Refer to figure 2-27. The contents of TOPM are equal in binary weight to bits 09 through 12 of the A-register, and are compared with them on a constant basis by the AOB (address out of bounds) comparator. Whenever the contents of A-bits 09 through 12 are equal to or greater than the value in TOPM, the level AOB...GO goes true, causing the contents of A and MBR to be added together. The sum (A+MBR) is gated to the PAPDIC logic as the S-memory address for the next micro fetch. See figure 2-29. Incrementation and counting up/down of A continues in the normal manner, but it now addresses S-memory rather than MSM.

Inputs to TOPM are from the 4 least-significant bits of the MEX or the 4-bit result bus. TOPM←4T0 must be true to load TOPM in either case, with 3+6C/.F0 also required if the source is the MEX. TOPM's output is continuously available to the CFAN comparator chip, and this serves as the B input. Bits 09 through 13 of the A-register are also fed continuously to the CFAN, comprising the A input. The chip outputs "A equal B" and "A greater than B" are tied together to produce AOB...GO in either case. Note that there are five A-register inputs as opposed to the four from TOPM. This is to ensure that all binary combinations which may occur when A is greater than TOPM are covered. Note also that TOPM is preset to clear to a value of 1000 (the level CLEAR2G. feeds the set input of bit-position 3 flip-flop and the reset input of the others). This is equal to the maximum possible size of MSM. When TOPM is loaded programmatically, any lesser value may be chosen, at the option of the programmer. In addition to going to the comparator, TOPM's output may also be gated to the auxiliary 4-bit exchange.

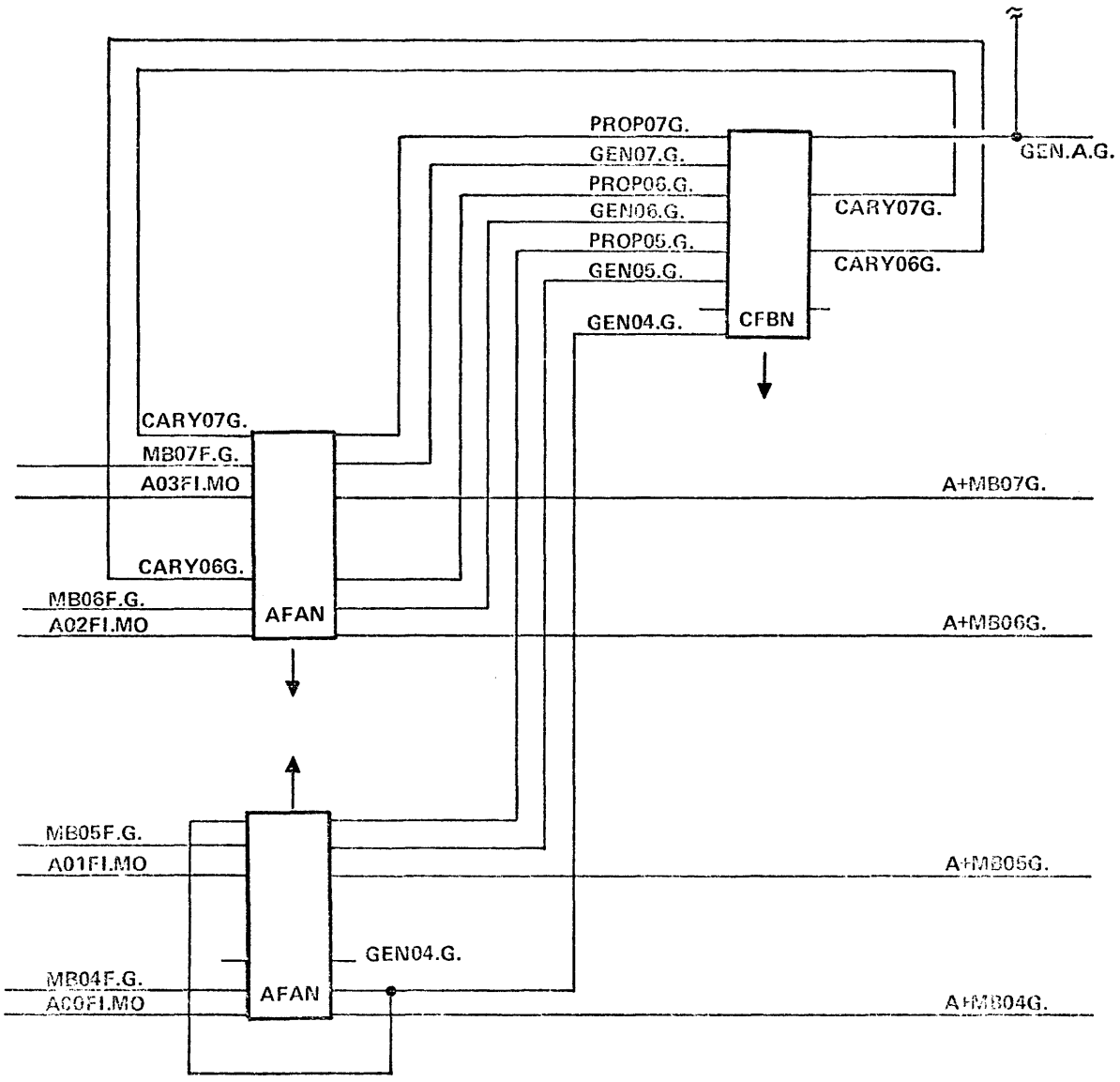


Figure 2-29. S-Fetch Address Adder (Bits 04-07 Shown)

FA REGISTER

The FA register is 24 bits in length and is used to store the S-memory address for read and write operations. In addition, it may be sourced or sinked as a general purpose 24-bit register for operations which do not involve memory access. For memory addressing functions it is possible to increment or decrement FA by binary values from 1 to 32, and to add to or subtract from its contents the value contained in a 24-bit literal from left scratchpad. These capabilities are provided by the count FA adder/subtractor and scratchpad relate FA

adder/subtractor which are included as an integral part of the register's logic. Inputs and outputs of FA are as follows:

Name	Purpose
Main exchange (input)	Load FA
Left scratchpad (input)	Load FA or add or subtract to value in FA
CPL or literal (input)	Control of incrementation/ decrementation
Main exchange (output)	Source FA
Direct path to PAPDIC logic (output)	Address S-memory
Direct path to left scratchpad (output)	Store contents of FA

FA REGISTER FUNCTIONAL DETAIL

The FA register (figure 2-30) is composed of RFAN 3-bit register chips which are divided into two groups. These groups consist of bits 00 through 05 (lower) and bits 06 through 23 (upper). The purpose of the division is to allow independent control of the chip operational mode for the two groups. Of the eight possible modes, only D-set, add, and subtract are used in this circuit. It is necessary to control the operating modes of the chip groups independently because the register must be configured differently for the various operations it can perform. The modes actually employed are shown in table 2-1. The significance of this is as follows:

Table 2-1. FA Chip Operational Modes

Operation	Upper Chip Mode	Lower Chip Mode	Notes
Load	D-set	D-set	Adders Inactive
Increment	Add	D-set	Count FA adder Active
Decrement	Subtract	D-set	Count FA adder Active
Scratchpad Relate Add	Add	Add	Scratchpad Relate Adder Active
Scratchpad Relate Subtract	Subtract	Subtract	Scratchpad Relate Adder Active

For a load operation, the value placed in FA is from an external source, and needs no modification. Therefore, the entire register may operate in the D-set mode.

For increment/decrement operations a literal value from 1 to 32 is added or subtracted to the value already in FA. For reasons of speed, this is done in an external high speed adder (count FA adder). This device has inputs from both the lower five bits of FA, and the CPL register (from which the literal is supplied). The add (or subtract) is performed within the count FA logic, and the result gated back to bits 00 through 05 of FA, which operate in the D-set mode. Carries or borrows generated by the increment/decrement are handled by the auxiliary carry logic which accompanies the adder. Note that only the carry/borrow inputs to the upper register chips are used during this operation.

During scratchpad relate add/subtract operations the value in FA is increased or decreased by the value contained in a 24-bit word of left scratchpad. This involves addition or subtraction performed by the RFAN chips themselves. Only the carries/borrows for bits 03 through 14 are determined externally, and this function is performed by the scratchpad relate adder/subtractor. This device is used to enhance the speed of the operation. The fact that only 14 bits (plus 1 carry) are accounted for by the adder limits the value which may be added or subtracted to 65,535.

Load

Loading FA from the MEX is enabled by the level FA←MX.RO and from left scratchpad by FA←PL.RO. Refer to figure 2-30. In either case, mode control lines LFAM0.RO and LFAM2.RO (lower group), and UFAM0.RO and UFAM2.RO (upper group) are false while LFAM1.RO and UFAM1.RO are true. This causes all chips to operate in the D-set mode. Gating the contents of FA to the MEX is brought about by MX←FA.RO coming true, and FA to left scratchpad by PL←FA.RO true.

Increment/Decrement FA

Incrementing FA occurs when the level CNTFA.RO (count FA) is true and FA+-MORO (FA add/subtract mode) is false. Refer to figure 2-31. Since the count FA adder outputs FA+-00C through FA+-05C. are continuously available, these levels serve only to enable gating. CNTFA.RO enables the gates admitting FA+-00C through FA+-05C. to the corresponding register inputs and, in combination with FAADD.C. (FA add mode), generates FA++..C. which enables the carry generation logic. The value contained in SPCPL0TO through SPCPL4TO determines the amount to be added to the FA register's contents. Note that the five least-significant bits of FA (FAR00LC0 through FAR05LC0) are also inputs to the adder. When this process of addition produces a carry, the level FACAR6C comes true, which enables the buffers associated with CNTFA.RO. and FAA++..C. This, in turn, produces FACI02C, which increments the upper register group by 1, and UPCI..C. which enables the outputs of the higher order carry logic. Additional operation of the upper register group in the add mode is caused by UFAM2.RO being true and UFAM0.RO and UFAM1.RO being false. Additional carries may or may not be produced, depending on the contents of the register. Note that incrementing FA by 16 (as when the console INC button is pressed) is done by forcing SPCPL4 true.

Decrementing FA is similar to incrementing, except that both the adder and upper chip groups operate in the subtract mode. This is caused by FA+-MORO being true in the case of the adder, and UFAM0.RO through UFAM2.RO true with UFAM1.RO false for the upper chip group. In addition to changing the mode of the AFAN chips of the adder, FA+-MORO enables generation of FA--..C., and by way of its complement, FAADD.C. (which is false), disables generation of FAA++..C. When the need for a borrow from a bit position higher than 05 arises, the level FACAR6C goes true, generating DOWNCIC. This, in turn, enables the borrow outputs of the higher order carry logic. Note that FACI02C is also generated, but is not significant because the bi-directional, carry/borrow terminal on the upper group RFAN (which it feeds) is not active as an input when the chip is in the subtract mode.

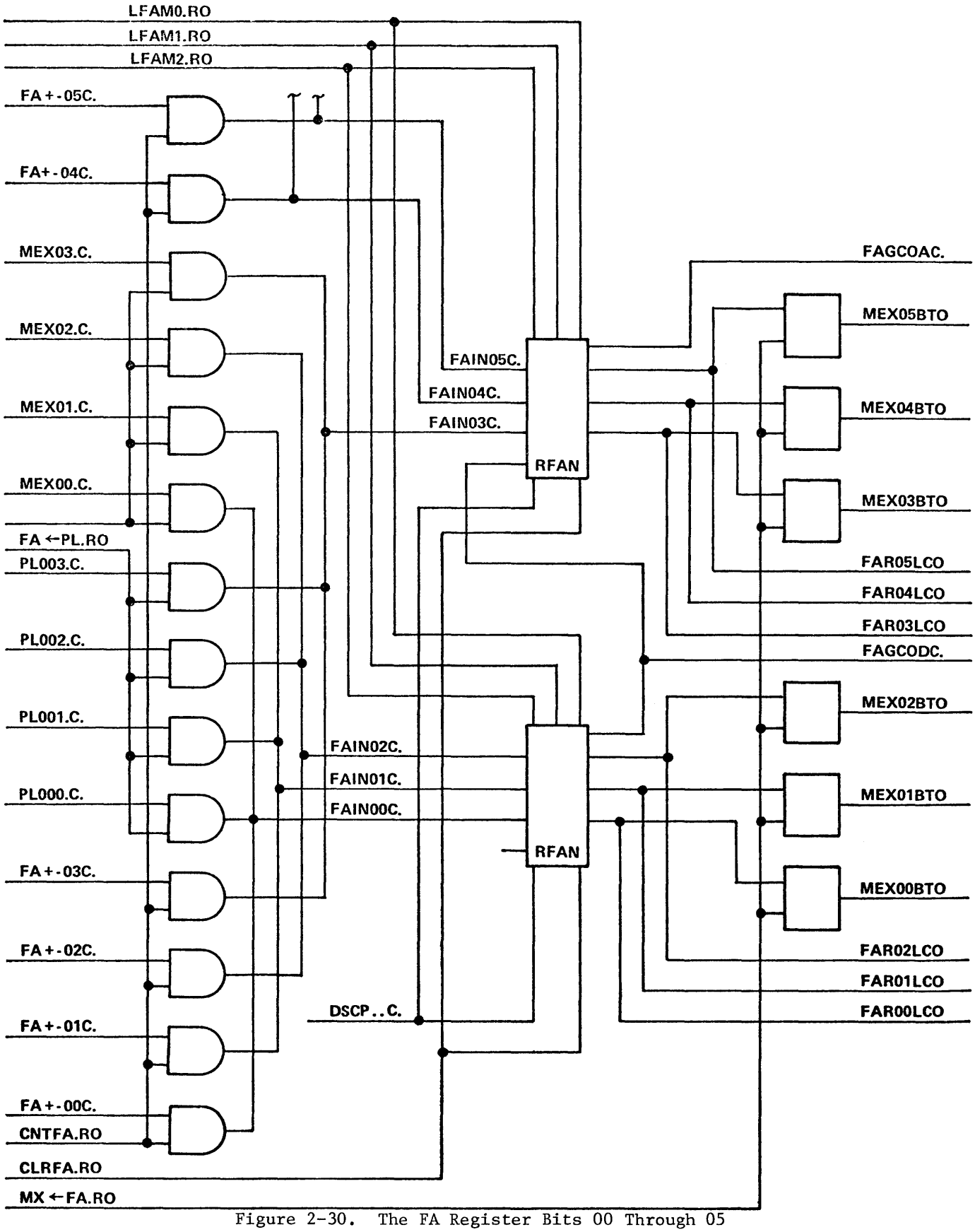


Figure 2-30. The FA Register Bits 00 Through 05

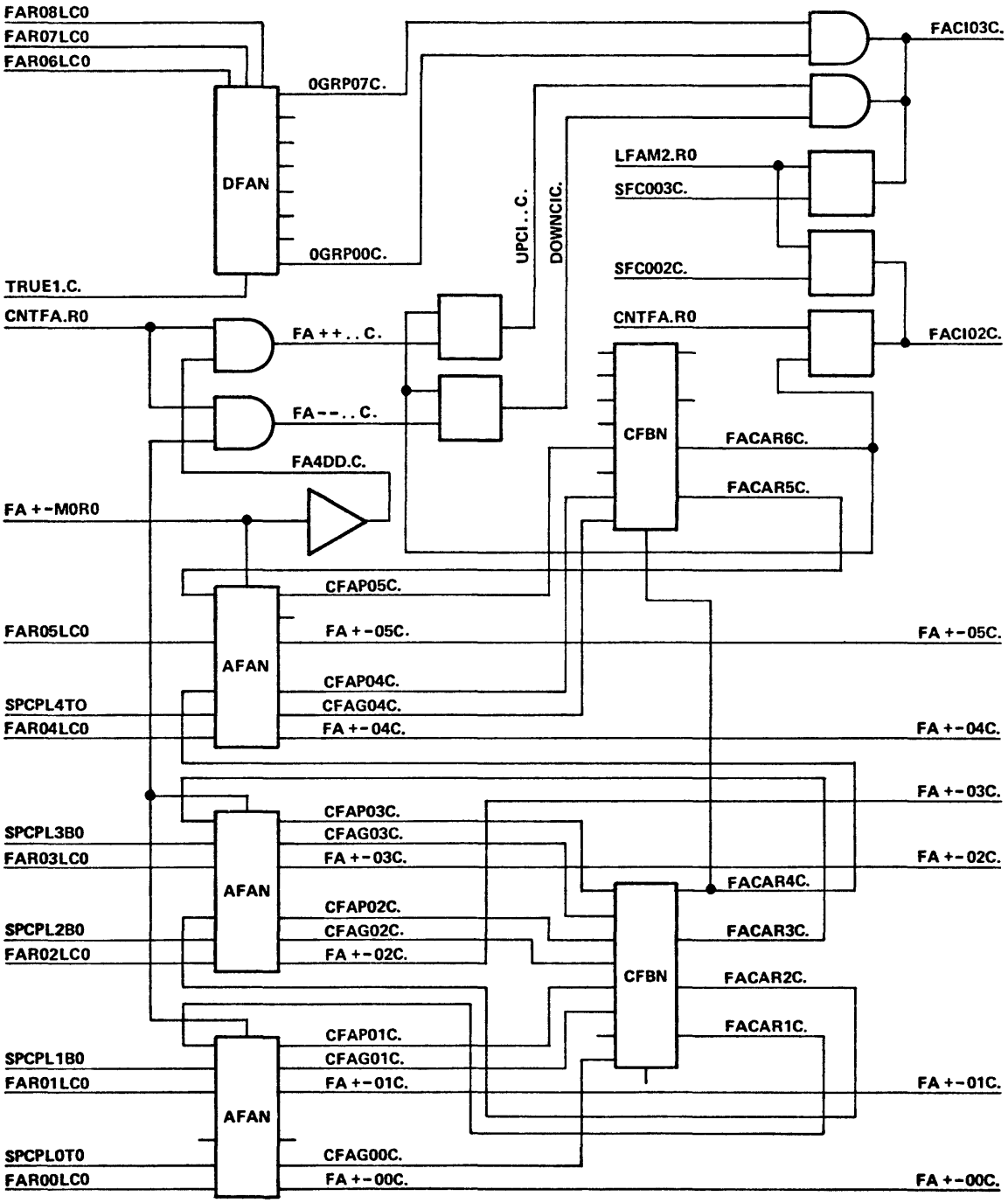


Figure 2-31. The FA Adder Bits 00 Through 05

Scratchpad Relate

The scratchpad relate FA adder/subtractor is active only during the 8D Micro. Refer to figure 2-32. It is not, in itself, a full adder/subtractor, but rather serves as the carry logic for the register itself when operating as a whole in the add or subtract mode. This logic is activated by the same levels which cause the lower group register chips to operate in add or subtract. LFAM2.RO, when true, enables all the carry outputs (FACI02C. through FACI07C.) while FAMD..RO, when true, changes the AFAN chips mode to subtract. This matches the mode change in the lower group RFAN register chips (both are now in subtract mode). Note that FAGC05C. and FAGC06C. bypass the carry logic. Note that FACI02C. through FACI07C. are created by ORing of several conditions. The gates which pass the scratchpad relate adder/subtractor outputs are shown on the count FA adder schematic (figure 2-31).

FB REGISTER

The FB register, like FA, is used to store information relating to S-memory operations. The values contained therein serve to describe the block of data in memory which is being dealt with, and as such serve as a reference only (they do not control any hardware directly). FB is addressable as a whole (24 bits), or in 4-bit groups denoted FU, FT, FLC, FLD, FLE, and FLF. In addition, the 16 least-significant bits are addressable as FL. FB and its sub-registers may be accessed as follows:

From/to 24-bit main exchange

From/to right scratchpad (direct path)

From 4-bit result exchange and to 4-bit auxiliary exchange

The 16-bit FL portion has an FL up/down counter associated with it, and may be incremented or decremented by any value from 1 to 32. This value comes from CPL or the literal contained in a micro instruction.

FB REGISTER FUNCTIONAL DETAIL

The FB register is composed of RFAN and RFBN register chips, plus several flip-flops. It is arranged in the configuration shown in figure 2-33. The somewhat unusual design of the FL portion was necessary to allow individual access to the 4-bit sub-registers within it, yet still permit FL to be incremented and decremented as a whole. To effect this, 4-bit registers (with an add/subtract capability) were created by joining an RFAN and a flip-flop. As with the FA register, control of the operational mode of the individual chips is used to configure the register for different functions. The modes employed are shown in figure 2-34. For load operations, all addressed portions of FB operate in the D-set mode because the values inserted come from an external source.

Incrementing or decrementing the FL portion involves modifying a value already present. As with FA, the five least-significant bits receive the output of an external adder/subtractor (FL count up and down counter), and therefore operate in D-set. The higher-order logic operates in add or subtract mode to account for carries/borrows. This includes the FLC and FLD flip-flops, which are connected so as to complement when a carry or borrow is received. Due to the variety of control levels involved, the functional description is broken down by operations.

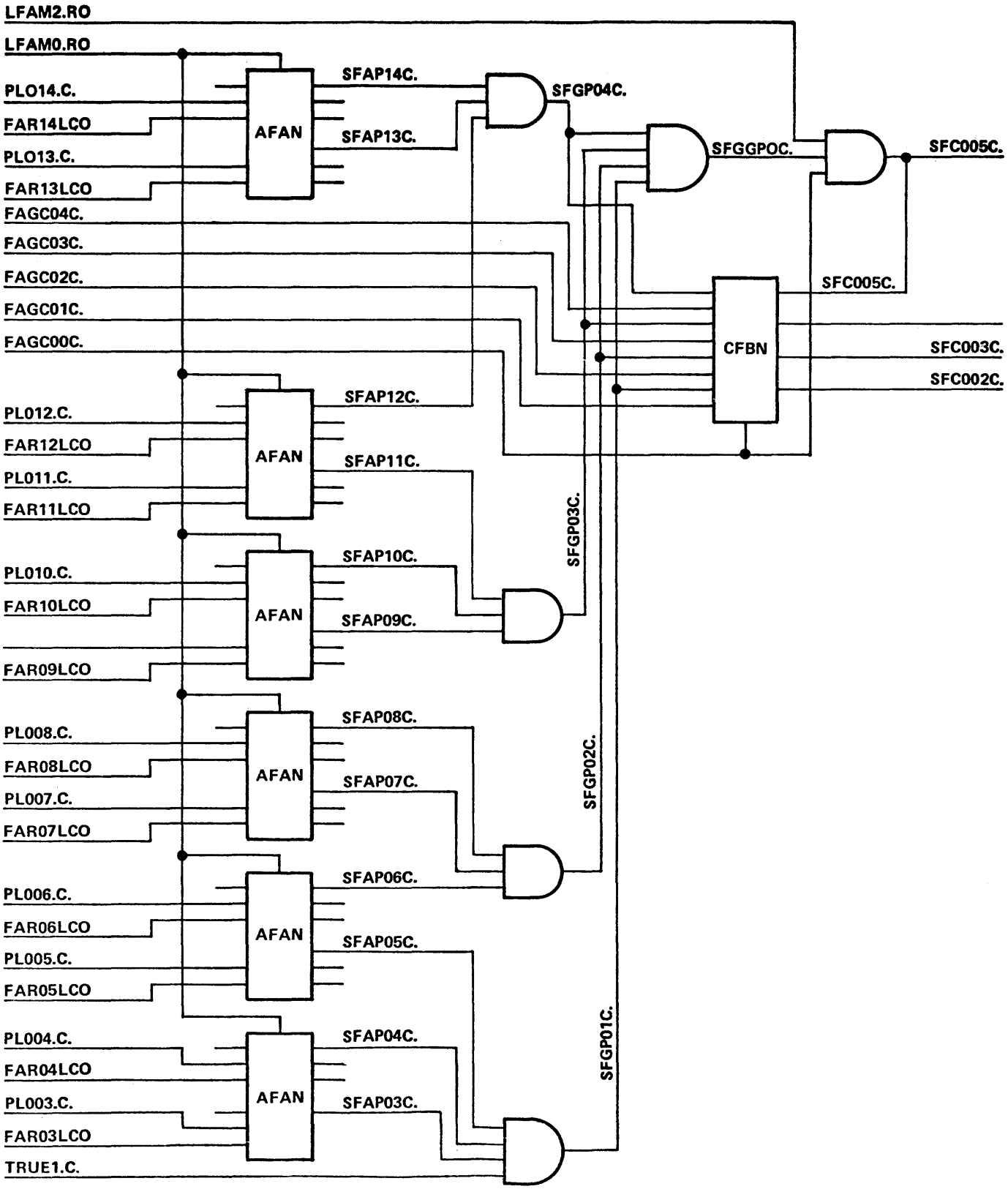


Figure 2-32. Scratchpad Relate Adder/Subtractor

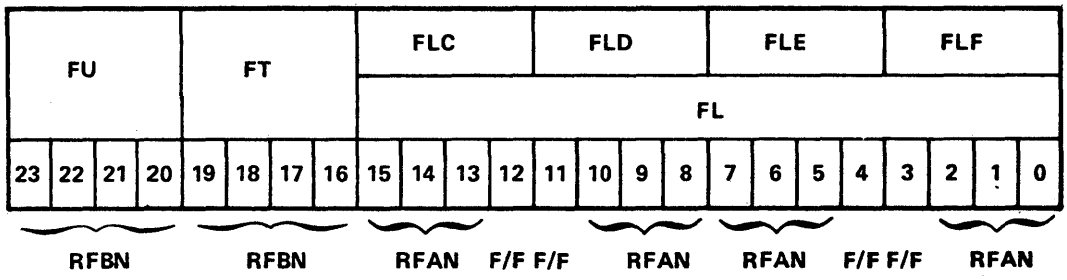


Figure 2-33. FB Register Operational Configuration

OPERATION	REGISTER PORTION									
	FU	FT	FLC		FLD		FLE		FLF	
	(RFBN)	(RFBN)	(RFAN)	(F/F)	(F/F)	(RFAN)	(RFAN)	(F/F)	(F/F)	(RFAN)
LOAD*	D-SET	D-SET	D-SET	D-SET	D-SET	D-SET	D-SET	D-SET	D-SET	D-SET
INCREMENT FL	N/A	N/A	ADD	J–K	J–K	ADD	ADD	D-SET	D-SET	D-SET
DECREMENT FL	N/A	N/A	SUBT	J–K	J–K	SUBT	SUBT	D-SET	D-SET	D-SET

*TRUE ONLY FOR THE PORTION(S) BEING ADDRESSED

Figure 2-34. FB Operating Modes

Load

To load FB (as a whole) entails use of either the MEX or right scratchpad inputs. Refer to figures 2-35 and 2-36. In the former case, the levels FL←MX.RO (main exchange to FL), FLF←MXD. (main exchange to FLF), and FB←MX.RO (main exchange to FB) must all be true. Three enabling signals are needed because it is possible to load FL only from the MEX, plus the fact that the four least-significant bits of the MEX are also used for entrance of the 4-bit result bus.

This latter condition requires that gating the MEX bits to the FLF portion be controlled separately. Loading FB from right scratchpad occurs when the levels LFB←PRD. (right scratchpad to lower FB) and UFB←PRD. (right scratchpad to upper FB) are true. The division is of no significance since both control levels follow the level FB←PR.RO.

Loading any 4-bit portion of FB from the 4-bit result bus is controlled by individual levels. As previously stated, 4RSLT enters the circuit by way of the 4 least-significant bits of the MEX, which is an available input of each 4-bit sub-register. Gating 4RSLT into the registers is as follows:

Register	Control Level
FU	FU←4BTD.
FT	FU←4BTD.
FLC	FLC←4BTD.
FLD	FLD←4BTD.
FLE	FLE←4BTD.
FLF	FLF←MXD.

In all loading operations, appropriate levels must be present to place the register elements in the D-set mode. These are:

Register Element	Control Level
FU	FUDSETD.
FT	FTDSETD.
FLC	FLCDSTD.
FLD	FLDDSTD.
FLE	FLE3M1D. + FLEFM1D.
FLF	FLFDSTD.

Note that FLE3M1D. and FLEFM1D. are both generated whenever FLEDSTD. or LFB←PRD. is true. This latter signal also generates all the others listed above.

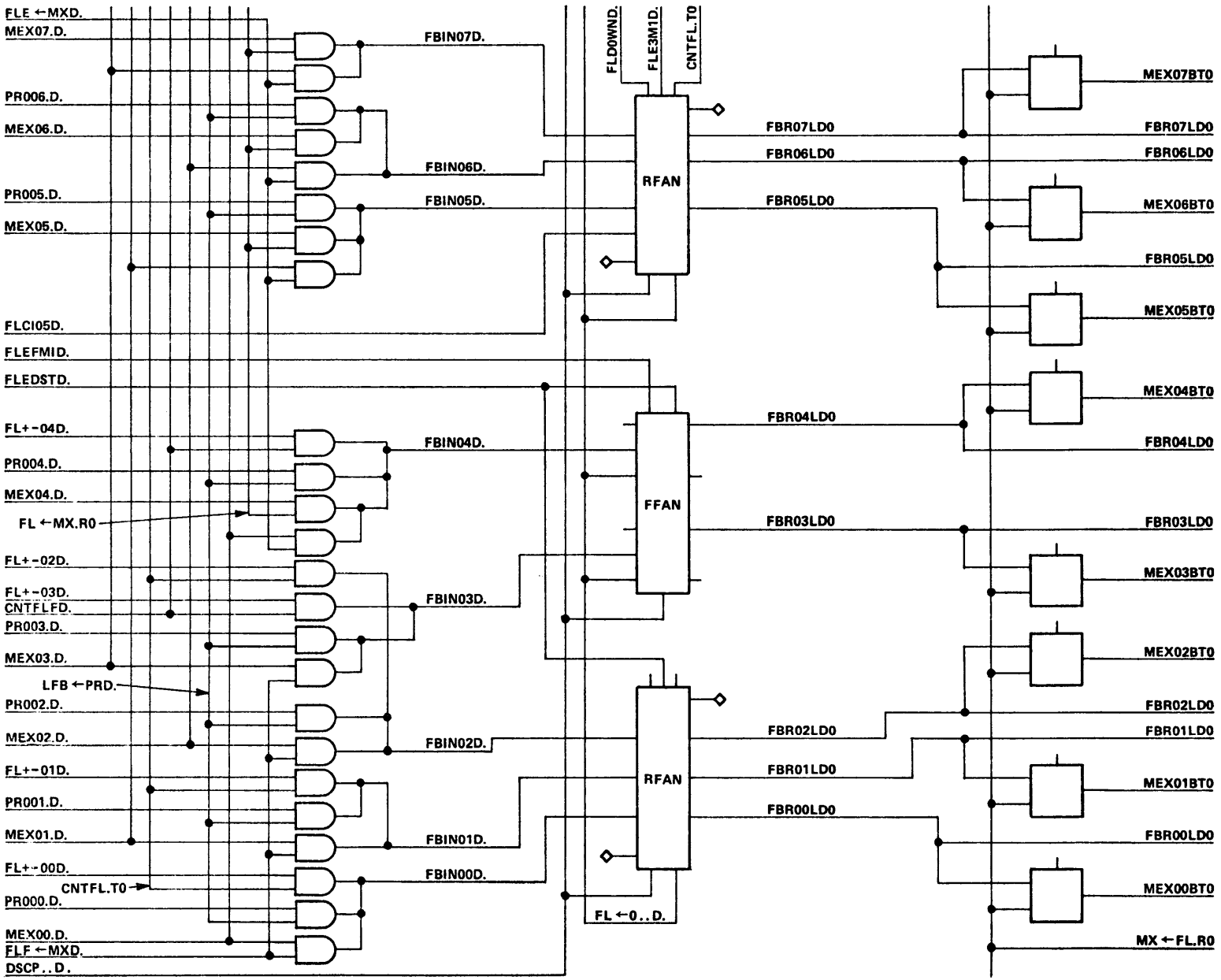


Figure 2-35. FL Register (Bits 00 - 07 Shown)

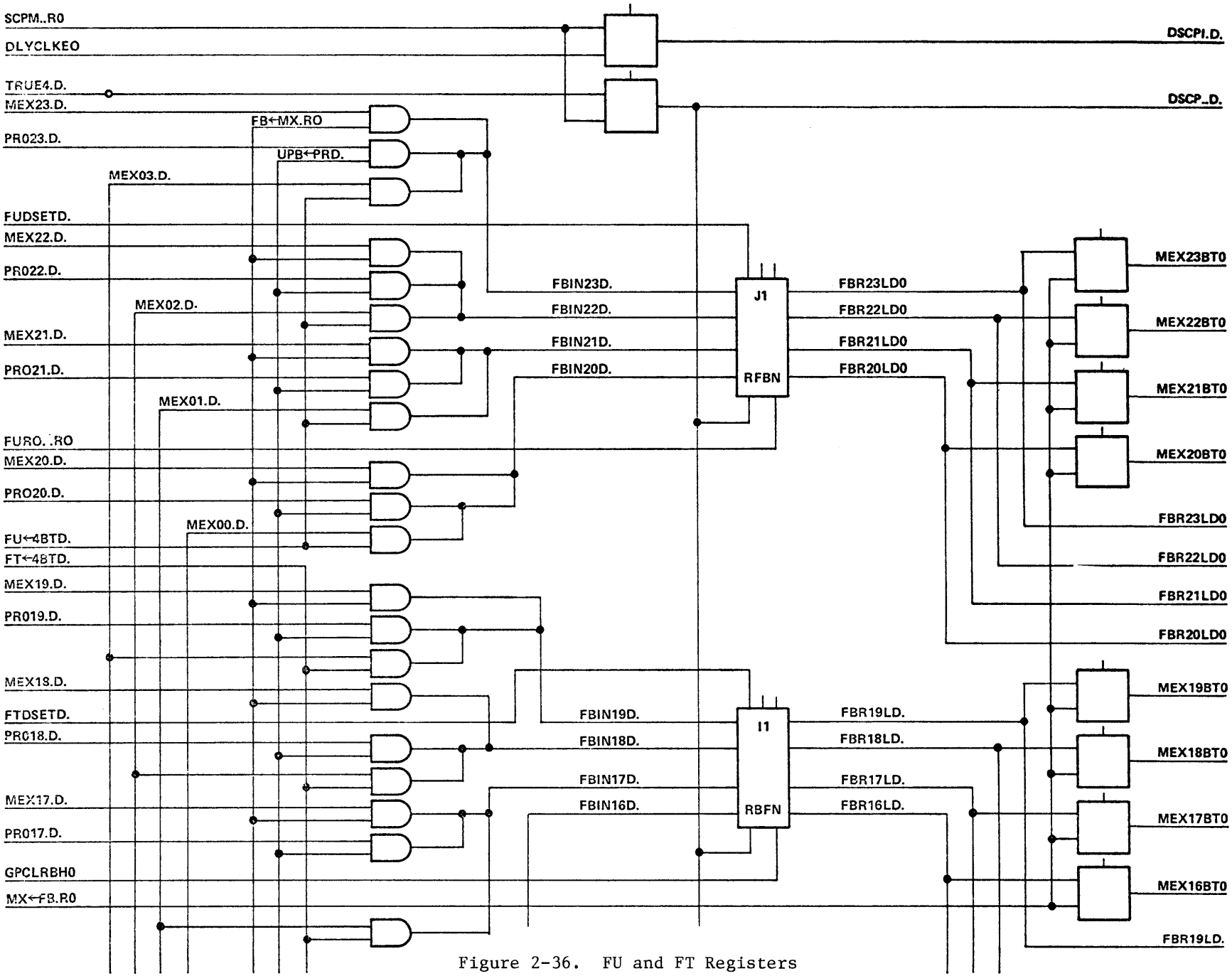


Figure 2-36. FU and FT Registers

Increment FL

Incrementing FL involves adding the contents of an external 5-bit input (SPCPLOTO through SPCPL4TO) to the existing contents of the 5 least-significant bits of FL, then returning the sum to the register. Refer to figure 2-37. To receive the sum, the circuit elements of FLF plus the FLE flip-flop operate in the D-set mode, with the remainder of the register acting as a counter to handle carries. Such carries are generated by the external carry logic associated with the count FL adder/subtractor. As with FA, the increment operation is controlled by two levels: CNTFL.TO (count FL) which is true for either an increment or decrement, and FL+-MOTO (FL add/subtract mode) which differentiates between the two. FL+ - MOTO is false for an increment.

The FL adder/subtractor is active at all times, continuously monitoring the contents of FL's five least significant bits and the SPCPL inputs. CNTFL.TO enables the input gates to FLF, in addition to the FLE flip-flop for the adder/subtractor's outputs (FL+-OOD. through FL+ - 05D.). At the same time CNTFL.TO places the balance of the FL register (3 most significant bits of FLE plus FLD and FLC) in the add mode to receive possible carries. Note that the FLC and FLD flip-flops are in the J-K mode when adding or subtracting. FL+-MOTO being false causes the AFAN adder/subtractor chips to operate in the add mode, and generates CNTFLUD. (count FL UP) which enables the output gates of the carry logic. This allows FLCI08D. through FLCI13D. to the FL carry inputs when the lowest-order carry input (FLCI05D) is true.

Decrement FL

Decrementing FL is done in the same manner as incrementing, except that both the adder/subtractor and FLC-FLD-FLE (less flip-flop) portions of FL operate in the subtract mode, and the borrow outputs of the carry logic are enabled. The mode change is caused by FL+-MOTO being true. This changes the AFAN chip mode directly, and generates the levels CNTFLDD. (count FL down) and FLDOWND. The former level enables the carry logic borrow outputs when FLCI05D. is true, and the latter changes the register chip mode. Note that CNTFL.TO must also be true for decrementing.

Clearing and Output Gating of FL

Clearing of FL occurs under several conditions. FL←O..D (reset FL) serves to do this by enabling both the unconditional clear input of the RFAN chips and the reset input of the flip-flops. FL←O.DD. is generated either by the external clear signal CLRFL.RO from micro decoding or internally, when the register is reduced to a value of one (while counting down). The latter condition forces a reset, preventing the register from wrapping around. This decrement reset occurs when levels 2FLOO.D. (most-significant 3 bits = 0), FLPDWND. (FL propagate down, bits 05 through 12 = 0), and CNTFLDD. are true, simultaneously. FL←01.D., when true, also disables CNTFLFD., removing the register chips from the subtract mode.

The FU and FT portions of FB are cleared by the FURO..RO and CPCLRBHO levels, respectively.

Gating the contents of FL to the MEX occurs when MX←FL.RO is true. FU and FT are gated to the MEX when MX←FB.RO is true, in which case MX←FL.RO is also true. The contents of all 24 bits of FB are continuously available to the right scratchpad input, where they are controlled by input gating.

The contents of the FU, FT, FLC, FLD, FLE, and FLF portions of FB are gated to the auxiliary 4-bit exchange via decoding logic as shown in figure 2-38. The decoders are enabled when 4 ClL8NO is true, and act as determined by the 4-bit source select code from MOP lines 08, 09, and 10.

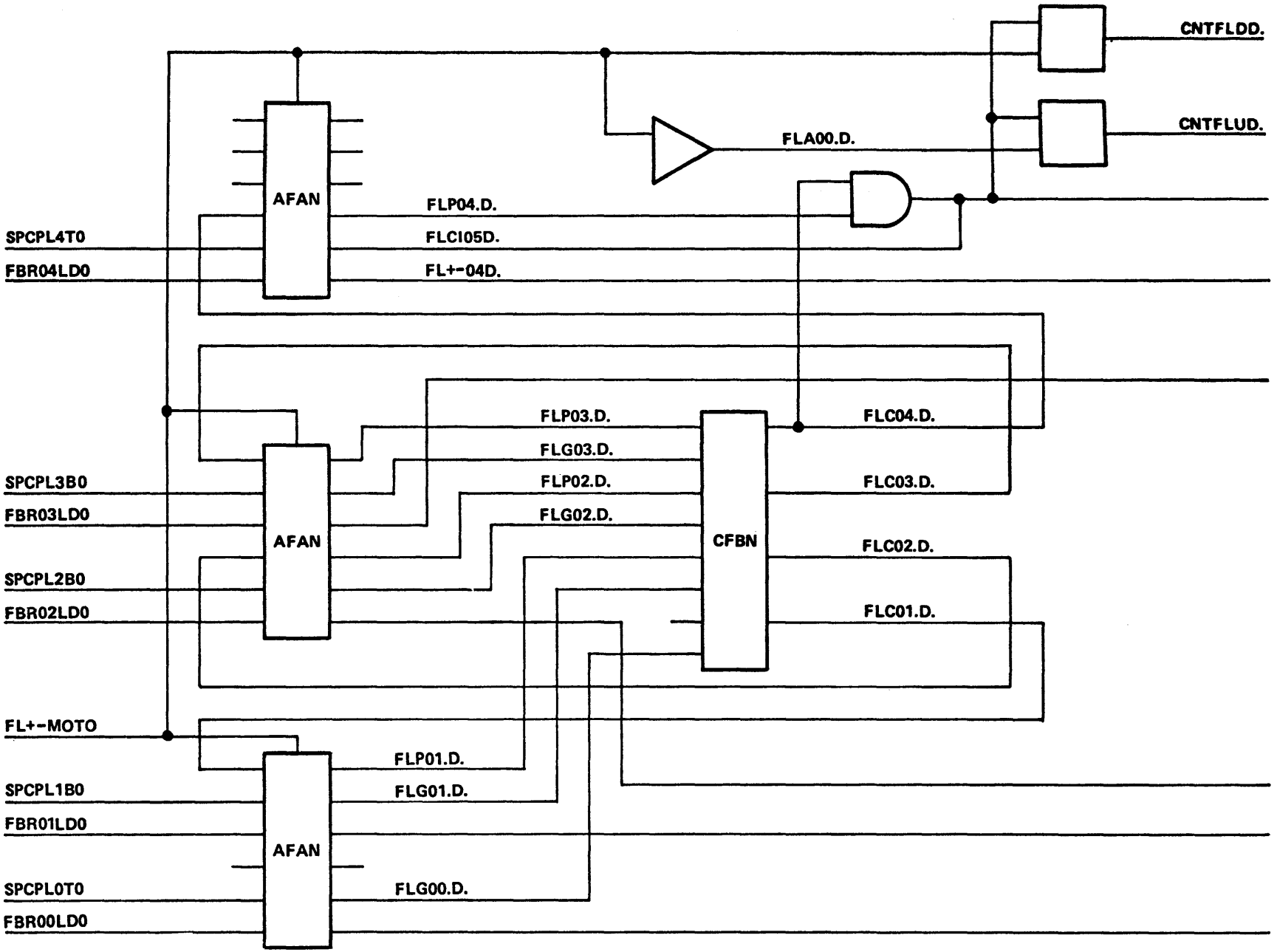


Figure 2-37. FL Adder-Subtractor

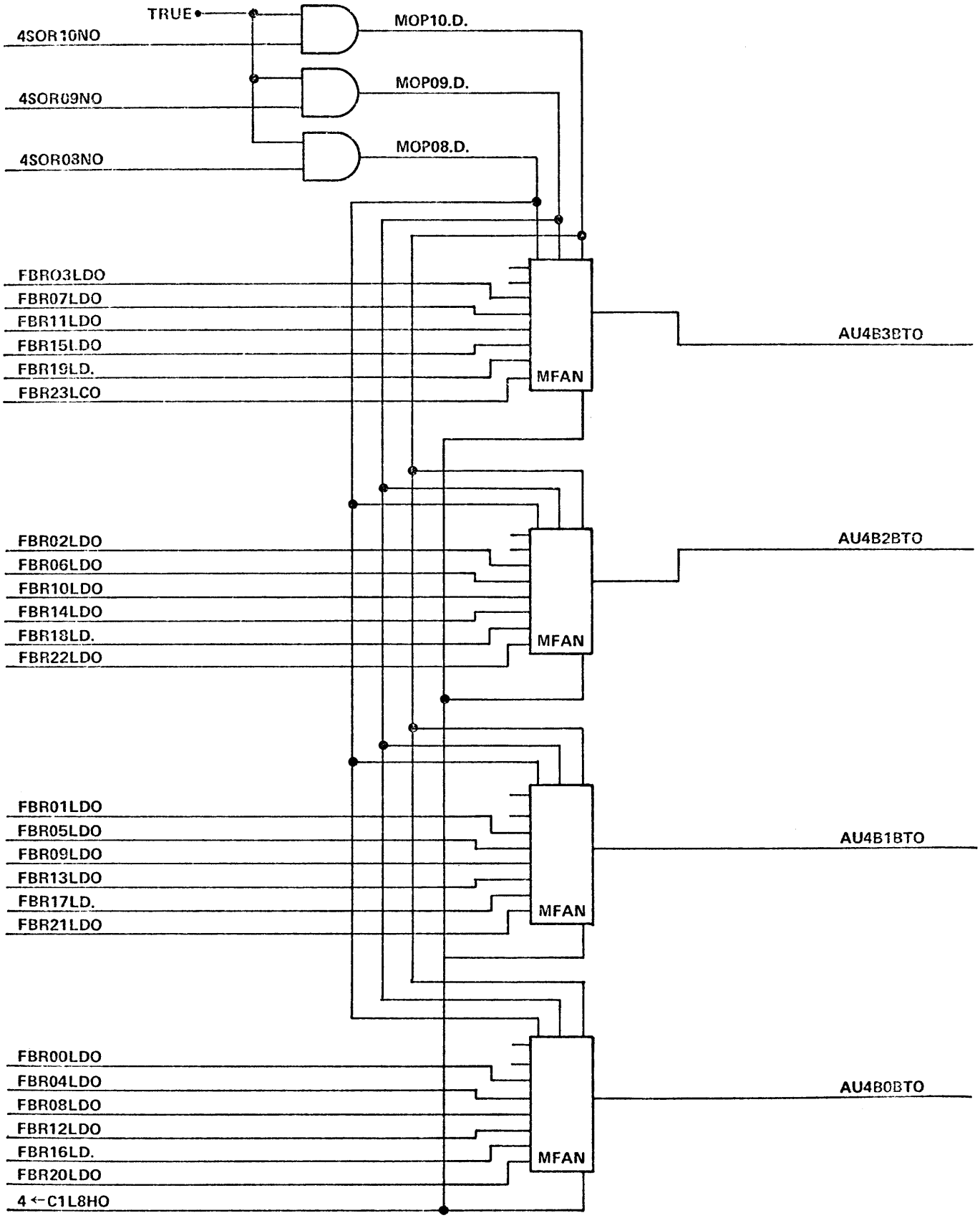


Figure 2-38. FB 4-Bit Output Decoding

C-REGISTER

The C-register is divided into four sub-registers (CA, CB, CC, and CD), the first two of which are available for general data storage. The latter two sub-registers are used exclusively to store interrupts and certain other conditions significant to processor operations. C is not addressable as a whole, but rather by its four-bit sections only. Any of these may be loaded from the four least-significant MEX lines or the 4-bit result bus. However, only the CA and CB portions may serve as a source, in which case they are gated to the auxiliary 4-bit bus. Certain of the bit positions of CC and CD may be loaded individually (in response to processor hardware functions), and the entire contents of both registers serve as individual control condition sources. Those CC and CD bit positions which represent interrupt conditions are monitored continuously by a circuit which serves as a general interrupt condition indicator.

C-REGISTER FUNCTIONAL DETAIL

The CA and CB portions of the C-register are each composed of a 3-bit RFAN register chip and one-half of an FFAN dual flip-flop. (See figure 2-39.) These constitute conventional 4-bit registers, with the 4-bit source data on T*L.20A. through T*L.23A being clocked into CA when CA←4X.A. is true and T*L.16A. through T*L.22A. clocked into CB when CB←4X.A. is true. Source selection for the input lines is external, and these same lines are also used to load CC and CD (in the same manner). CA←4X.A., being true, puts both the RFAN and upper-half of the FFAN in the D-set mode when CA is a sink. CB operates likewise. No other modes (except idle) are used in the RFANs, although the FFANs revert to the J-K mode when the sink select signal is false. Only the reset inputs of the flip-flops are utilized, and this for clearing purposes only.

CC and CD are made of FFANs only, and these are operated in the D-set or J-K modes, depending on what is being done. Loading from the MEX or 4-bit result bus (via the T*Lnn.A. lines) occurs when the sink select signal CC←4X.A. (CC register) or CD←4X.A. (CD register) is true. Both signals place the chips they control in the D-set mode, with the direct input being used for data input. At all other times the CC and CD flip-flops operate in the J-K mode, and are set by the hardware generated input signals as follows:

Bit	Signal	Significance
CC3	NONE	State Lamp - Set by Software Only
CC2	BS.CC2HO	Timer Interrupt *
CC1	SRVRQ.QO	I/O Bus Interrupt*
CCO	C/S.I.T1	Console Interrupt*
CD3	BS.CD3HO	Memory Read Parity Error*
CD2	NONE	Address Out of Bounds Override; Set by Software
CD1	BS.CD1HO	Address Out of Bounds (Read)
CDO	BS.CDOHO	Address Out of Bounds (Write/Swap)*

*Indicates five of the seven interrupt conditions monitored for an indication of a general interrupt condition.

Monitoring for a general interrupt condition is accomplished by an EFAN encoder chip, the output of which is known as 5/7INTAO. This signal is ORed with two additional interrupt conditions on card H to produce 7INTORH1 (bit 2 of the XYST (X-Y state) pseudo register).

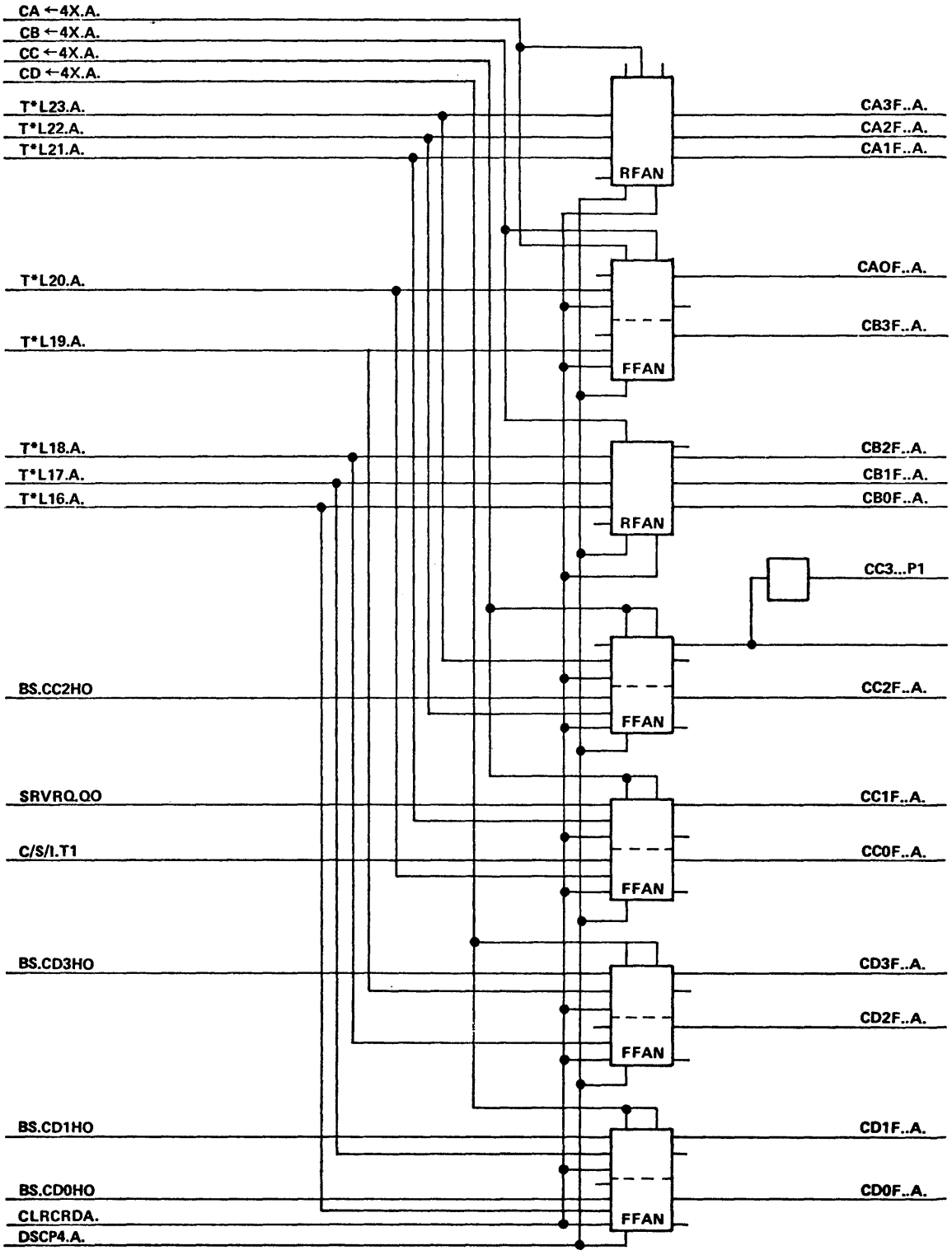


Figure 2-39. C Register

BR AND LR REGISTERS

The BR and LR registers are discussed together because they are identical, and share the same input and output gating. BR and LR are ordinary 24-bit hardware registers which are generally used programmatically to store reference memory addresses. Both registers are addressable as a whole only, and may be sourced or sinked by way of the MEX.

BR AND LR REGISTERS FUNCTIONAL DETAIL

BR and LR are physically divided into 4-bit sections, each of which is composed of one RFAN 3-bit register chip and one-half of an FFAN dual flip-flop (see figure 2-40). Loading the registers occurs when BR←MXLQ. (BR register) or LR←MXLQ. (LR register) is true, respectively. In each case, this enabling signal places both the RFAN and FFAN chips in the D-set mode, allowing the data present on the MEX to be loaded at the trailing edge of the next clock pulse. The RFAN and FFAN chips revert to the idle and J-K modes, respectively, when the enabling signal is false. Only the reset input of each flip-flop is used in the J-K mode, and this for clearing purposes.

Output gating of BR and LR to the MEX is combined with that of the I/O bus, and consists of AND gates (see figure 2-41). The contents of BR are gated to the MEX when MX←BR.Q. and ALLOW.Q., being true, gates the contents of BR to the MEX.

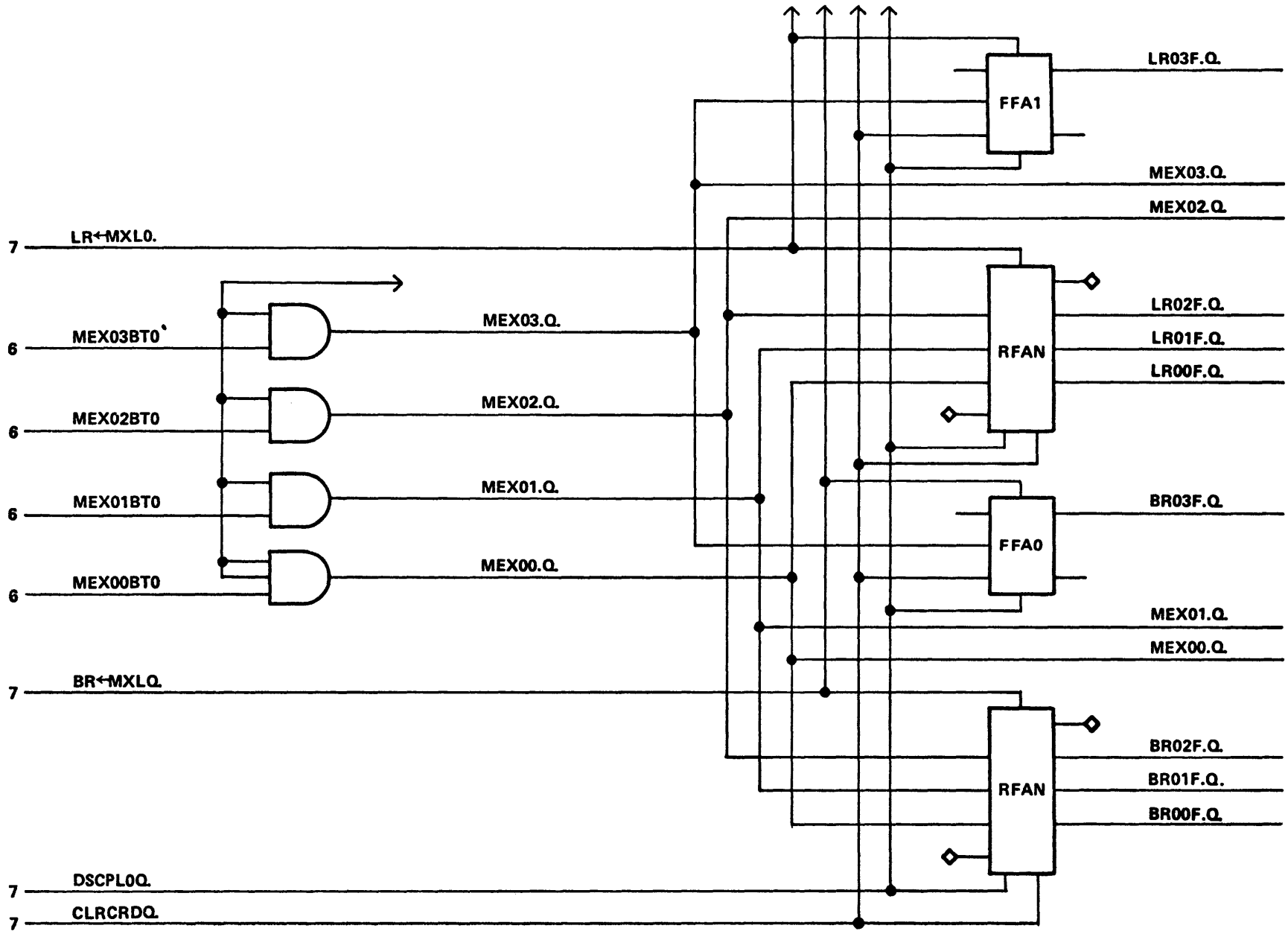


Figure 2-40. BR and LR Registers Bits 00-03

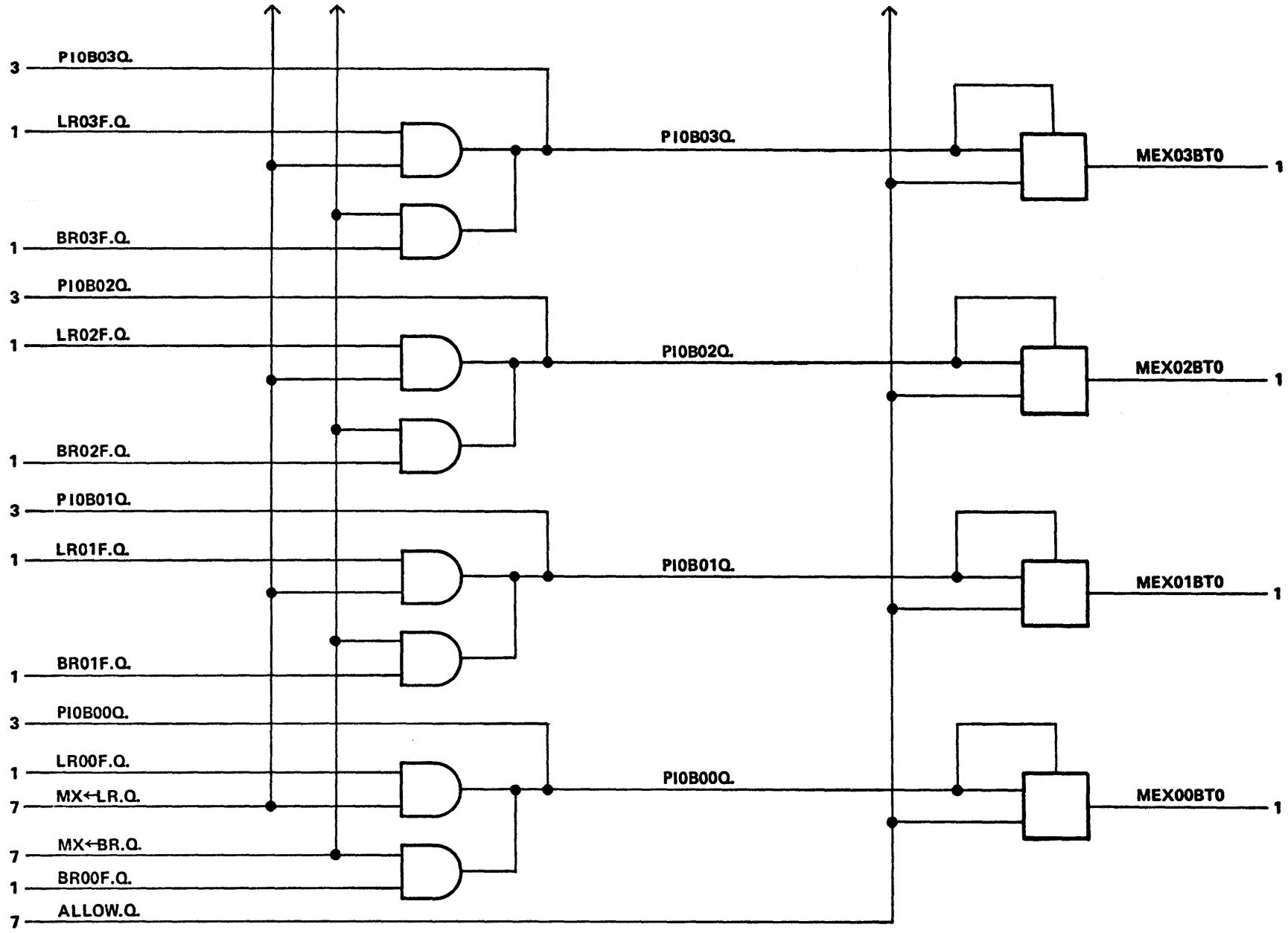


Figure 2-41. BR and LR Registers and I/O Bus to MEX

SCRATCHPAD

The scratchpad is a general purpose, in-processor storage medium for 24-bit data fields. Scratchpad is divided into two halves: left and right. Each half contains sixteen 24-bit storage locations. Left and right scratchpad may be used individually as 16 x 24-bit registers, or together as a 16 x 48-bit register. Note that storing 48-bit data fields in one or more word addresses of left and right scratchpad does not prevent use of the remaining words of each scratchpad for individual 24-bit registers.

SCRATCHPAD FUNCTIONAL DETAIL

Scratchpad is composed of 4 x 16-bit RFCN memory chips, and, therefore, is divided into 4-bit segments. Each segment consists of input gating, a 4-bit latch, a memory chip, and inverted output gating. (Refer to figure 2-42.)

The input and output gating provides data paths respectively from/to the MEX and FA register. Input latches are provided because incoming data appears too late to be written into the memory chips within one clock period of the controlling micro instruction. Since there are minor differences, left and right scratchpad are discussed separately.

Left Scratchpad

During a write to left scratchpad, data is gated from the MEX or FA, depending on which of the signals PLF←MXRO and PLF←FA.RO is decoded from the MOP lines. This input data is stored temporarily in the LFAN latches which operate in the D-set mode during write operations. The operating mode of the latches is controlled by mode lines PLLM0.RO and PLLM1.RO, both of which are true to select D-set. When not performing a write operation, both mode lines are false, causing the latches to be idle. The latches are cleared when PLLM1.RO (only) is true. This occurs when GPCLR.R. (general processor clear) is true.

The scratchpad address for read and write operations comes from the scratchpad address control logic, entering via the SPA0..RO through SPA3..RO lines. Writing occurs during the first half of the clock period following acceptance of the write data in the latches (taking place when the negative going write enable pulse PLWE/.RO appears). Note that the processor is already executing the next micro instruction at this time.

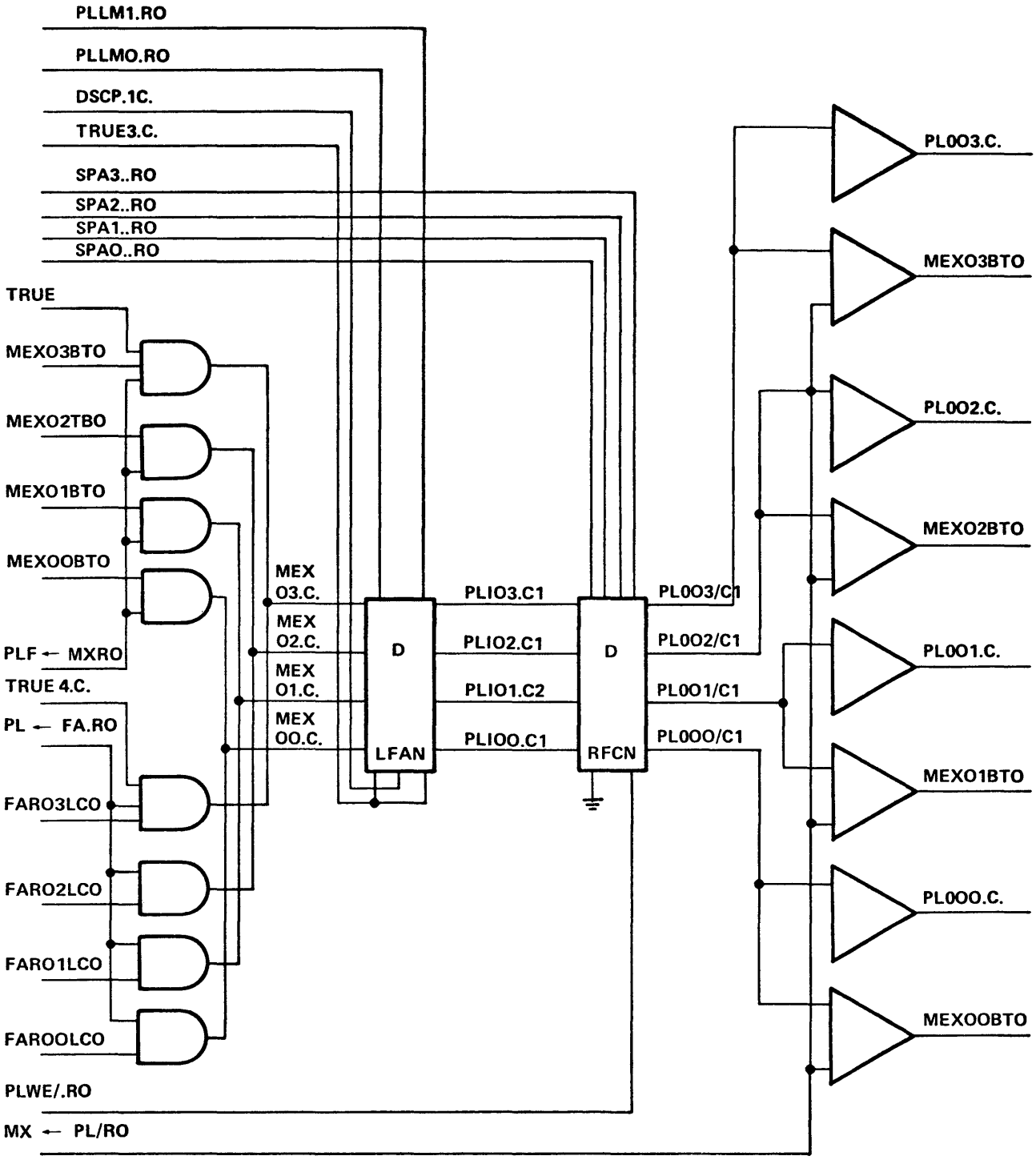


Figure 2-42. The Left Scratchpad Bits 00 Through 03

Reading from scratchpad is accomplished by supplying the desired address to the RFCN chips via address lines (SPA0..R0 through SPA3..R0), and gating the output to a sink. Read data output from the RFCN chips is continuously available, except when PLWE/.R0 is false. Gating to sinks occurs when MC←PL/R0 is false, with sink selection being controlled by input gating at the destination. All outputs of the RFCNs are inverted because they are TTL logic elements.

Right Scratchpad

Right scratchpad is basically the same as the left, with the exception that its address zero is reserved for storage of field unit and field length information (SFU and SFL).

Data can be written into right scratchpad from either the MEX or FB register, with the write taking place when PRF←MXR0 (source = MEX) or PRF←FB.R0 (source = FB) is true, along with PRWE/.R0. Gating data out to an external sink is similar to left scratchpad, occurring when MX←PR/0 is false. To meet system timing requirements, right scratchpad logic incorporates a discrete 20-bit hardware register which receives, along with the memory chips, input data destined for address zero (only). This register constitutes the SFL and SFU pseudo registers, from which store data is available to the bias logic. Entry of input data to SFL and SFU occurs at the same time as in the scratchpad input latches, and is therefore available to the bias logic while still being written in the memory chips themselves. A portion of the SFL-SFU logic is shown in figure 2-43.

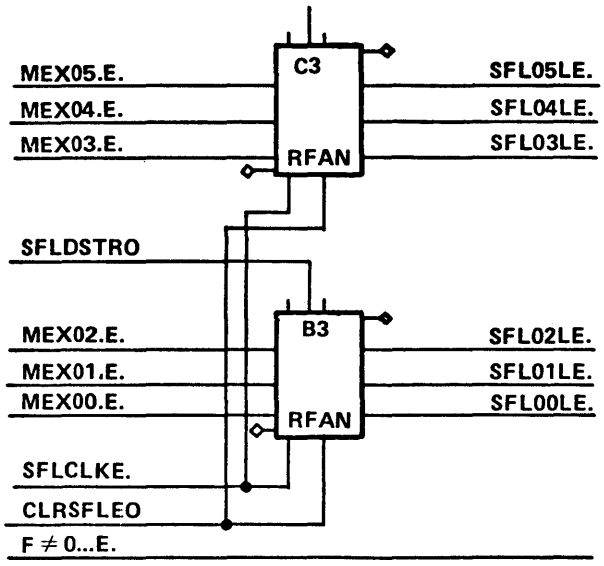


Figure 2-43. SFL Registers and FLCN Logic

Scratchpad Address Control

Operational control of the accessing functions for scratchpad comes from the associated scratchpad address control, which is located on processor card R. This circuitry provides overall timing of the write function, plus scratchpad address source selection during both read and write operations. For the write function only, facilities are provided for temporary storage of the incoming address.

Although all scratchpad addresses come directly from the MOP lines, those lines actually used depend on the micro operator being executed. Therefore, input gating has been provided to select between MOP lines 00 through 03 and 04 through 07 (see figure 2-44). This configuration was included to satisfy the requirements of the 7D micro, which involves both read and write operations in scratchpads. Since independent read and write address selection is provided, and the two operations must occur at different times, provisions for temporary storage of one address are also necessary. This requirement is met by a register in the write address data path. Reading is done first when performing a 7D micro to avoid loss of data when the read and write addresses are the same.

Operationally, read and write addresses always enter the circuit over the separate paths provided, regardless of the micro being executed. Also, all write addresses pass through, and are temporarily stored in the register, even though the storage requirement does not exist for micros other than 7D. On a typical scratchpad read operation, the source doublepad word address enters via MOP lines 00 through 03, and are passed to the scratchpad address lines (SPA0..R0 through SPA3..R0) by way of buffers. Refer to figure 2-44. These buffers are enabled by ADCNTLR., which is true at all times except during execution of a scratchpad write. The timing circuits do not affect read operations.

Performing a write operation requires source MOP line selection, and this is determined by micro decoding. For 2C and 4E micros, the address appears on the same MOP lines as for a read, but the isolated MOP outputs MP00..R. through MP03..R. are used for input to the write address data path. Such input occurs when 2C+4E.R. is true, with the 7C input of MP04..R. through MP07..R. occurring when 0C7D..R. is true. To allow read/write address selection by a single control signal (ADCNTLR.), the write address is inverted prior to storage in the RFBN register. Gating into the register occurs at the trailing edge of ADCLK.R. (address clock) when PLRWR.R. is true. This latter signal places the RFBN in the D-set mode. Once stored in the register, the inverted address is available for gating to the scratchpad address lines. Gating occurs when ADCNTLR goes false, with re-inversion of the address taking place in the NOR gate inverters.

Scratchpad Write Timing

The scratchpad write timing logic serves to control the application of the PRWE/.R0, PLWE/.R0, PLRWR.R., and ADCNTLR. signals, manipulation of which controls all the necessary functions for accomplishing a write operation. The timing logic consists of a series of delay lines and gating elements, and derives all scratchpad write control signals from the early clock (SCPS....).

A scratchpad write operation is initiated whenever PLWR..R. (left scratchpad write), PRWR..R. (right scratchpad write), or both are received from the micro decoding logic. Such command signals, when true, produce the following sequence of actions (refer to figure 2-45). For the purposes of discussion, it is assumed that a left scratchpad write is occurring.

a. PLWR..R. goes true, generating PLWRSTR. (pad left write start) and PLRWR.R. (pad left/right write). The former signal removes one of the OR conditions holding PLWE/.R0 true, allowing the internal timing to assume control. A similar function occurs as a result of PLRWR.R. going true; this being the generation of WRSTORR. which removes one OR condition holding ADCNTLR. true. PLRWR.R. also puts the

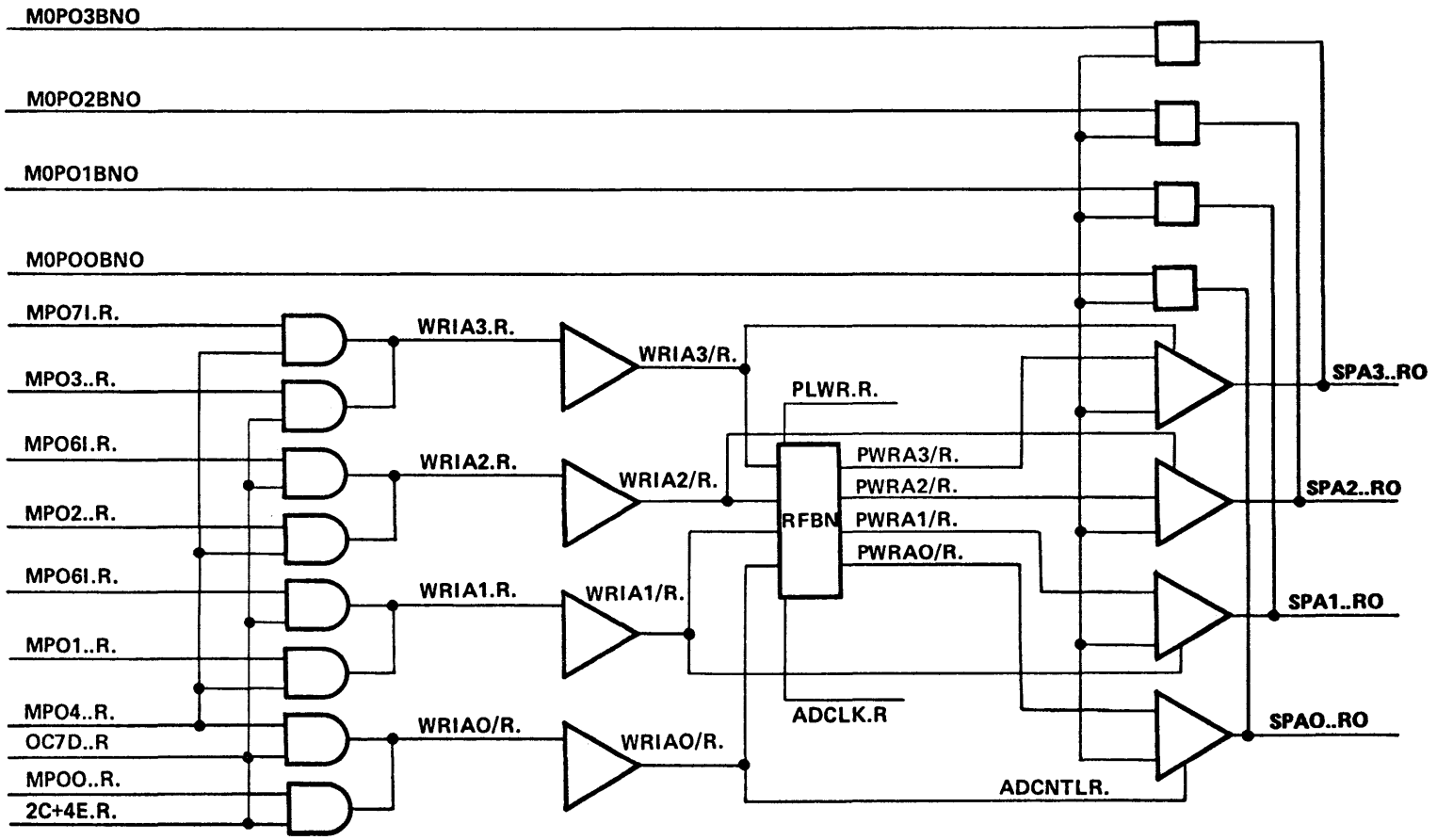


Figure 2-44. Scratchpad Address Control Logic

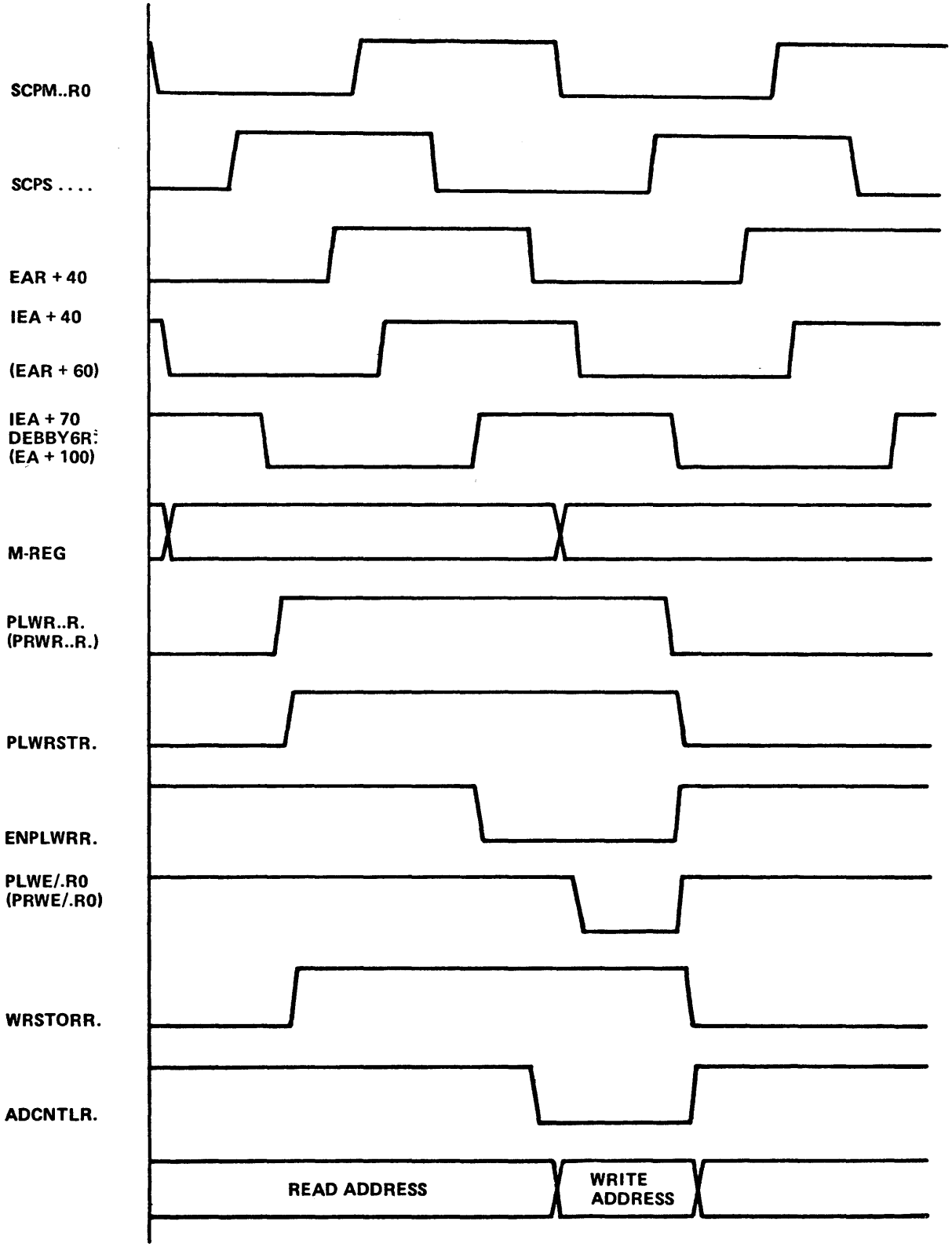


Figure 2-45. Scratchpad Write Enable Timing Chart

scratchpad address register in the D-set mode, preparing it to accept the incoming address. Note that the term PLRWR.R. is also used for a right scratchpad write.

b. At the leading edge of the next SCPS.... pulse, ADCLK.R. is generated, clocking the scratchpad address into the register. Note that the term ADCLK.R. is also used for a right scratchpad write.

c. At Early Clock +100 nanoseconds time, DEBBY6R. goes true, removing the second OR condition holding ADCNTLR true. Likewise, IEA+70R. goes true at the same time, removing the second OR condition holding PLWE/.RO true. Note that the terms DEBBY6R. and IEA+70R. are also used for a right scratchpad write.

d. Twenty nanoseconds following the above action, EAR+40R. goes false, removing the last OR affecting ADCNTLR., and it goes false, enabling gating of the write address to scratchpad via the SPA0..RO through SPA3..RO lines. Note that the term EAR+40R. is also used for a right scratchpad write.

e. Twenty nanoseconds later, IEA+40R. goes false, removing the last OR affecting PLWE/.RO, and it goes true, enabling scratchpad. The write takes place at this time. Note that the term IEA+40. is also used for a right scratchpad write.

f. The write is terminated 40 nanoseconds after PLWE/.RO goes true, this resulting from a simultaneous change in state of PLWR..R.* and IEA+70R. Both of these signals go false, forcing PLWE/.RO and ADCNTLR. true again. Note that the terms PLWR..R. and IEA+70R. are also used for right scratchpad write.

L-REGISTER

The L-register is, with the exception of several details concerning input and output gating, identical to the T-register. Therefore, L is not discussed in detail, but rather only its distinguishing features are pointed out. Refer to card A, sheet 3, of the logic schematics if required.

The L-register shares input and output gating with the T-register, with both source and sink selection external to the register logic itself. Unlike T, the least-significant seven bits of L are not addressable separately as either source or sink. Since inclusion of this feature in T involves the addition of certain extra gating elements, such provisions were simply omitted in the L-register. L remains addressable as a whole, and as individual 4-bit sub registers. Significant control signals affecting L are as follows:

Term	Control Signal
L←MX..TO	Move main exchange to L
LA←4X.A.	Move 4-bit auxiliary exchange to LA
LB←4X.A.	Move 4-bit auxiliary exchange to LB
LC←4X.A.	Move 4-bit auxiliary exchange to LC
LD←4X.A.	Move 4-bit auxiliary exchange to LD
LE←4X.A.	Move 4-bit auxiliary exchange to LE
LF←4X.A.	Move 4-bit auxiliary exchange to LF
L←0...RO	Reset L: generates CLEARLA., as does CLRCRDA.
MX←L..A	Move L to main exchange

A-STACK

A-stack is a 24-bit wide x 32-word deep memory which operates as a push down stack with a "last in/first out" structure. To accomplish this function the memory is addressed by a counter known as the stack pointer. The stack pointer is automatically incremented/decremented by data moves into or out of the stack and will wrap around in either direction. The location within the stack which is referenced by the pointer is known as the top-of-a-stack (TAS), and is the only portion of the circuit accessible by the processor hardware as a source or sink. A-stack is normally used for storage of addresses significant to program structures, such as exit points for various subroutines. Although external control of the A-stack address (referenced by the pointer) is not provided, any desired location may be accessed by repeated TAS-to-NULL moves, which serve to decrement the pointer without destroying the data stored within the stack.

A-STACK FUNCTIONAL DETAIL

A-stack is composed of RFCN memory chips, with input latches and inverted output gating. Each RFCN chip is capable of storing 16 4-bit words, requiring an array of 12 chips to make up the 24 x 32 configuration of A-stack. The 12 chips are arranged in two rows, with each 4-bit input and output data path being shared by two chips. The two rows themselves are enabled sequentially as the stack pointer proceeds through its count. Input and output for the stack is from/to the MEX only. One 4-bit section of A-stack is shown in figure 2-46.

Sourcing A-stack involves simply gating the data appearing at its output to a selected sink by way of the MEX, and occurs when LTASE1E. is false and MX←TASTO is true. The data actually sourced when this is done is that contained in the location within A-stack which is currently designated TAS by the stack pointer. The binary count from the stack pointer is used as the address, and enters via address lines ASA00.E1 through ASA03.E1 and ASA10.E1 through ASA13.E1. Each of the two groups of address lines feeds the corresponding row of memory chips, with chip selection being determined by the ASA4..E1 and ASA4/.E1 address lines. Those chips receiving a false chip select level are active.

Sinking A-stack involves gating the incoming data into the input latches, incrementing the pointer, then writing the data into the addressed location. This occurs when TAS←MXTO and IGNOR/E. are both true. From these signals, STAS←XE. (which enables the input gates from the MEX), and ASM0..E. and ASM1..E. (which place the input latches in the D-set mode) are generated. From the latches, the data is written into the RFCN chips when the write enable signals (1ASWE/E1 and 2ASWE/E1) go false. Note that only the chip row which also receives the chip enable signal (1ASWE/E1 for the first row, and 2ASWE/E1 for the second) is affected.

A-Stack Pointer

The A-stack pointer (figure 2-47) consists of two parallel-counter sections which produce the A-stack addressing signals.

Each counter section feeds one of the two chip rows constituting the A-stack memory, with the count output from both being identical at all times. The counters are configured for both addition and subtraction, with wrap around permitted in both directions. Associated with the counters is a chip row select flip-flop which enables the two rows of memory chips in sequential order. This causes the stack to operate as a 32-word circulating memory. Upcounting by one occurs whenever a move into TAS is performed, with TAS←MXTO and IGNOR1E. generating CNTTASE. (Count TAS) and ASADD.E. (A-stack add mode). When the counter reaches a value of seven, the level AS=07.E. comes true, allowing ASADD.E. to generate TOG1..E on the next write operation. This complements the flip-flops at B2, producing ASA03.E1 and ASA13.E1. These signals provide address bit 8 (binary weight 8) for the two chip rows.

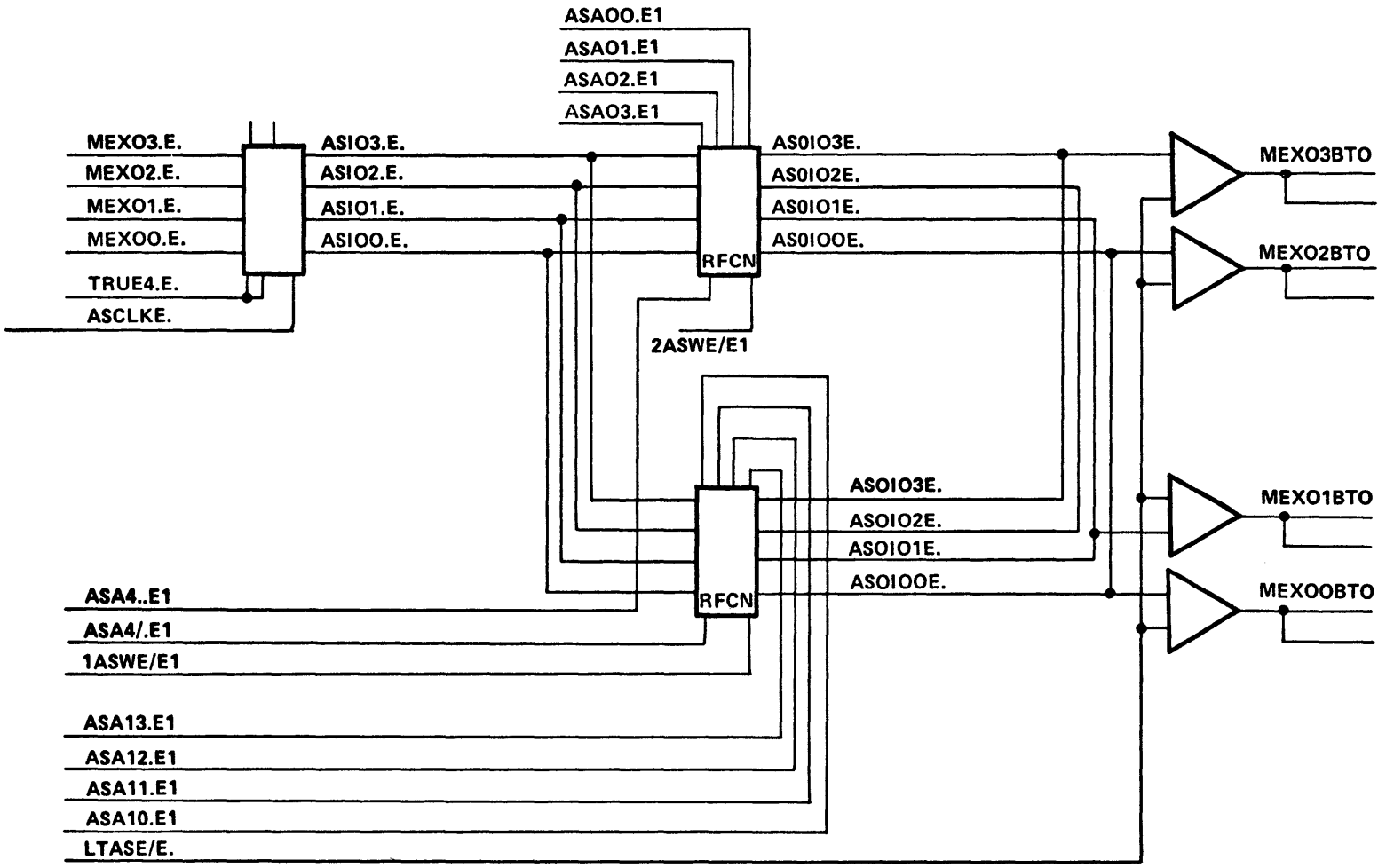


Figure 2-46. A-Stack Bits 00-03 with Input Latches

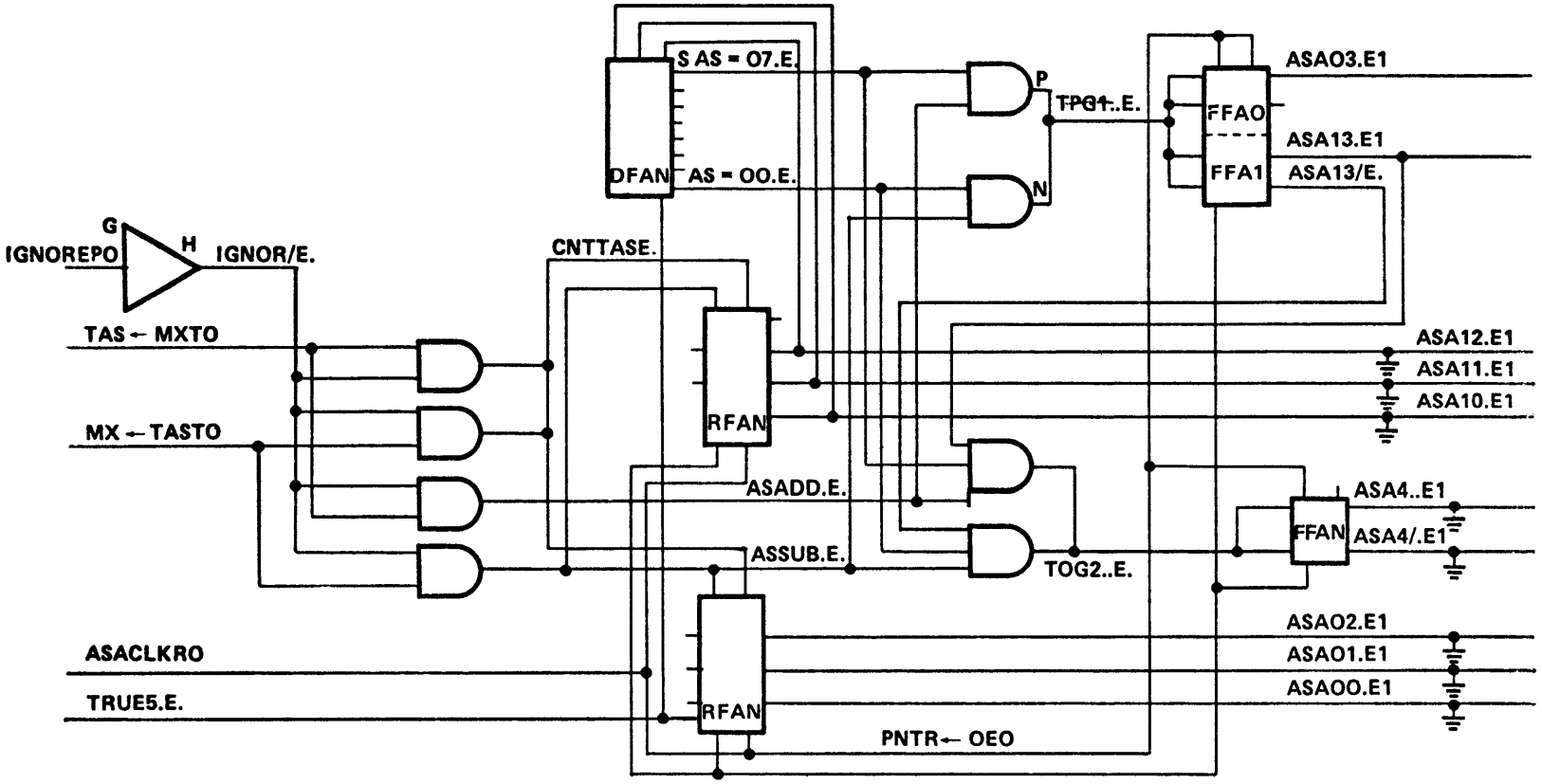


Figure 2-47. A-Stack Pointer

When the counter address reaches 15, AS=07.E comes true again, and this, along with ASA13.E1, enables generation of both TOG1..E. and TOG2..E. on the next write operation. This combination of signals causes both the B2 and C2 flip-flops to complement as the RFAN chips wrap around to zero. The complementing of C2 causes ASA4.E1 and ASA4/.E1 to change states, enabling the second chip row and disabling the first. B2 complementing resets address bits ASA03.E1 and ASA13.E1 which, with the zero output from the RFANs, causes address zero of the second chip row to be selected.

During write operations, TOG1..E. occurs when the address is 7, 15, 23, or 31, and TOG2..E. occurs when the address is 15 or 31. During a read, the counters subtract, with the difference being that TOG1..E. occurs when the address is 00, 08, 16, or 24, and TOG2..E. when the address is 00 or 16.

The signal IGNOR/E. is used to prevent exposure of the A-stack logic to the various enabling signals for more than one clock of any given micro. Since the stack pointer and write enable controls are activated by internal clock signals, the presence of an enabling signal for more than one clock period would cause multiple writes.

A-Stack Write Enable Control

The A-stack write enable control (figure 2-48) consists of circuit elements for producing the enabling signals required for an A-stack write. These include placing the A-stack input latches in the D-set mode, clocking the incoming data into the latches, and causing this data to be written into the memory chips themselves. These three steps occur in a programmed sequence, with timing controlled by delayed clock signals derived internally from the system clock. The write sequence is initiated when TAS←MXTO and IGNOR/E. are both true, with the various output signals being produced as shown in figure 2-49. Note that upcounting the stack pointer is a separate function which takes place concurrently. The write control signals may be described as follows:

ASM0..E1 (A-stack mode 0): Produced when TAS←MXTO and IGNOR/E. are both true, and used in conjunction with ASM1..E. to place the A-stack input latches in the D-set mode.

ASM1..E. (A-stack mode 1): Produced when TAS←MXTO and IGNOR/E. are both true or GPCLRBTO. is true. Used with ASM0..E. as described above, and alone to clear the latches when a general processor clear is performed.

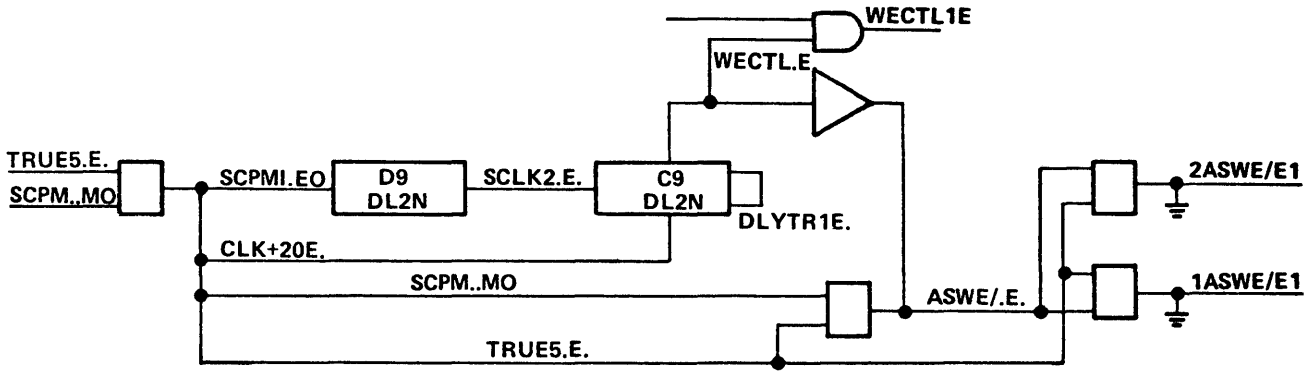


Figure 2-48. A-Stack Write Enable Control

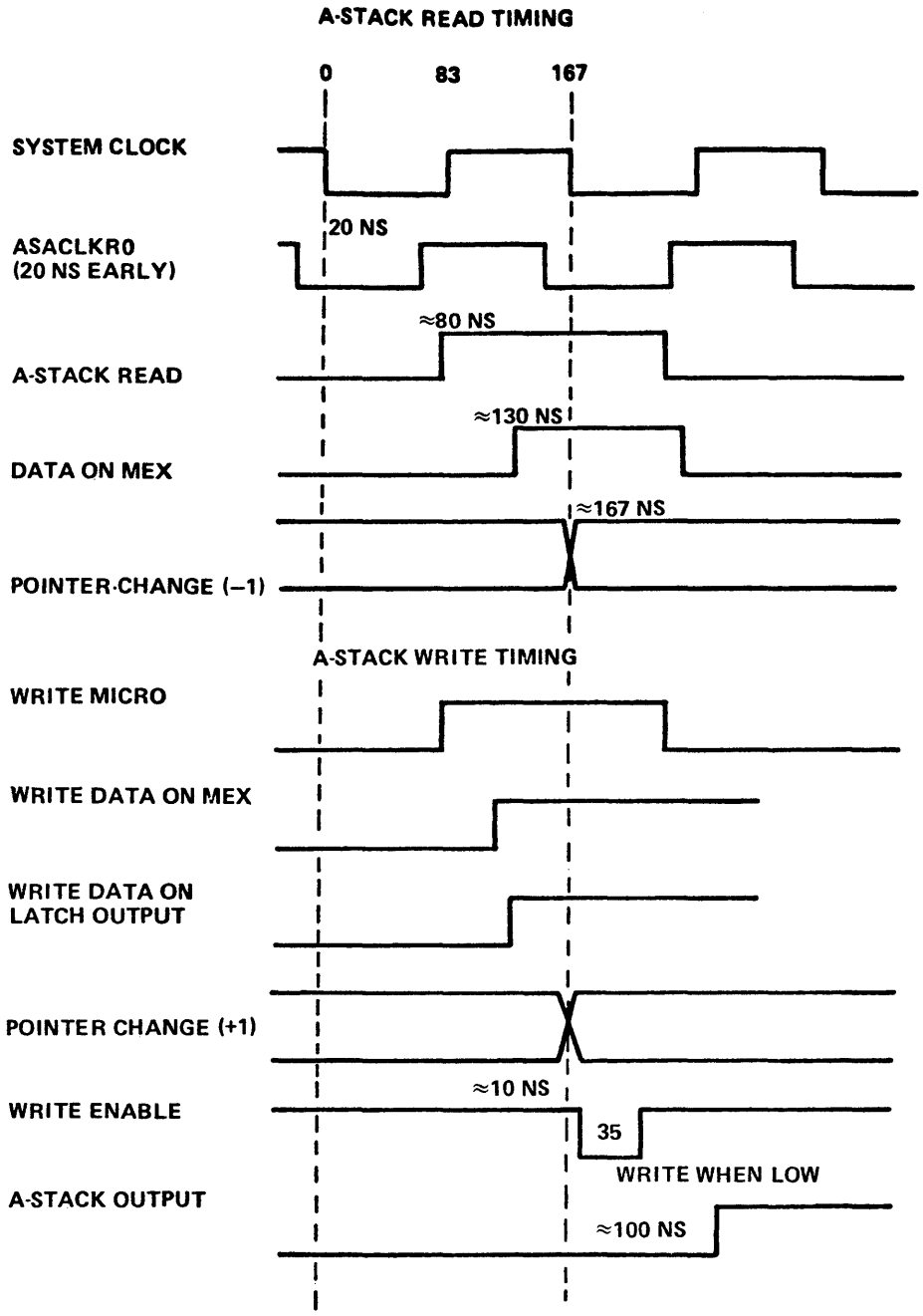


Figure 2-49. The A-Stack Read/Write Timing Chart

ASLCLKE (A-stack latch clock): A delayed clock derived from the logical ANDing of system clock (SCPM..HO) and system clock +20 nanoseconds (CLK+20E.).

1ASWE/E1 and 2ASWE/E1: A-stack write enable for first and second rows of RFCN memory chips. These are single, 35 nanosecond, negative-going pulses produced by logical ANDing of the systems clock (SCPM..HO) and delayed clock signal WECTL.E. (whenever this condition is exposed by CLRLCHE. being false). At the same time both clock signals are false. This latter condition occurs only when WR.CLKE. is forced true out of its normal sequence by TAS←MXTO and IGNOR/E.

MAXS AND MAXM

MAXS and MAXM consist of jumper chips which are hard-wired to produce a binarily-weighted bit pattern representing the physical size of installed S- and M-memory. These bits are gated to the main exchange (when sourced) via the 24-bit function box output logic. MAXS and MAXM, along with the various possible jumper configurations, are shown in figure 2-50. Note that, as with other binarily-weighted values, the increments shown in figure 2-50 are additive. For a MAXS size of 40K, jumpers F - J and R - S would be installed on chip 17.

24-BIT FUNCTION BOX

The 24-bit function box provides the common arithmetical and logical functions between two operands, including sum, difference, AND, OR, exclusive OR, plus complement and mask of a single operand. Input to the function box is always by way of the X- and Y-registers, with the result being returned to the MEX. In addition to the 24-bit result output, the function box also produces three 4-bit outputs which serve to indicate the status of a number of internal conditions. These 4-bit outputs are known as the XYST, XYCN, and BICN pseudo registers, and are used for programmatic decisiion making. To accomplish its required actions, the 24-bit function box is composed of the following sections (refer to figure 2-51).

a. The X- and Y-registers, which in addition to storing input data, are capable of shifting left or right, either separately or together, as a continuous 48-bit register.

b. Static compare logic which produces the pseudo register outputs.

c. A binary adder/subtractor which serves to produce arithmetic functions between the X/Y operands. Also, some of the logical functions are derived from control levels produced within the adder/subtractor.

d. BCD correction logic which converts the binary output of the adder/subtractor to binary coded decimal units when so specified.

e. A mask generator which restricts the number of bits gated to the output of the function box in accordance with the value specified by the CPL register.

f. Additional gating and logic elements for producing the complement function, and those logical functions not derived from the adder/subtractor.

g. An output multiplexor which selects the appropriate output of the function box for gating to the MEX.

MAXM

JUMPER L8	WEIGHT		CORR. MAIN EXCHANGE BIT
	WORDS OF M-MEMORY	BITS OF M-MEMORY	
E–K	4K	64K	12
F–J	2K	32K	11
C–H	1K	16K	10

MAXS

JUMPER	WEIGHT		CORR. MAIN EXCHANGE BIT
	BYTES OF S-MEMORY	BITS OF S-MEMORY	
A–P (L7)	1024K	8192K	23
B–N (L7)	512K	4096K	22
C–M (L7)	256K	2048K	21
D–L (L7)	128K	1024K	20
E–K (L7)	64K	512K	19
F–J (L7)	32K	256K	18
G–H (L7)	16K	128K	17
R–S (L7)	8K	64K	16
A–P (L8)	4K	32K	15

NOTES:

1) MAXM – DEFINES SIZE OF M-STRING-. MEMORY RESOLUTION. 1K WORDS (16 BIT-WORD) OR 16K BITS.

2) MAXS – DEFINES SIZE OF S-MEMORY RESOLUTION. 4K BYTES (8 BIT-BYTE) OR 32K BITS.

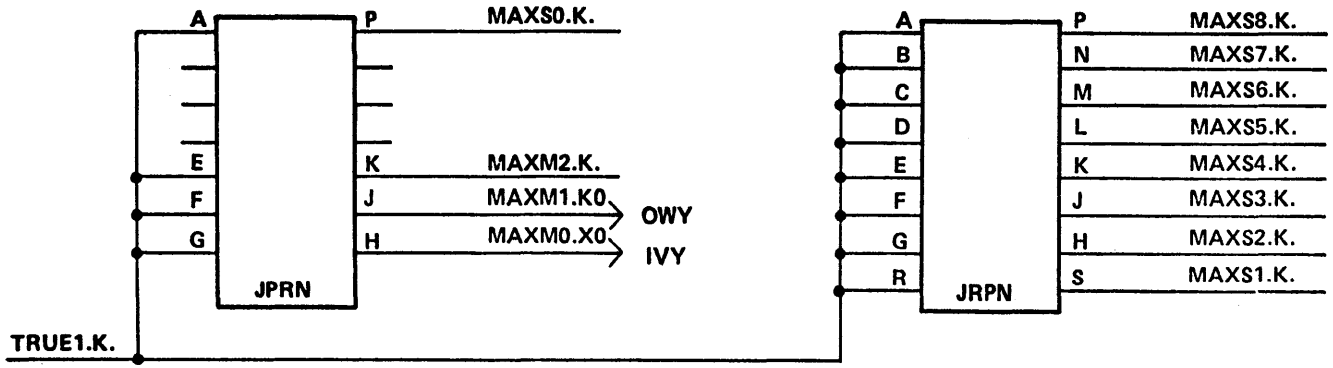


Figure 2-50. MAXS and MAXM

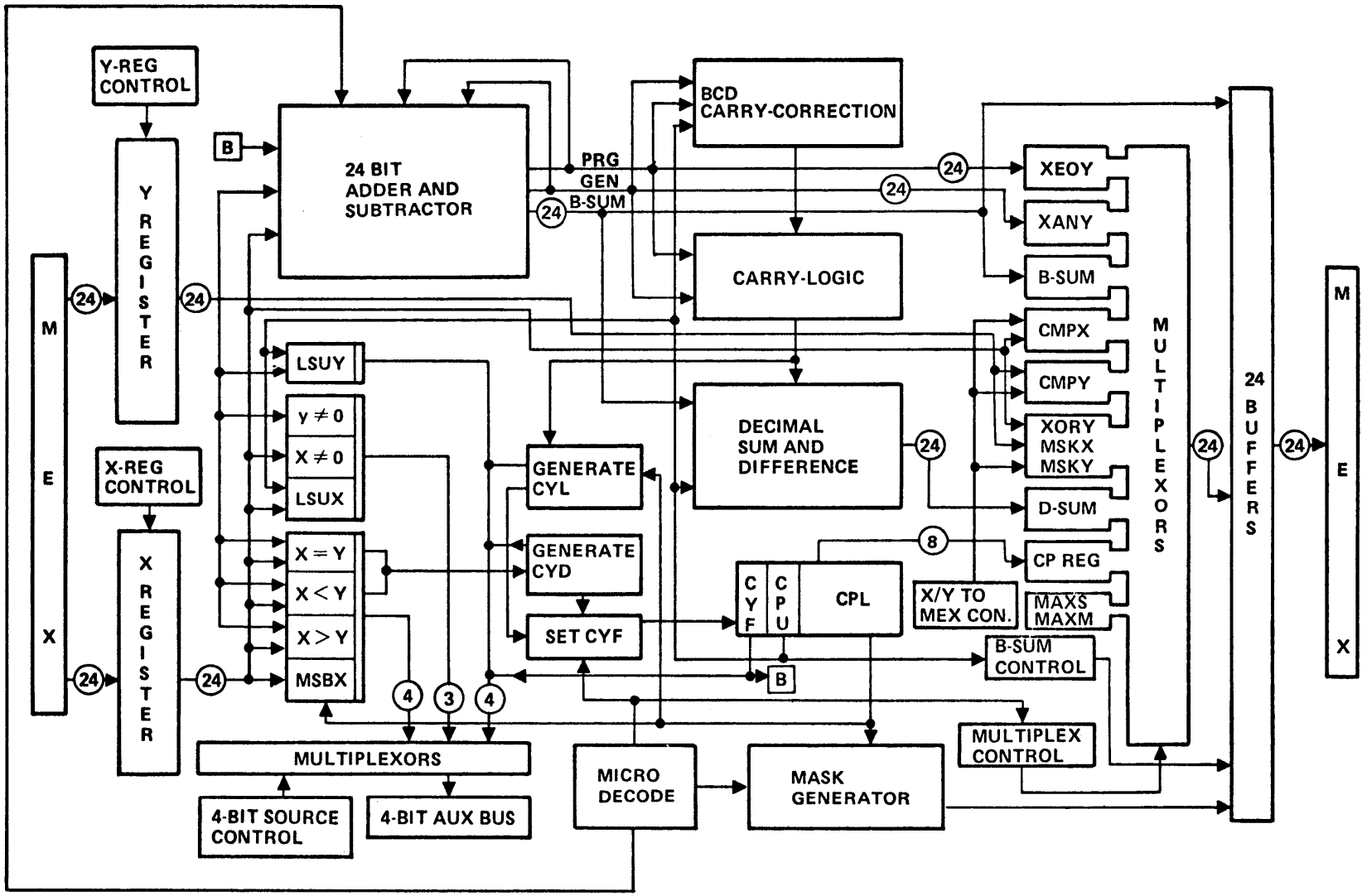


Figure 2-51. The 24-Bit Function-Box Block Diagram

X- and Y- REGISTERS

The X- and Y-registers serve as the two operand inputs to the 24-bit function box. Although this is their dedicated function, both may also be used separately as hardware registers without involving the function box as such. Both are addressable in their entirety only.

X- and Y-Register Functional Detail

The X- and Y-registers are identical, each being composed of 8 RFAN chips. In addition, both share common input lines from the main exchange. These input lines are isolated, but not gated, so the contents of the MEX are available to the registers on a continuous basis. One-half of the X-register is shown in figure 2-52. The outputs of X and Y feed the adder/subtractor AFAN chip inputs, plus the X/Y compare logic and function box output gating. When so designated by a micro operator, X and Y may be added, subtracted, or be subjected to the AND or Exclusive OR function in the adder/subtractor stage. The OR, complement, and mask functions are accomplished in the output gating, which can also move the unmodified contents of either register back to the MEX.

The operation of X and Y is controlled by the mode lines XCONMOLO through XCONM2LO and YCONMOLO through YCONM2LO, respectively. Of the eight possible modes, four are employed in this circuit; Idle, D-set, shift up, and shift down. D-set is used for loading, with shift up/down reserved for the rotate X/Y functions. The registers are idle at other times, but retain the data previously stored. The mode control lines of both registers are active as shown in table 2-2.

Table 2-2. X/Y Register Mode Control

Mode	XCONMOLO (YCONMOLO)	XCONM1LO (YCONM1LO)	XCONM2LO (YCONM2LO)
Idle	0	0	0
D-Set	0	1	0
Shift UP	0	1	1
Shift Down	1	1	1

Both registers can be cleared with GPCLR.R. true or when the clear register micro with MP04..R. (X-register) or MP05..R. (Y-register) true is executed. These functions come from processor card R, generating XCLEARRO and YCLEARRO. Shifting/rotating left or right is a multi-clock operation, with a move of one bit position occurring per clock. Note that rotation of X/Y is identical to shifting, except for the manner in which the overflow is handled. Shift operations result in loss of the overflow, depending on the configuration selected. For individual X/Y shifts, the data is lost as it departs the register. For concatenated shifts, only the data departing the most-significant end of the X-register or least-significant end of the Y-register is lost. No loss of data occurs on rotate operations, since the contents circulate.

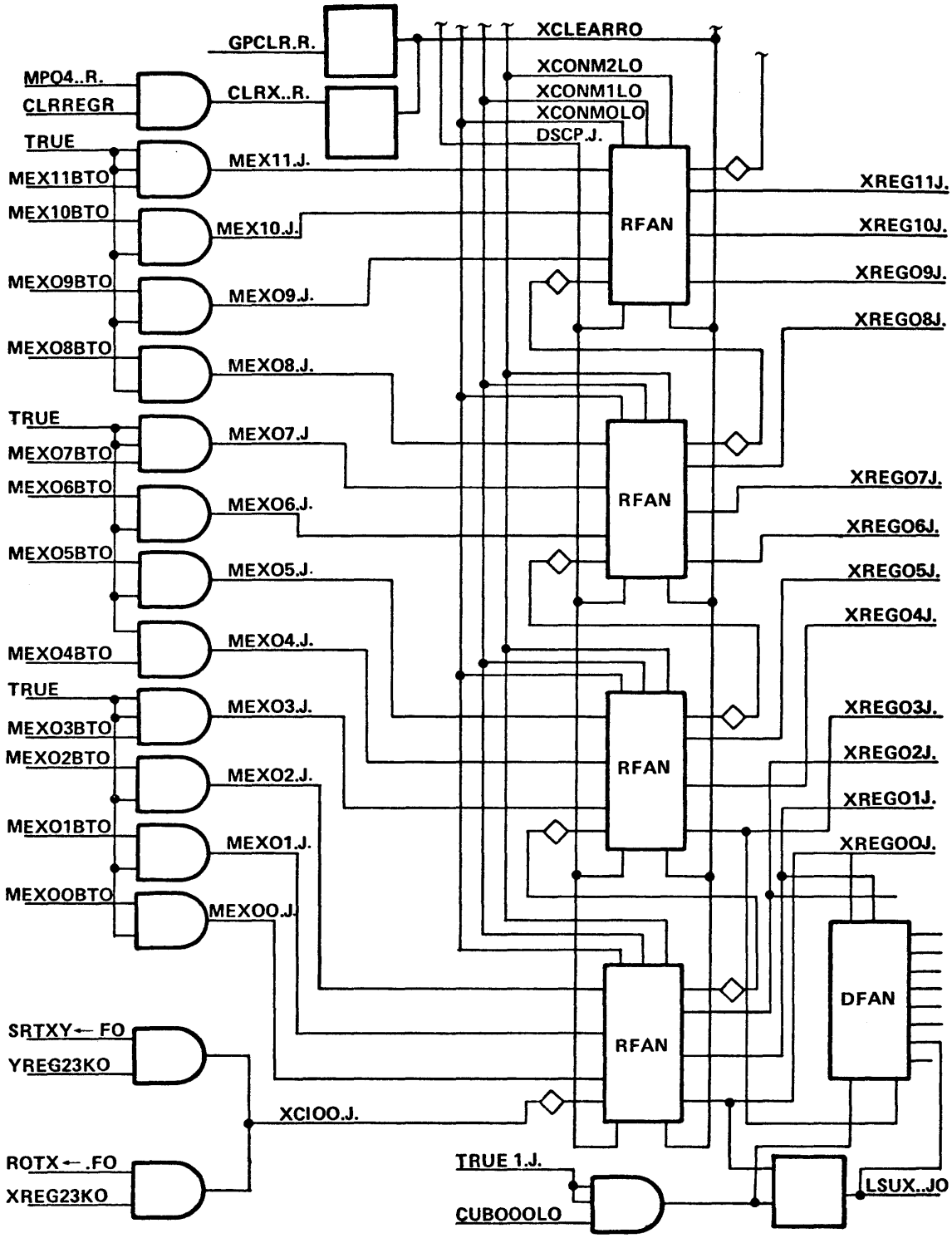


Figure 2-52. The X-Register

X- and Y- Register Control

The X/Y register control (figure 2-53) consists of decoding logic for generation of the X- and Y-mode control signals. Since X/Y operational mode is dependent on the micro operator being executed, the functioning of the control logic is discussed in terms of the micros which exercise it.

1C Micro: A 1C micro register move with X or Y as sink generates XCONM1LO or YCONM1LO, respectively, through buffers A or B. The inputs of A and B are 1NX...L. and 1NY...L. which are decoded from the register group selection levels, and the signal IGNOR1L., which is normally true. However, if binary/BCD sum or difference is sourced with X or Y as a sink, IGNOR/L. goes false for one clock period to await completion of the function box operation (to which X and Y were inputs).

The signal SLOW.24. achieves the same delay for all other sinks when sum or difference is sourced.

2C Micro: When sourcing a scratchpad word and moving it to X or Y, XCONM1LO or YCONM1LO, respectively, are generated via buffers A or B as decoded from the register group selection levels.

7C Micro: When memory information is transferred to X or Y, the signals X←MX...TO and Y←MX...HO generate XCONM1LO and YCONM1LO, respectively, via buffers H and L.

11C Micro: An extract T micro can sink its results into X or Y. In this case gates 5 and 6 select X or Y via buffers A and B; with MP05/.L. true enabling gate 5 and MP06/.L. true enabling gate 6.

4D/5D Micro: Under the control of MP06..L., which represents the shift/rotate directional control, gates 1, 2, 3, and 4 put the X- and Y-registers into the shift mode. With XCONTRL. true, XCONM1LO and XCONM2LO are generated via buffers C and F, placing X in the shift-up mode. MP06..L. true generates XCONM0LO, via buffer E, which, with the other two mode signals, places X in the shift-down mode. The same functions are generated identically for the Y-register, involving YCONTRL., MP06..L. and buffers I, J, and K.

3F Micro: When executing the normalize X micro, gate 12 and buffers M, D, and G generate XCONM1LO and XCONM2LO. This places X in the shift-up mode, a state which continues until either MSBX=1 or FL=0.

BINARY ADDER/SUBTRACTOR

Addition and subtraction within the 24-bit function box is, in all cases, performed in binary units. However, provisions have been made for the conversion of the adder/subtractor output to binary coded decimal (BCD) units when required. This conversion process is external to the adder itself, using special additional logic included for this purpose. The adder itself is a full 24-bit adder/subtractor consisting of six modulo-16 adders with associated carry logic. Refer to figure 2-54. Each 4-bit adder section utilizes two AFAN chips and one CFBN carry logic chip, plus a number of buffers and AND gates. The basic 4-bit adder unit may be broken down into three sections: the adder chips themselves, binary carry logic, and decimal carry correction logic. The latter section actually functions as part of the BCD sum correction logic, which is a different stage of the 24-bit function box.

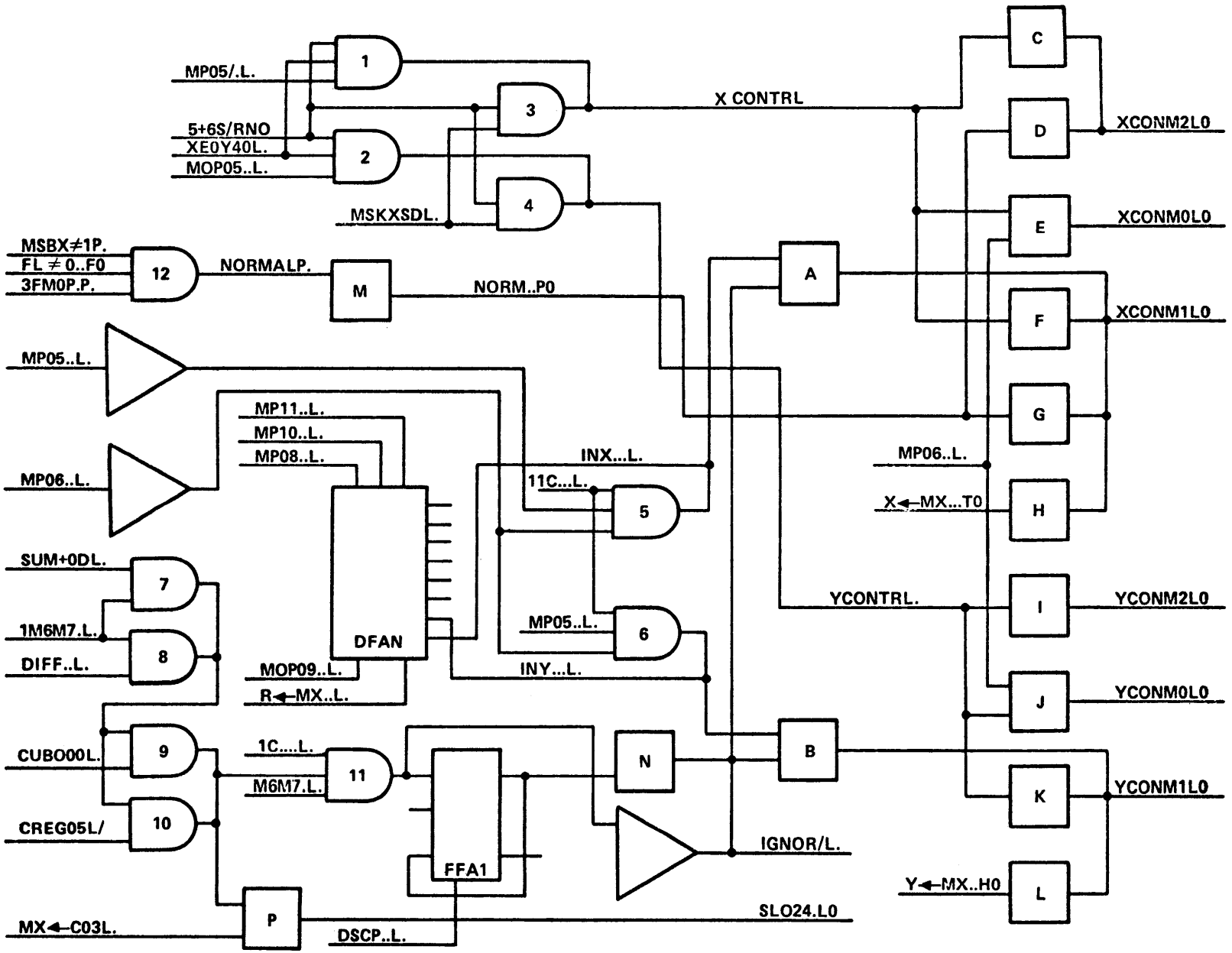


Figure 2-53. X/Y Register Control Logic

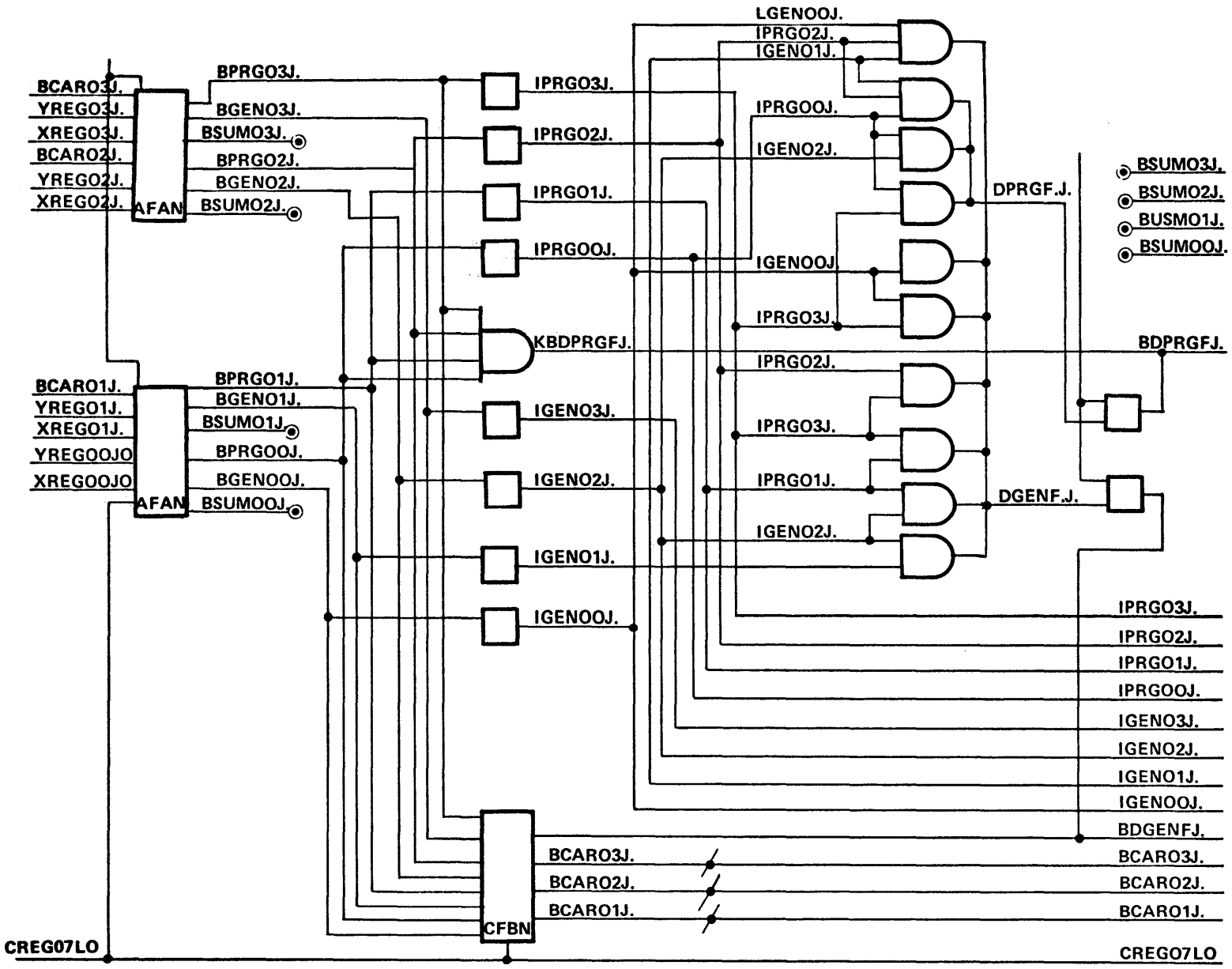


Figure 2-54. Binary Adder and Carry Logic

Add Example

In order to describe the logic functions, it is necessary to understand the arithmetic operation the circuit is designed to perform. Assume that a binary sum is to be produced when the X-register contains a value of eight (1000) and the Y-register contains a value of nine (1001). The desired result is (in binary) 10001, or seventeen. To accomplish the sum of these operands the logic will produce 0001 from the lowest 4-bit adder section, with a carry out to the next higher section. This is shown in figure 2-55. The handling of the carry thus produced is dependent on the field length set by CPL. If CPL has designated a field of four, then this carry would not be sent to the next section, but instead would be sent out of the function box as CYL. It would be a software function to detect the carry out or overflow and act accordingly. Note that the carry out, in the case of CPL designating a field length of four, is not automatically stored in the carry flip-flop. This may be accomplished only by executing the 6E micro, which is a separate operation.

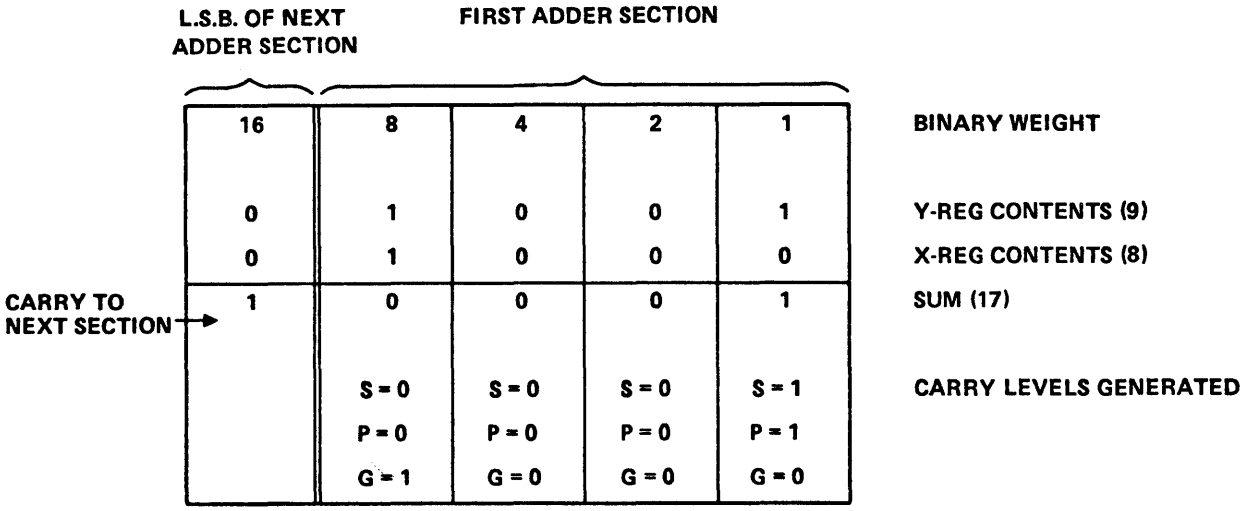


Figure 2-55. Binary Add Example

The actual functioning of an adder section is illustrated in figure 2-56, using the sum discussed in the previous paragraph as an example. Refer to the adder/subtractor description for the formulas from which the various internal control levels are derived.

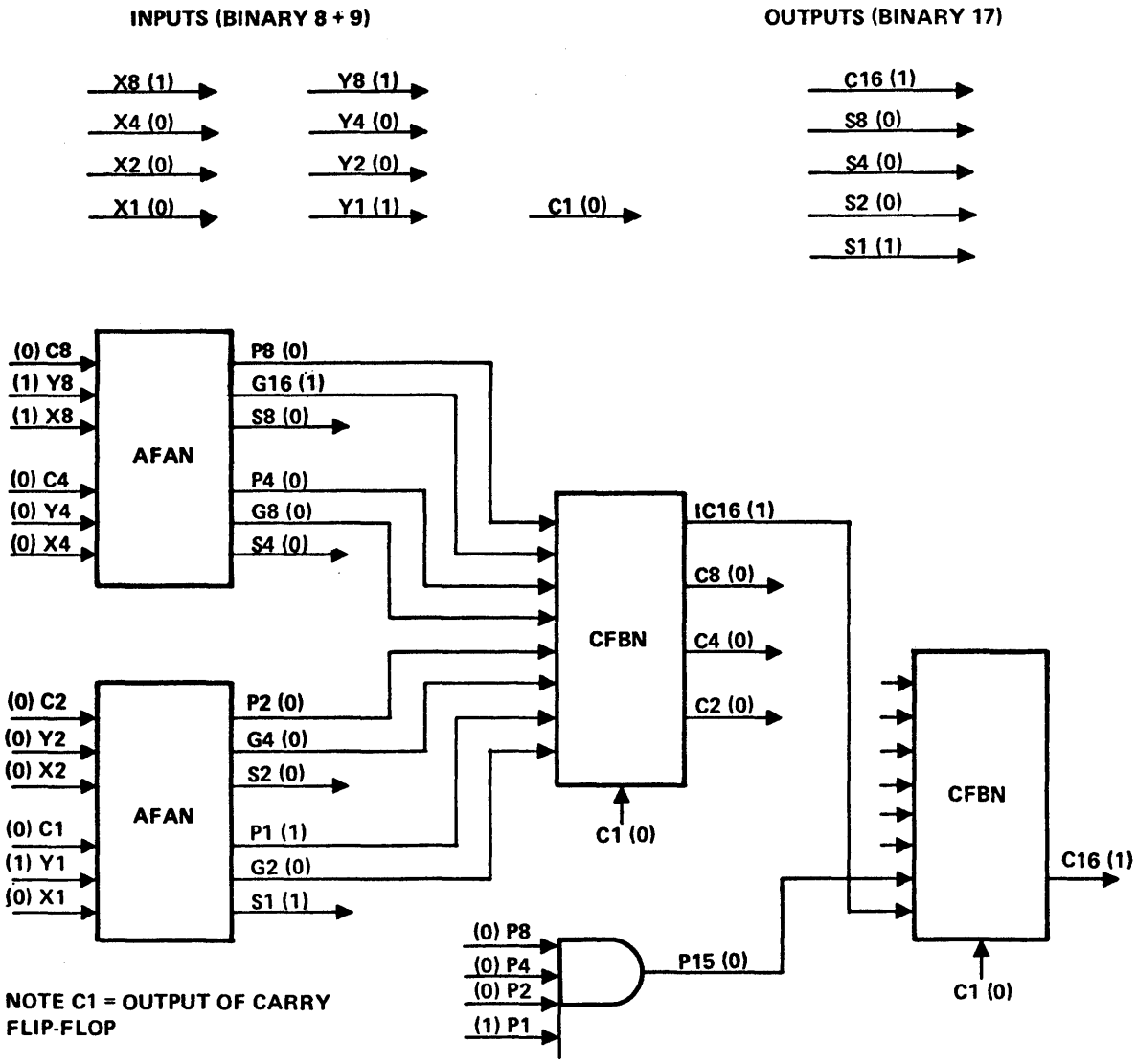


Figure 2-56. Add Example Logic Operation Showing Binary Weight of Signal Lines

BCD Sum/Difference Correction Example

Correction of a binary sum to BCD involves analyzing the binary value in 4-bit sections, and modifying those sections which contain a value greater than nine. This is necessary because the decimal numbering system includes number values from one to nine only, whereas four binarily-weighted bits can contain values up to sixteen. It is important to note that direct conversion of binary to BCD is not possible. That is to say, where a BCD result is desired, the operand inputs to the 24-bit function box must be encoded in BCD also. Although the adder/subtractor processes the operands as binary, correction of the result to BCD is possible simply by adding six to those 4-bit sections containing a value greater than nine, and sending a carry to the next higher section.

ADDITION

In example 1, a BCD value of 7 is added to a BCD value of 1, with the result being 8 (binary). Because the sum is less than 10, this result serves as well for BCD, and no correction is necessary.

```
Example 1:  (7+1)      0111 = 7
                      +0001 = 1
                      ---------
                       1000 = 8     No correction needed.  Valid BCD value.
```

The second example adds BCD7 to BCD8, to obtain a binary value of 15. To produce a decimal 15, six must be added to the initial sum, causing the lower four bits to wrap around to a value of 5, with a carry of 1 bit being sent to the next higher section (BCD 15 = 0001 0101).

Correction of the result is similarly required when performing a BCD subtract operation. In this case, the output of any 4-bit section which has received a borrow from the next higher section must be corrected by subtracting six.

```
Example 2:  (7+8)      0111 =  7
                      +1000 =  8
                      ----------
                       1111 = 15    Correction necessary.  Invalid BCD value.
```

The correction process is illustrated below:

```
Correction:           0000 1111 = 15
                     +0000 0110 = 6
                     --------------
                      0001 0101 = 10+5    Note that BCD 0001 is decimal weight 10 by
                                          position.
```

SUBTRACTION

In example 3, a borrow from the next higher section did not occur; therefore BCD correction was not required.

```
Example 3:  (9-7)      1001 = 9
                      -0111 = 7
                      ---------
                       0010 = 2     Result correct in either binary or BCD.
```

In the fourth example the binary adder assumed that the subtrahend was 23, and therefore gave a difference of 14. In actuality, the BCD value in the adder was 17, which would give a difference of 8. Therefore, 6 must be subtracted from the adder output to obtain the correct result. If the value of the entire binary or BCD number in the X-register is less than the value in the Y-register, then a borrow out from the function box will be produced. This borrow out is known as CYD, and like CYL produced by all operations, must be handled by software.

```
Example 4:  (17-9)     0001 0111 = 17      Binary 23.
                      -0000 1001 =  9
                      --------------
                           1110 = 14      Result not a valid BCD character.
```

The subtract BCD correction process may be illustrated as follows:

```
Correction:           1110 = 14
                     -0110 = 6
                     ---------
                      1000 = 8     Correct result.
```

Adder Chips

Operationally, the adder chips each represent two bit positions of the complete unit, having inputs for the contents of the corresponding X- and Y-register bits, plus carries from the carry logic. Outputs of the AFANs include sum, which is in fact the result output, plus generate and propagate which are used for generation of carries. The AFAN chips are not clocked, and operate subject to only one mode control line, SUBM11J. or SUBM11K. (depending on which section of the 24-bit adder/subtractor is being discussed). When the mode control line is false, the chip operates as an adder, and when true as a subtractor.

Logic formulas for production of the sum, generate, and propagate outputs from each 4-bit section of the adder are as follows:

$$Sn\ (sum) = Cn \cdot (\overline{An} \cdot \overline{Bn} + An \cdot Bn) + \overline{Cn} \cdot (\overline{An} \cdot B\ n + An \cdot Bn)$$

$$Gn\ (generate) = MO \cdot \overline{An} \cdot Bn + \overline{Mo} \cdot An \cdot Bn$$

$$Pn\ (propagate) = Mo \cdot (\overline{An} \cdot \overline{Bn} + Sn \cdot Bn) + \overline{Mo} \cdot (An \cdot Bn + An \cdot \overline{Bn})$$

In addition, the generate and propagate terms may be defined as follows:

Propagate: Propagate is produced if an individual section of the adder has the potential of sending a carry to the next state of the adder. This signal is produced whenever either operand input is true, regardless of whether or not an incoming carry is received. Therefore, it may be stated that propagate indicates anticipation that a carry is coming. If a carry does, in fact, come to this section, it simply passes it on to the next stage of the adder.

Generate: Generate is produced if an individual stage or section of the adder definitely produces a carry. A carry is produced by a stage or section only if both operand inputs are true.

Mask Generator

The mask generator (figure 2-57) is essentially a circuit which decodes the binary-coded field length stored in C-register bits 00 through 04, producing up to 24 1-bit enabling signals which allow gating the output of the 24-bit function box to the MEX. The mask bits (TMSK00J. through TMSK11J. and TMSK12K. through TMSK23K.) are generated in groups of four by CFBN chips, which receive the decoded CPL input signals. It should be noted that the CFBN chips are operated with all propagate inputs tied true. This changes their mode of operation considerably with respect to the usual carry logic application. In this configuration, the Boolean equations governing the use of the CFBN chip are as follows:

$$G3 = C3$$

$$G2 = C2 \cdot C3$$

$$G1 = C1 \cdot C2 \cdot C3$$

$$G0 = C0 \cdot C1 \cdot C2 \cdot C3$$

$$CI = C0 \cdot C1 \cdot C2$$

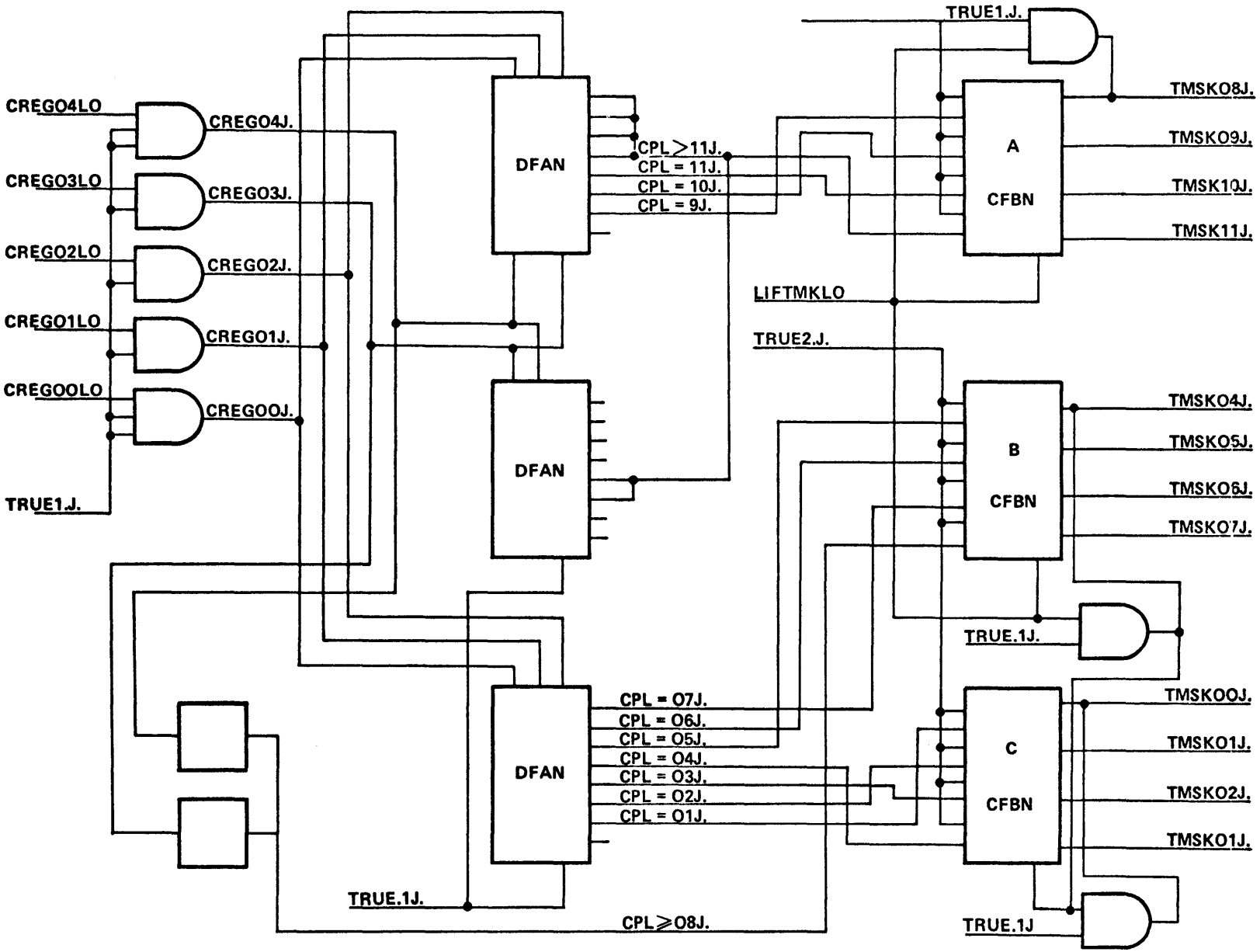


Figure 2-57. Mask Generator Bits 00 thru 11 (Sheet 1)

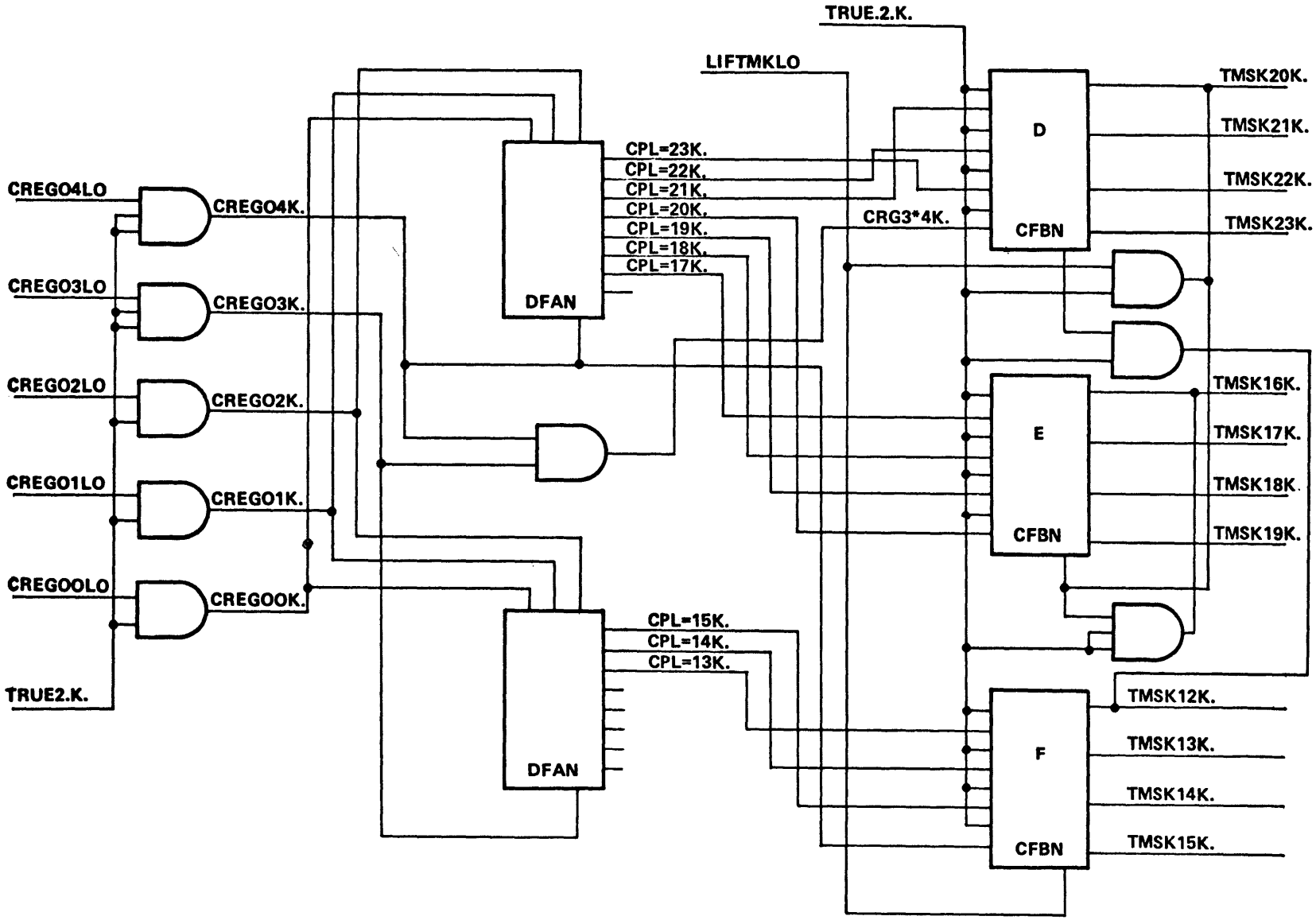


Figure 2-57. Mask Generator Bits 12 thru 23 (Sheet 2)

This provides the ability to create an additional effect whereby the most-significant mask bit which is true causes all those below it to be true also. In addition, use of the CI input allows a secondary means of forcing the mask bits true, and this is accomplished with the LIFTMKLO signals. In practice, LIFTMKLO forces the three most-significant mask bits within a CFBN chip true by way of the Cl input, and the fourth bit by way of an external gate.

Various methods are used to provide the additional effect between 4-bit sections of the mask generator. A summary of these is as follows:

Value in CPL	Tie-In Method
CPL > 03	TMSK04J tied to CI Input of CFBN A in figure 2-57.
CPL > 07	CPL $\geq$ 08J. signal derived from CREG03J and CREG04J tied to CI input of CFBN B in figure 2-57.
CPL > 11	CPL $\geq$ 11.J. signal decoded from CREG 00-04 tied to GO input of CFBN C in figure 2-57.
CPL > 15	CREG04K. tied directly to CO input of CFBN D in figure 2-58.
CPL > 19	TMSK20K. tied to CI input of CFBN E in figure 2-58.
CPL = 24	CRG3*4K signal derived from CREG03K. and CREG04K. tied to CO input of CFBN E in figure 2-58.

The lift mask control logic, as shown in figure 2-58, can generate the signal LIFTMKLO in response to a number of inputs. Buffers 1 or 3 lift the mask if the X- or Y-registers are sourced during a 7C micro operation. Gates 4 and 5 lift the mask via buffer 2 if MAXS or MAXM is sourced, provided that register select column 3 (M6M7B1L) is true. Gate 6 lifts the mask when the CP register is sourced, and gates 7 and 8 are used when the X- or Y- registers are sourced during a 1C or 2C register move micro. Gates 6, 7, and 8 are enabled by 2M6/M7L, which designates register select column 2.

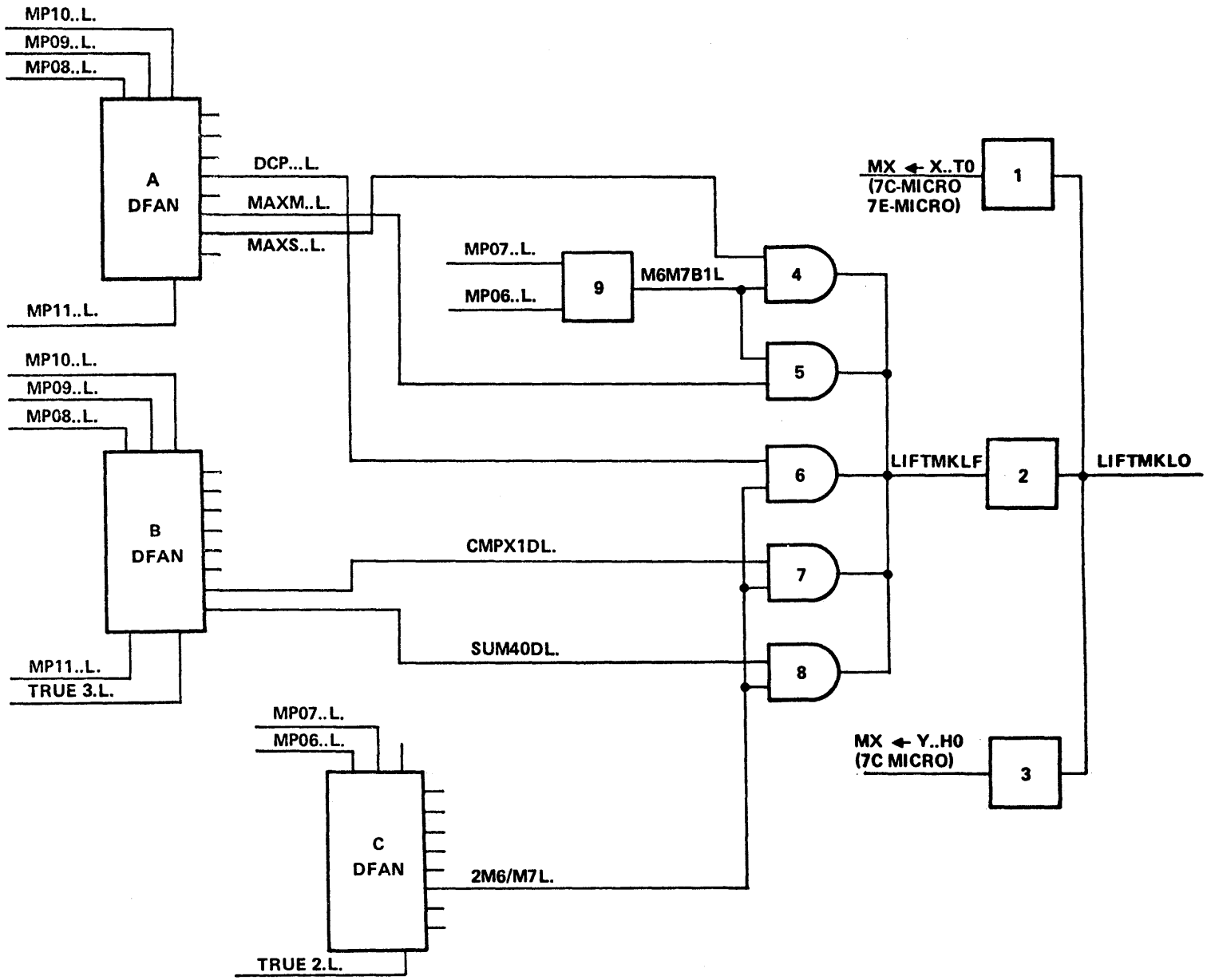


Figure 2-58. Lift Mask Control Logic

Binary Carry Logic

The binary carry logic for each 4-bit adder section consists of one CFBN chip, plus a 4-leg AND gate. Refer to figure 2-59. This portion of the circuit receives the generate and propagate outputs of the adder chips, using them to produce three carry levels (for use within the 4-bit section) plus one each secondary generate and propagate level (for use in producing a carry level to the next 4-bit section).

The secondary carry logic consists of an additional CFBN chip which receives the secondary generate and propagate levels from each of the three 4-bit adder sections in one-half of the 24-bit adder. From these are produced the intersection carries which have binary weights equal to increments of 16 (bit positions 04, 08, 12, 16, 20, and 24).

The following equations and descriptions of equations are intended to explain the production of the carry terms and the significance of the propagate, generate and carry terms in the equation. All terms are referenced to the binary weight as shown in figure 2-59.

C2=(G2)+(P1·C1): carry out of the first stage results from the X- and Y- inputs being both equal to 1 (1+1=0 with a carry out) or with either X or Y equal to 1 (1+0 or 0+1) and a carry in to the stage (1+0+C1=0 and carry out).

C4=(G4(+(P2·G2)+(P1·P2·C1): G4 indicates a carry which was generated by the inputs to this stage of the AFAN (1+1=0 with a carry out). P2·G2 indicate a generation of a carry in the previous stage and one of the inputs to this stage is true (either X or Y). P1·P2·C1 indicate a carry from the previous stage (P1·C1) along with one input to this stage.

C8=(G8)+(P4·G4(+(P4·P2·G2)+(P4·P2·P1·C1): In a similar manner, C8 recognizes a generation of a carry in this stage (G8) or propagate and generate terms developed in previous stages which cause a carry out of this stage.

C16=(IC16)+P8·P4·P2·P1·C1): C16 is produced as a result of an internal carry 16 which actually is a result of G16 or combinations of the propagate and generate terms from the previous stages. The following is the formula for the IC16: (G16)+(P8·G8)+(P8·P4·G4)+(P8·P4·G4)+(P8·P4·G8·G2).

BCD Carry Correction Logic

The BCD carry correction logic (figure 2-60) consists of a network of AND gates fed by the generate and propagate outputs of the 4-bit adder section. Its function is to force the generation of an inter-section carry whenever the adder output exceeds a binary value of 8. These gates monitor the propagate and generate outputs continuously, but their own outputs (DPRGF.J. and DGENF.J. in the example) are gated to the secondary carry logic only when DGPRONJ. (decimal mode) is true.

The generate and propagate adder outputs which feed the BCD carry correction logic are buffered for isolation. The buffers also feed the generate and propagate levels to the output multiplexors of the 24-bit function box, where they are used to produce the XORY and XANY functions, respectively.

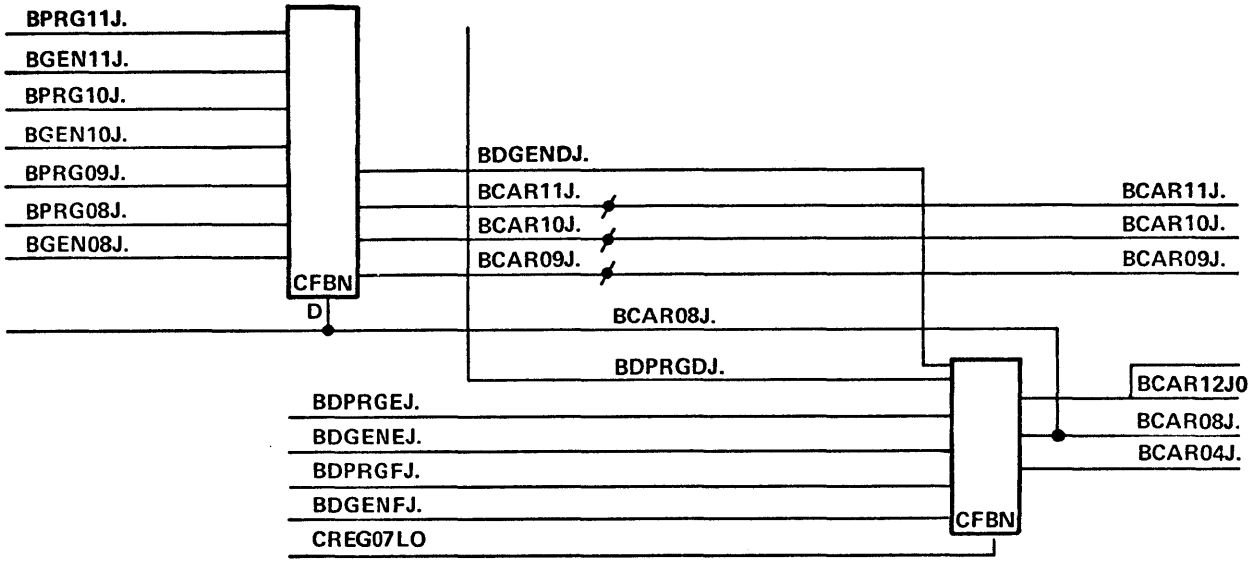


Figure 2-59. Binary Carry Logic

4-Bit BCD Sum Correction Logic

If, in any decimal add operation, the binary sum at the output of a 4-bit adder section is greater than nine, it must be decimally corrected by adding a value of six, and sending a carry to the next higher section. A similar situation occurs on decimal subtract operations whereby a value of six must be subtracted from any 4-bit section where a borrow from the next higher section has occurred. The BCD sum correction is accomplished by the logic shown in figure 2-61, which is designed to add/subtract six only. Refer to the 4-bit BCD sum/difference example for a discussion of the add/subtract six requirement.

The BCD correction is accomplished in a separate stage to avoid conflict with the natural generation of binary carries involved in producing the binary result. The correction logic is made up of AFAN chips which receive both the binary sum and the decimally-corrected inter-section carries.

As may be seen in figure 2-62, whenever the inter-section carry (BCAR04J in this case) is present, a value of six is added to the contents of the three most-significant BSUM lines of the group. The least-significant BSUM line is not affected because adding a binary value of six cannot change its status. Since a value of six is added only when the 4-bit group originally contained a value greater than nine, the result always wraps around to a value between 0 and 6.

Unlike the sum operation, there is no correction of the carry/borrow for the BCD difference. Like the sum operation, however, if a borrow is present from one of the 4-bit sections, then the difference result in the next lower section must be corrected. When a subtract operation is being performed, the mode line (DSUBMOJ. in this case) is true, causing the AFAN chips to operate in the subtract mode.

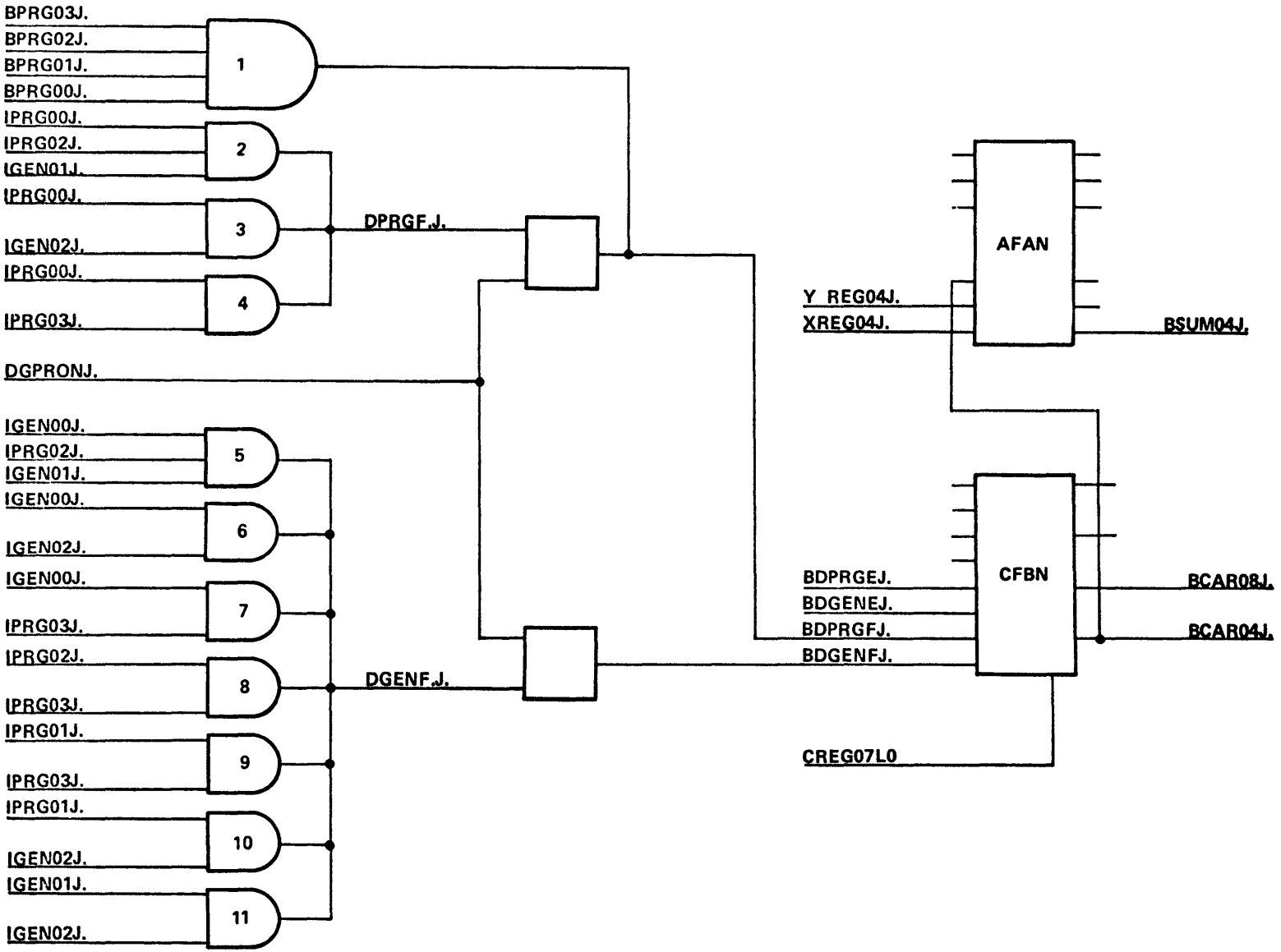


Figure 2-60. BCD Carry Generation

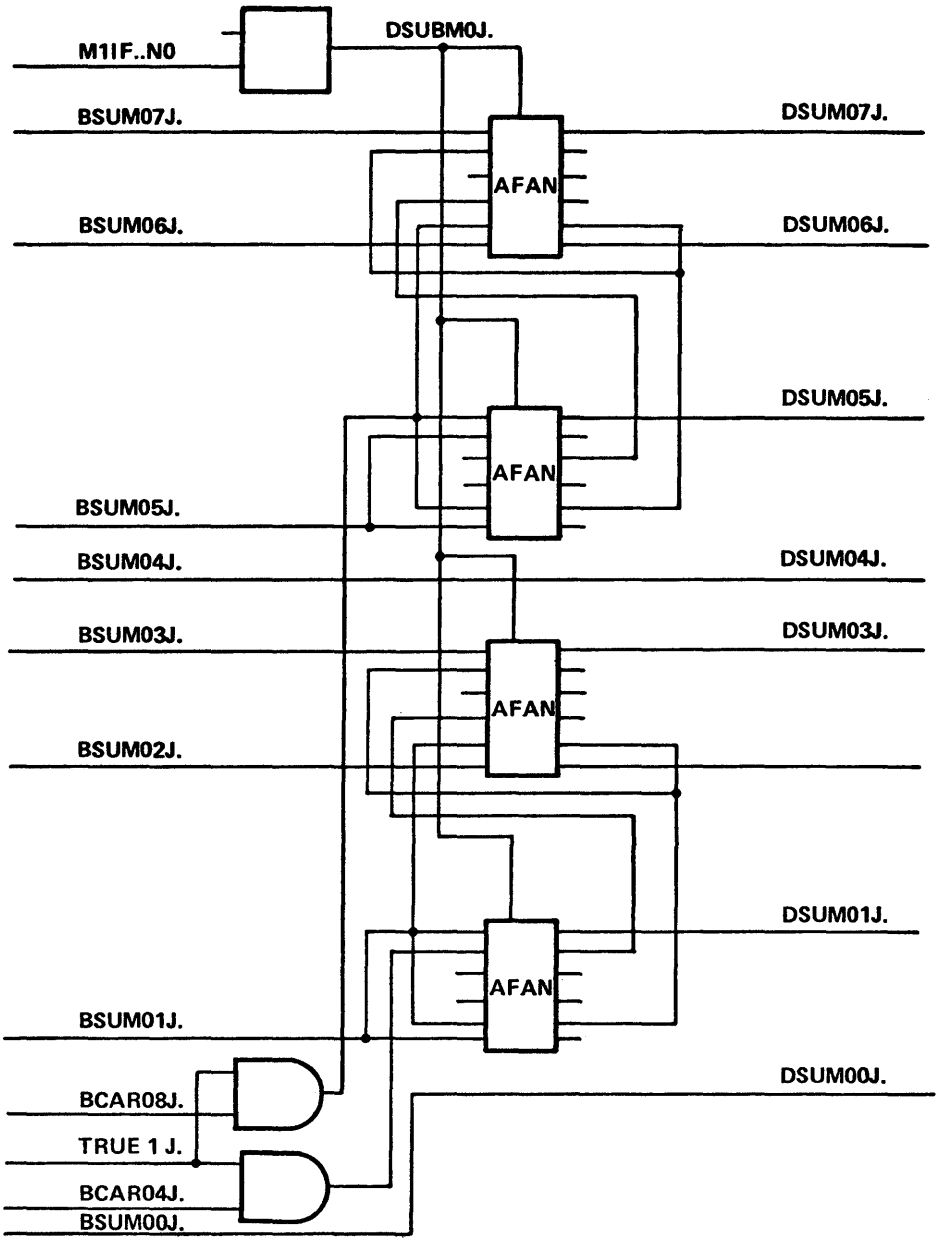


Figure 2-61. BCD Sum Generation

STATIC COMPARE LOGIC

The static compare logic consists of circuits which monitor the contents of X and Y, plus the function box carry and borrow outputs and the status of the carry flip-flop (from the C-register). Since these circuits differ considerably from one another, they are described according to function.

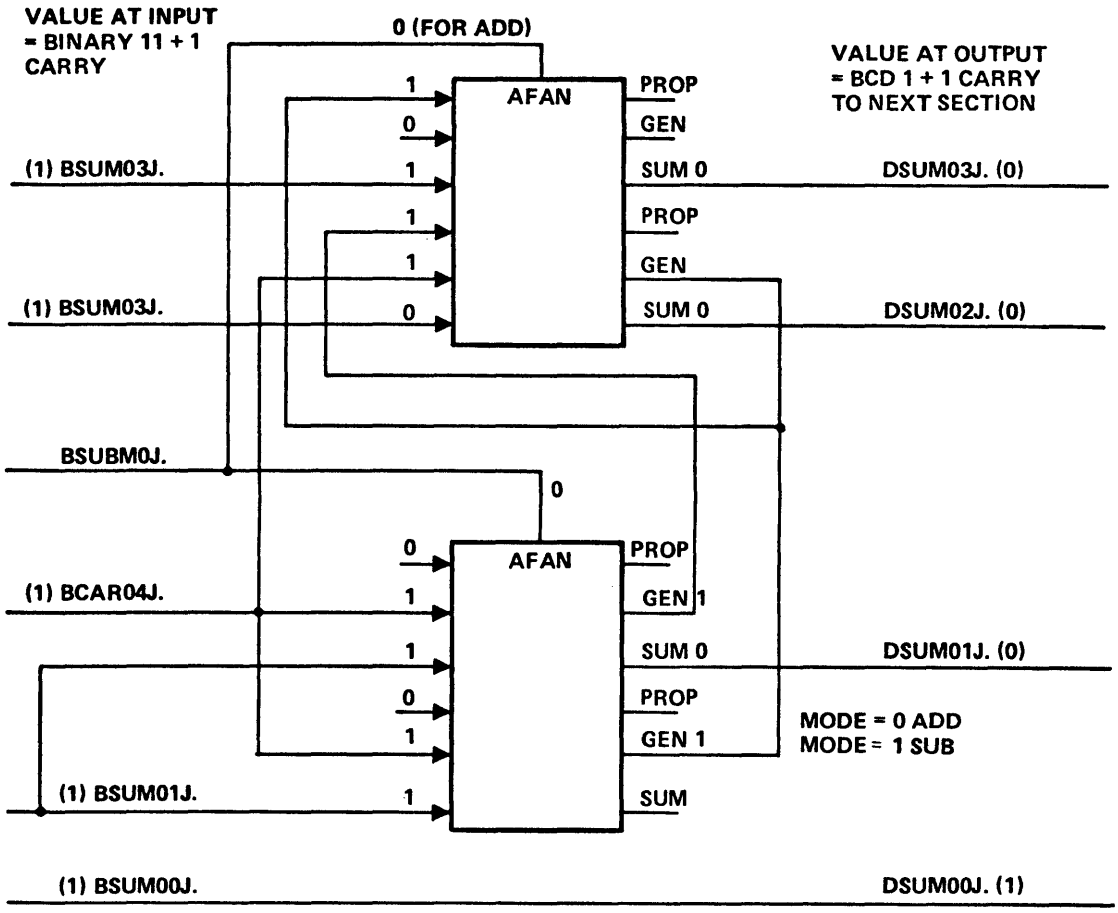


Figure 2-62. BCD Sum Correction Example

Most-Significant Bit of X (MSBX)

Contents of the most-significant bit of X (as referenced by CPL) is detected by MFAN multiplexor chips. Inputs to the MFANs are the bits from the X-register (except for bit 23). Refer to figure 2-63. Since it is desired to monitor the contents of one bit position only, both enabling and input selection for the MFANs is controlled. Bit selection is by the contents of bits 00 through 02 of the C-register (CREG00L0 through CREG02L0), while chip enabling is accomplished by additional logic which monitors the contents of CREG03L0 and CREG04L0, and is actually part of the mask generator.

For CPL values 00 through 07, the level CPL < 08J. is true, enabling MFAN I1, which causes the input selected by CREG00J. -CREG02J. to become MSBX..T0. Note, however, that CPLF00 always produces a zero output, since the least-significant input to I1 is not used. For CPL values 08 through 15, MFANs I0 (on card J) and E2 (on card K) are enabled, monitoring the condition of X-bits 07 through 15, inclusive. CPL816J. and CPL816K. enable the two chips, respectively. Although two chips are enabled at one time, conflicts do not occur because different inputs are used on each. The outputs of all MFAN chips are ORed to produce MSBX..T0. MFAN D0 (on card K) monitors X-bits 16 through 23, being enabled by CP1624K.

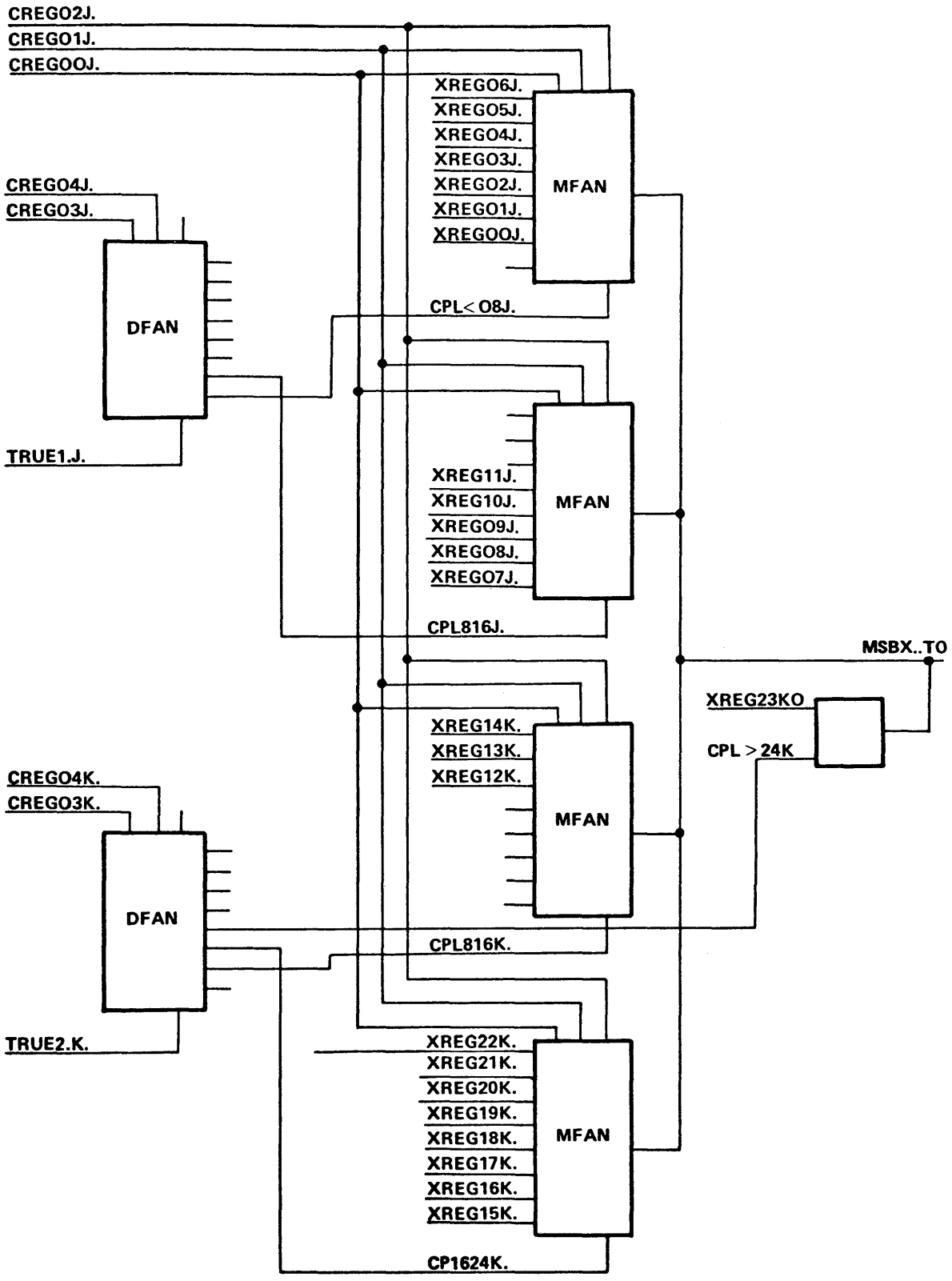


Figure 2-63. The Most Significant Bit of X-Logic

X Not Equal to Zero (X≠0) and Y Not Equal to Zero (Y≠0)

The X and Y not equal to zero logic (figure 2-64), like the MSBX circuit, consists of elements which monitor the status of each register bit. In this case, however, all such elements are active on a continuous basis, with the outputs of each group (those monitoring X and Y, respectively) ORed together, so that a true condition on any register bit will produce a true output. Using the Y≠0 logic as an example, X-register bits 0 through 11 feed an EFAN chip and four buffers. Only the overall OR output of the EFAN is used, and this is tied together with all the buffer outputs. The Y≠0...TO output of this circuit is ORed with the output of an identical circuit which monitors Y-register bits 12 through 23. An identical pair of circuits monitors the contents of the X-register to produce X≠0...TO.

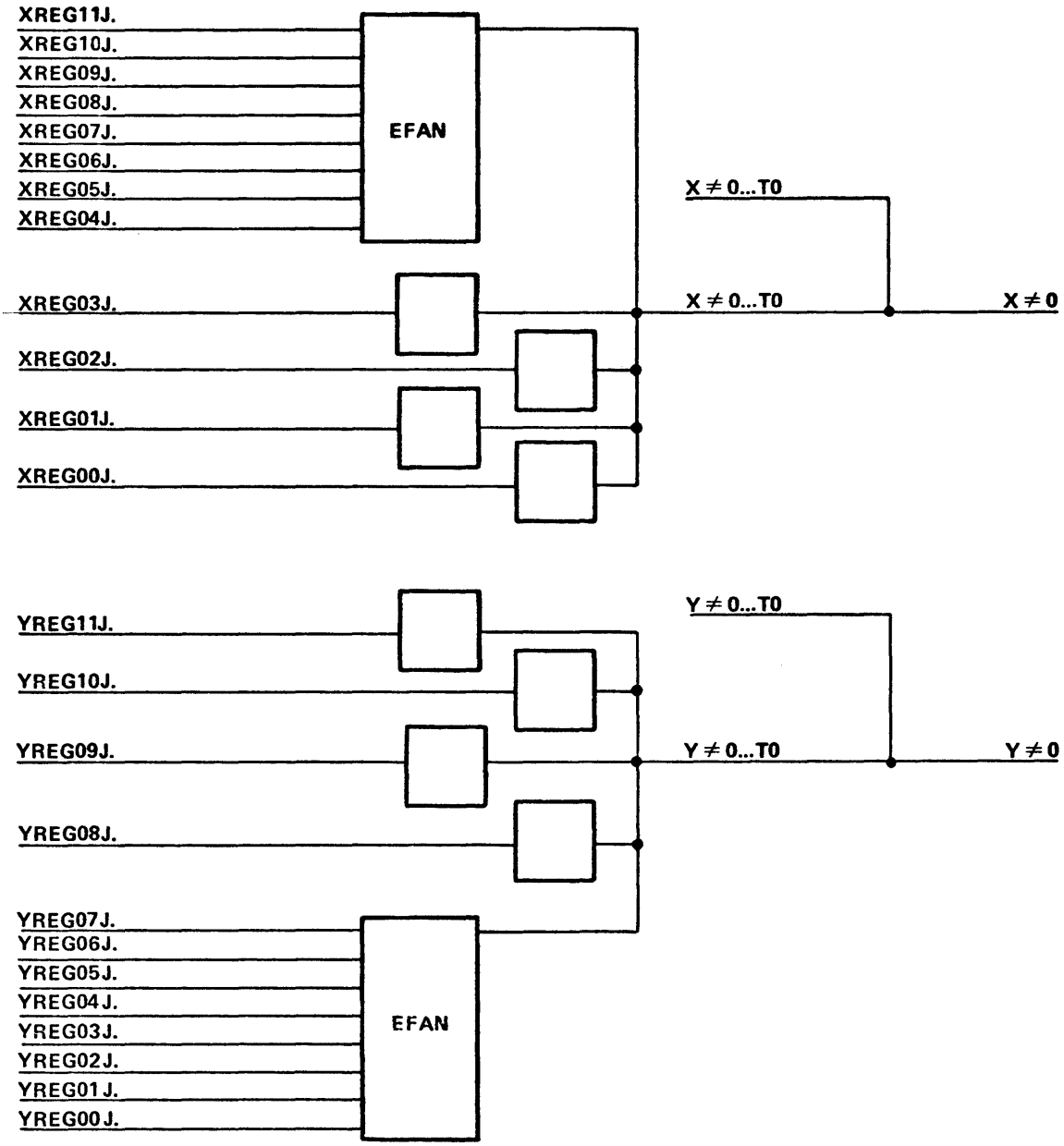


Figure 2-64. The X/Y Register States (XYST)

X Compared With Y (X=Y, X>Y, X<Y)

The X compared with Y logic (figure 2-65) consists of six CFAN chips which monitor all 12 bit positions in each register. These CFANs are divided into two groups corresponding to the physical split of the X and Y registers and the 24-bit function box, (with bits 00 through 11 on processor card J and bits 12 through 23 on processor card K). Since the two halves are identical, only the portion dealing with the lower 12 bits are described. The "equal," "greater than" and "lesser than" outputs of each of the CFAN chips are utilized, being combined by output logic as described in the following Boolean equations.

$$(X=YDEFJO) = (X=YGPFJ.) \cdot (X=YGPEJ.) \cdot (X=YGPDJ.)$$
$$(X>YDEFJO) = (X>YGPDJ.) + (X=YGPDJ.) \cdot (X>YGPEJ.) + (X=YGPDJ.) \cdot (X>YGPEJ.) \cdot (X>YGPEJF.)$$
$$(X<YDEFJO) = (X<YGPDJ.) + (X=YGPDJ.) \cdot (X>YGPEJ.) + (X=YGPDJ.) \cdot (X=YGPEJ.) \cdot (X<YGPFJ.)$$

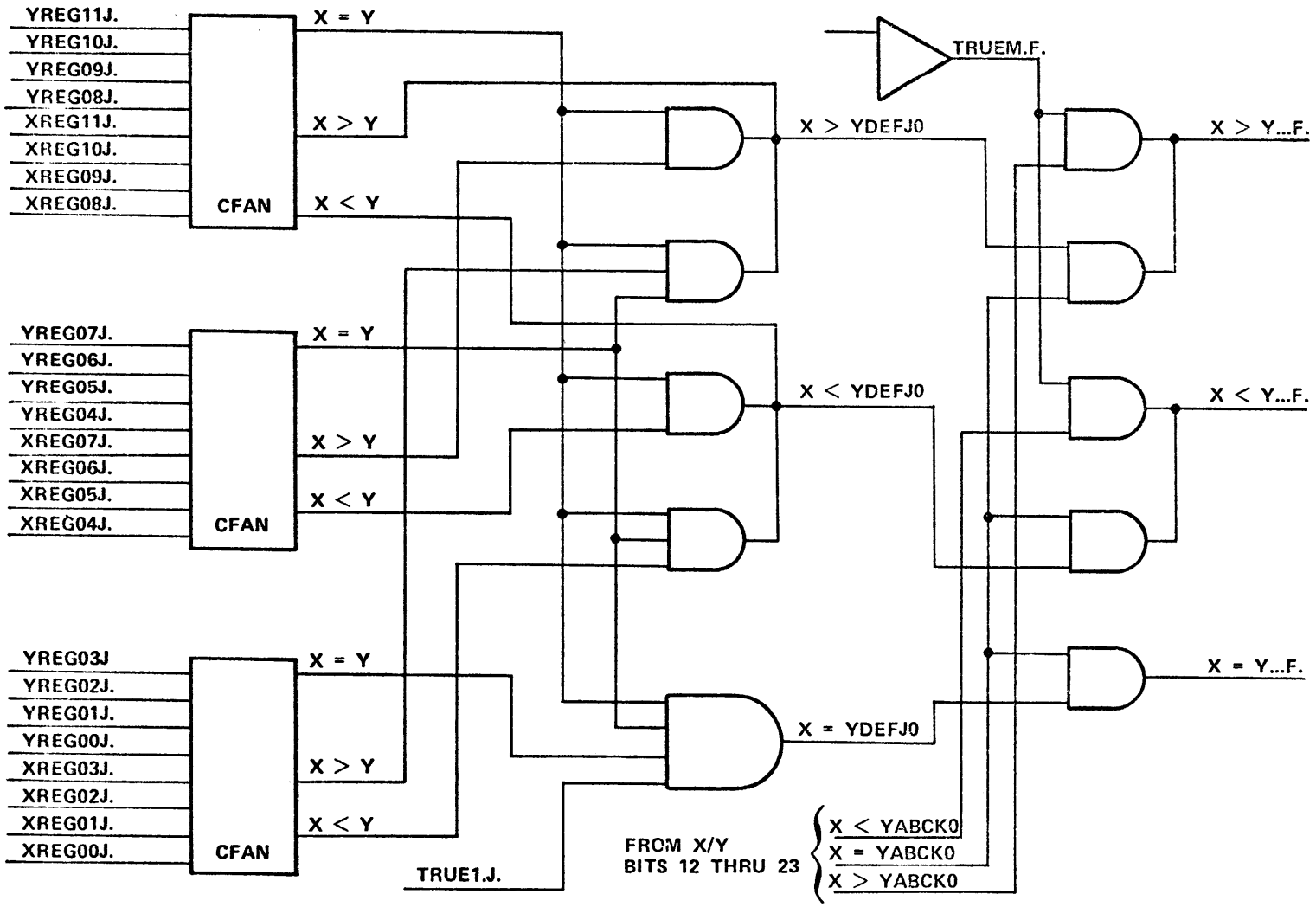


Figure 2-65. X/Y Register Conditions (XYCN)

The X/Y compare output signals of the two groups are further combined in a secondary level of gating which is located on processor card F. The following boolean equations describe the relationship between the group output signals and the final XYCN levels:

$$(X=Y..F.) = (X=YABCKO) \cdot (X=YDEFJO)$$
$$(X>Y...F.) = (X>YABCKO) + (X>YDEFJO) \cdot (X=YABCKO)$$
$$(X<Y...F.) = (X<YABCKO) + (X<YDEFJO) \cdot (X=YABCKO)$$

Least-Significant Unit of Y (LSUY)

The least-significant unit of Y logic is identical to the same circuit in the X-register (LSUX). Refer to the discussion of that circuit if further information is required. LSUY is bit 3 of the BICN pseudo register, and indicates, when true, that bit 0 of Y is true in the binary mode, or that bits 0 and 3 of Y are true in the BCD mode.

Carry Level (CYL)

The carry level (bit 0 of BICN) is generated whenever a most-significant carry has been generated during an add operation. A most-significant carry may be defined as any which would produce a result greater than the field length specified by CPL. The logic for generating CYL is shown in figure 2-66. Briefly, it consists of a series of MFAN multiplexor chips which monitor the status of all 23 carry lines, plus C-register bit 7. Only one of the MFANs is enabled at any given time, this being determined by the value in CPL as detected by the two DFAN chips. Therefore, the contents of that carry line selected by the contents of CREG00J. through CREG07J. becomes CYL...TO. CREG07L0 (carry flip-flop) is CYL...TO when CPL=0. Note that a separate gate generates CYL...TO when CPL$\geq$24K. and ICAR24K. are true.

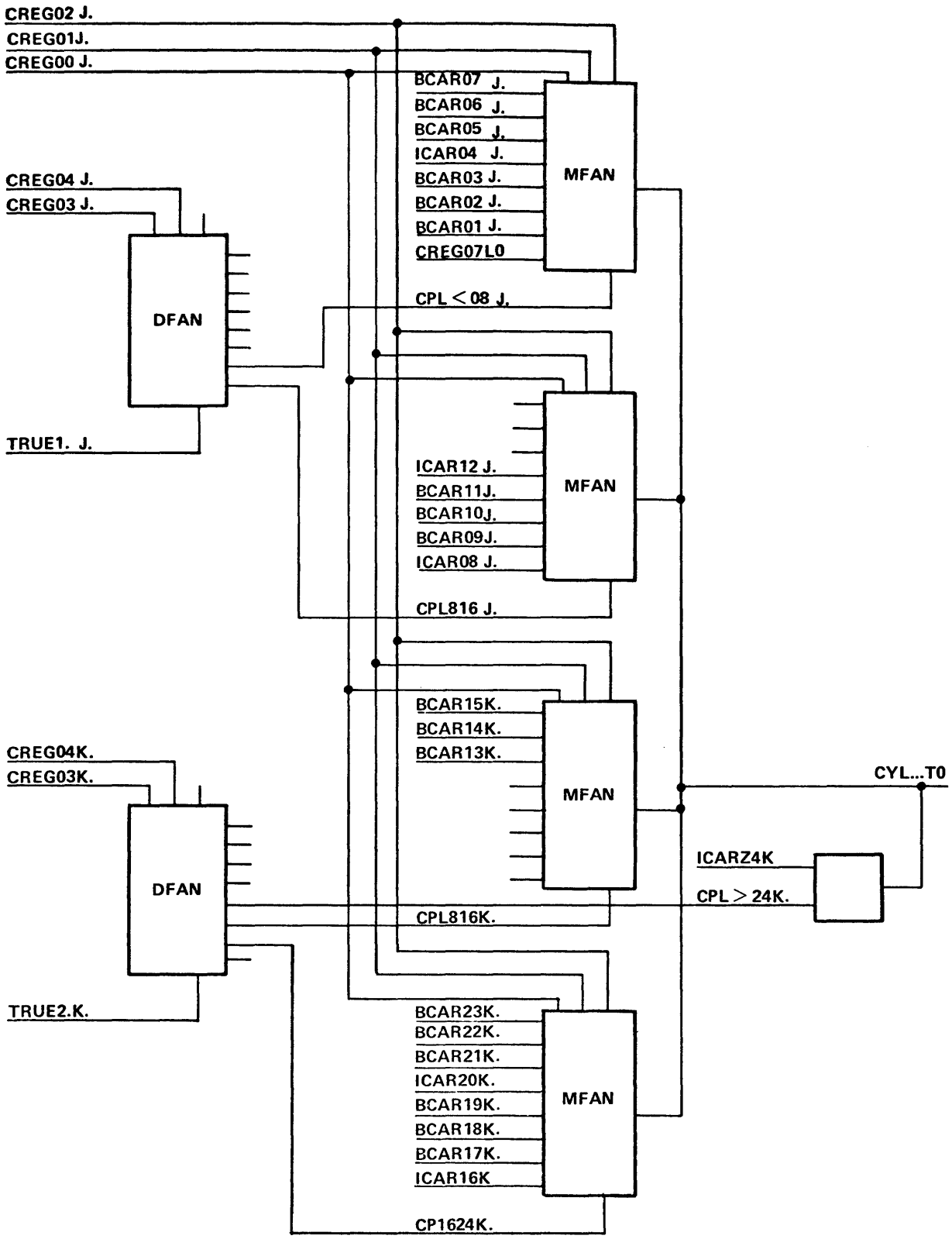


Figure 2-66. Carry Level (CYL) Generation

Borrow Level (CYD)

The borrow level, when true, indicates that a borrow has been generated during a subtract operation. CYD is true when either Y>X or if X = Y. In both cases CYF is true. CYD...FO is derived from the outputs of the X/Y static compare logic, and the carry flip-flop as shown in figure 2-67. CYD is bit 1 of BICN.

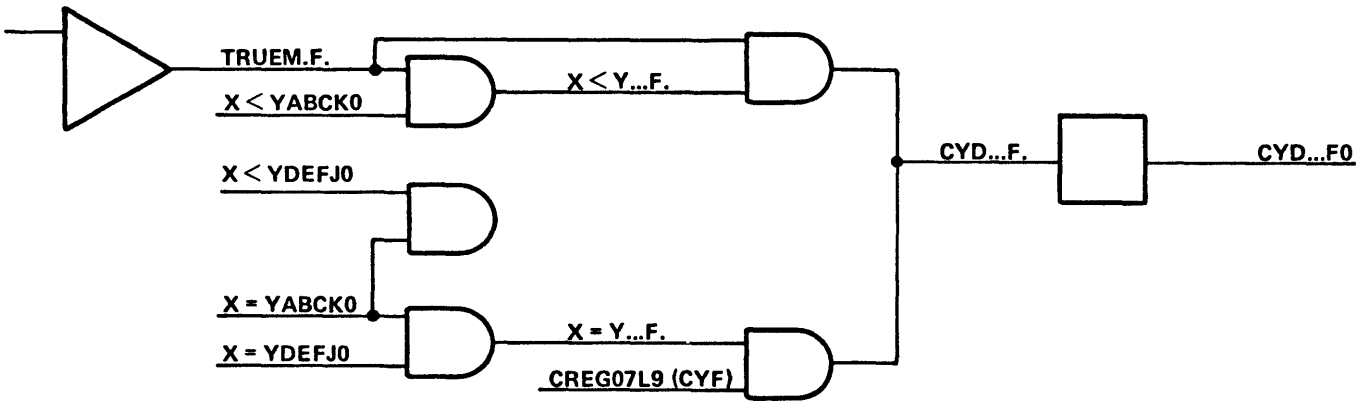


Figure 2-67. CYD Generation

Carry Flip-Flop (CYF)

Bit 2 of BICN contains the output of the carry flip-flop, which is not a part of the 24-bit function box. CYF is an independent flip-flop which is manipulated by the manipulate carry flip-flop micro (6E).

CYF is actually C-register bit 07, and may be set (through gates) in four different ways (Refer to figure 2-68):

Gate 1: variant 2 of the 6E micro sets CYF unconditionally.
Gate 2: variant 3 of the 6E micro sets CYF if CYL is true.
Gate 3: variant 4 of the 6E micro sets CYF if CYD is true.
Gate 4: when the signal CP←MX.L. is true, MEX07BTO (MEX bit 7) true sets CYF. This occurs when the CP register is designated sink on a register move micro.

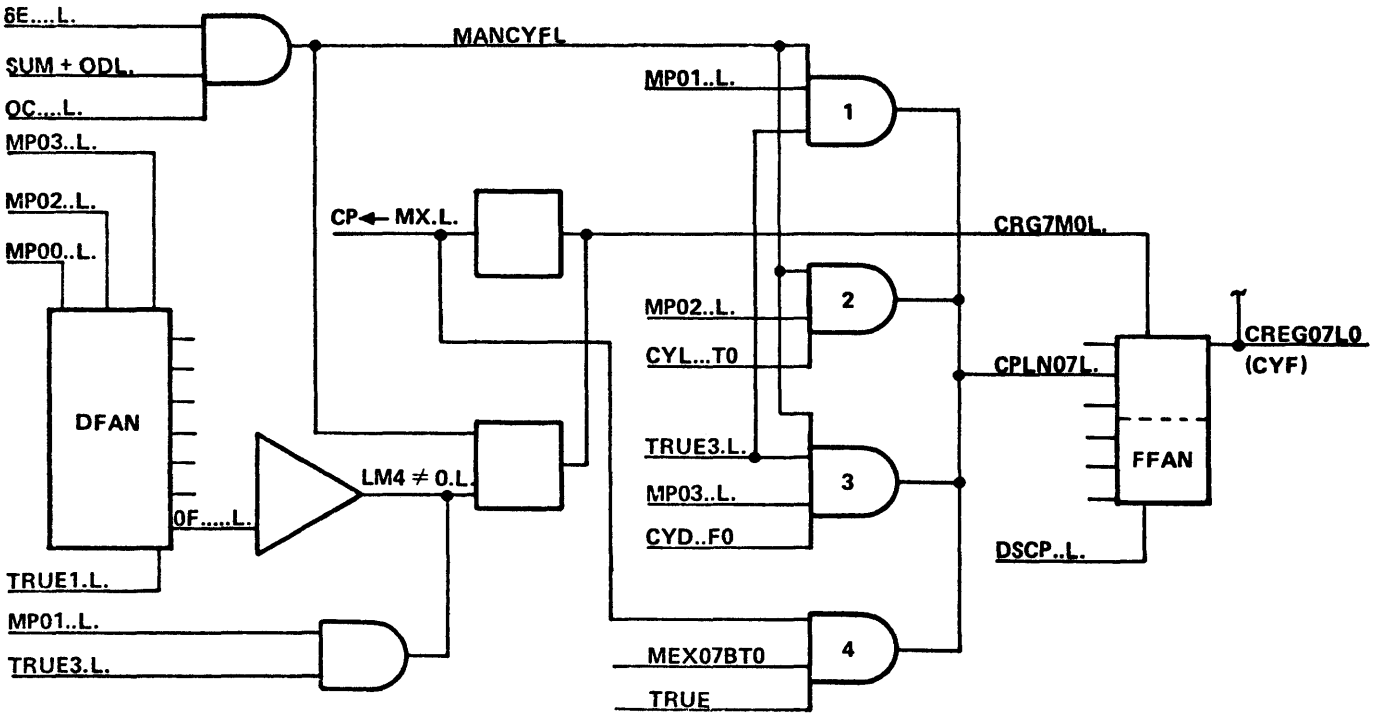


Figure 2-68. Carry Flip Flop (CYF) Logic

X/Y FUNCTION LOGIC

The special X/Y functions such as X AND Y (XANY) and X OR Y (XORY) are produced by a number of different methods. In some cases, this involves merely utilizing an intermediate output of the binary adder/subtractor, while in others, discrete logic is provided to accomplish the desired function. The derivation of each function is described in the following paragraphs:

X AND Y (XANY)

The X AND Y function is taken directly from the generate outputs of the adder/subtractor. These output lines appear as IGEN00J. through IGEN11J. and IGEN12K. through IGEN23K. at the output multiplexor.

X Exclusive OR Y (XEOY)

The X Exclusive OR Y function is taken directly from the propagate outputs of the adder/subtractor. These output lines appear as IPRG00J. through IPRG11J. and IPRG12K through IPRG23K. at the output multiplexors.

X OR Y (XORY)

The X OR Y function is produced by gating both the X- and Y-register contents to the MEX, simultaneously. This is brought about by the MX←X..L0 and MX←Y..L0 enabling signals being true at the same time. Refer to figure 2-69. Note that the output gates of both X and Y feed common lines (TXY00.J. through TXY11.J. and TXY12.K through TXY23.K.) which, in turn, go to the output multiplexors.

Complements of X/Y (CMPX/CMPY)

The complement of either the contents of X or the contents of Y is accomplished by gating the register output through inverting elements. See figure 2-69. These inverters are located between the register output gates and the function box output multiplexors. The inverter outputs are identified by CXY00.J through CXY11.J. and CXY12.K through CXY23.K..

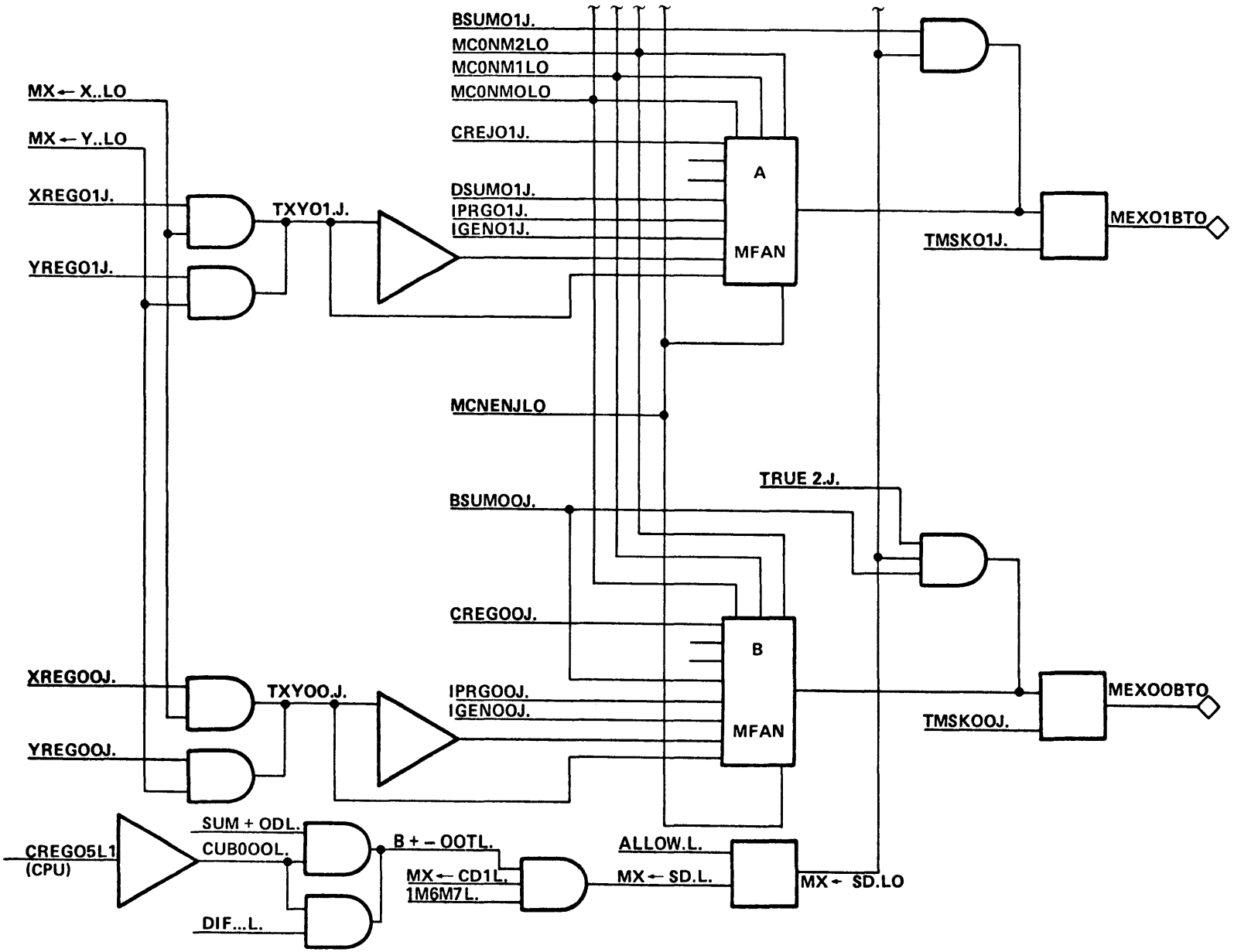


Figure 2-69. 24-Bit Function Box Output Logic

Mask of X/Y (MSKX/MSKY)

MSKX and MSKY are simply the number of bits of either register that are equal to the field length specified by CPL. Control for MSKX/MSKY is accomplished in the mask generator by causing the signal LIFTMKLO to go false, limiting the output of enabling signals for the function box output buffers to the value decoded from CPL. See the mask generator discussion for further details.

24-BIT OUTPUT MULTIPLEXOR

Except for the B-sum, all 24-bit results are multiplexed through 24 MFAN's on their way to the main exchange. The outputs of the MFAN's are then buffered onto MEX under the control of the mask bits. Figure 2-69 illustrates the first two MFAN's for bits 00 and 01 (MEX00BT0 and MEX01BT0). The 24-bit binary sum bypasses the multiplexors and is gated under the control of MX←SD..L. and ALLOW.L. to the output buffers. MX←SD..L. is true whenever the binary sum or difference is sourced. ALLOW.L. allows the registers onto main exchange at all times except when sending data or address to port adapter across the PA/PD interface cable.

Note that the B-sum of every first bit of a 4-bit BCD unit is equal to the D-sum, and is an input to the MFAN's. Table 2-3 lists the necessary control signals used to multiplex the various results to the main exchange.

Table 2-3. Results to Main Exchange

MCONM2LO	MCONM1LO	MCONM0LO	MCENnLO	MX←X.LO.	MX←Y.LO	Output
0	0	0	1	1	0	X-REG/MSKX
0	0	0	1	0	1	Y-REG/MSKY
0	0	0	1	1	1	XORY
0	0	1	1	1	0	CMPX
0	0	1	1	0	1	CMPY
0	1	0	1	X	X	XANY
0	1	1	1	X	X	XEOY
1	0	0	1	X	X	DSUM
1	0	1	1	X	X	MAXM (MEX10-12)
1	1	0	1	X	X	MAXS (MEX15-23)
1	1	1	1	X	X	CP-REG. (MEX0-7)

X = Not significant.

X/Y Output Controls

Whenever the X-register or the Y-register is sourced by a micro operator, the signals MX←X..LO or MX←Y..LO are set true to enable the input gates to the MFAN's. The two control signals are generated as follows (refer to figure 2-70):

Gate 1: true when the X-register is sourced (1C and 2C micros)
Gate 2: true when MSKX is sourced (1C and 2C micros)
Gate 3: true when XORY is sourced (1C and 2C micros)
Gate 4: true when CMPX is sourced (1C and 2C micros)
Gate 5: true when Y-register is sourced (1C and 2C micros)
Gate 6: true when MSKY is sourced (1C and 2C micros)
Gate 7: true when XORY is sourced (1C and 2C micros)
Gate 8: true when CMPY is sourced (1C and 2C micros
Buffer C: true when X-register is sourced (7C and 7E micros)
Buffer D: true when Y-register is sourced (7C and 7E micros)

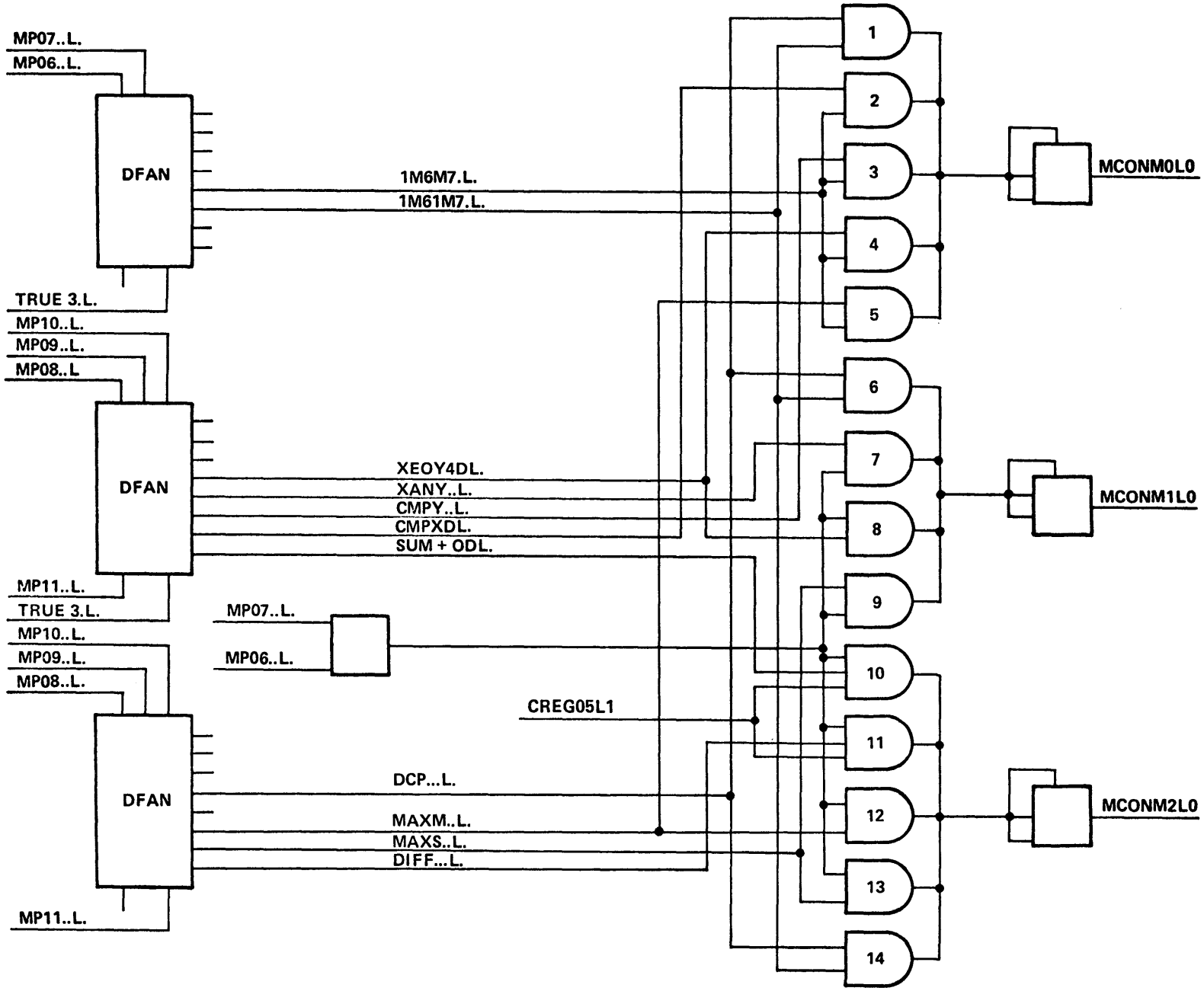


Figure 2-70. X/Y Registers to Main Exchange Controls

24-Bit Multiplexor Controls

In order to enable the 24 multiplexors, the two levels MCNENJLO and MCNENKLO are generated through a set of parallel buffers (A and B). Buffers C and D enable the MFAN's during the 7C micro when either the X- or Y-register is sourced. Gate 1 enables the MFAN's when any of the upper-column three registers is sourced during a 1C or 2C micro (BCD SUM/DIFF, CMPX, CMPY, XANY, XEOY, XORY MSKX, MSKY). Gate 2 enables the MFAN's for all the registers for which the mask is lifted when they are sourced in a 1C or 2C micro (X, Y, CP, MAXM, MAXS). Refer to figure 2-71.

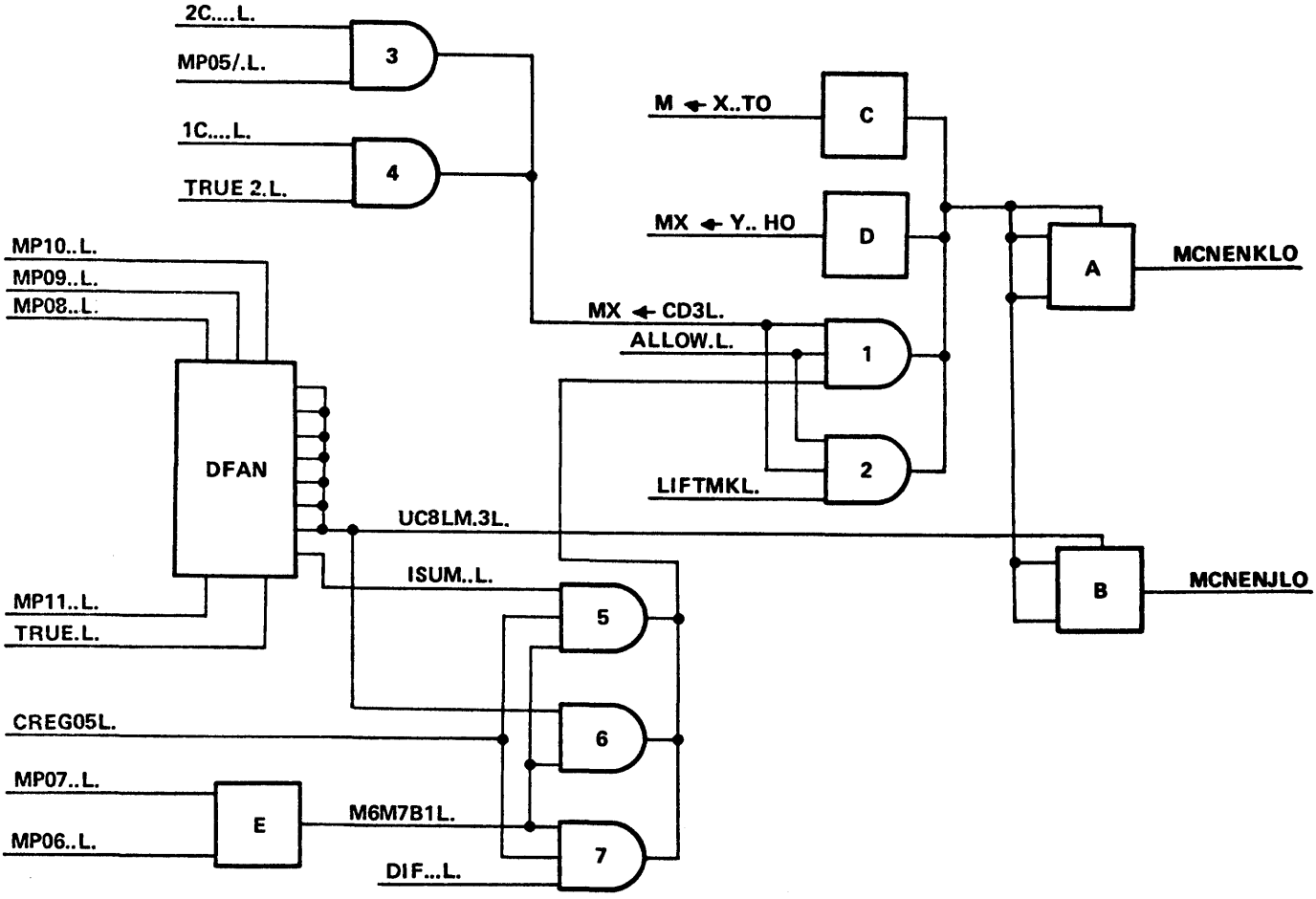


Figure 2-71. Output Multiplexor Enable Logic

The three address lines which control the 24 MFAN's are generated according to the desired output. Table 2-3 illustrates the selected address lines for the different outputs. Refer to figure 2-72.

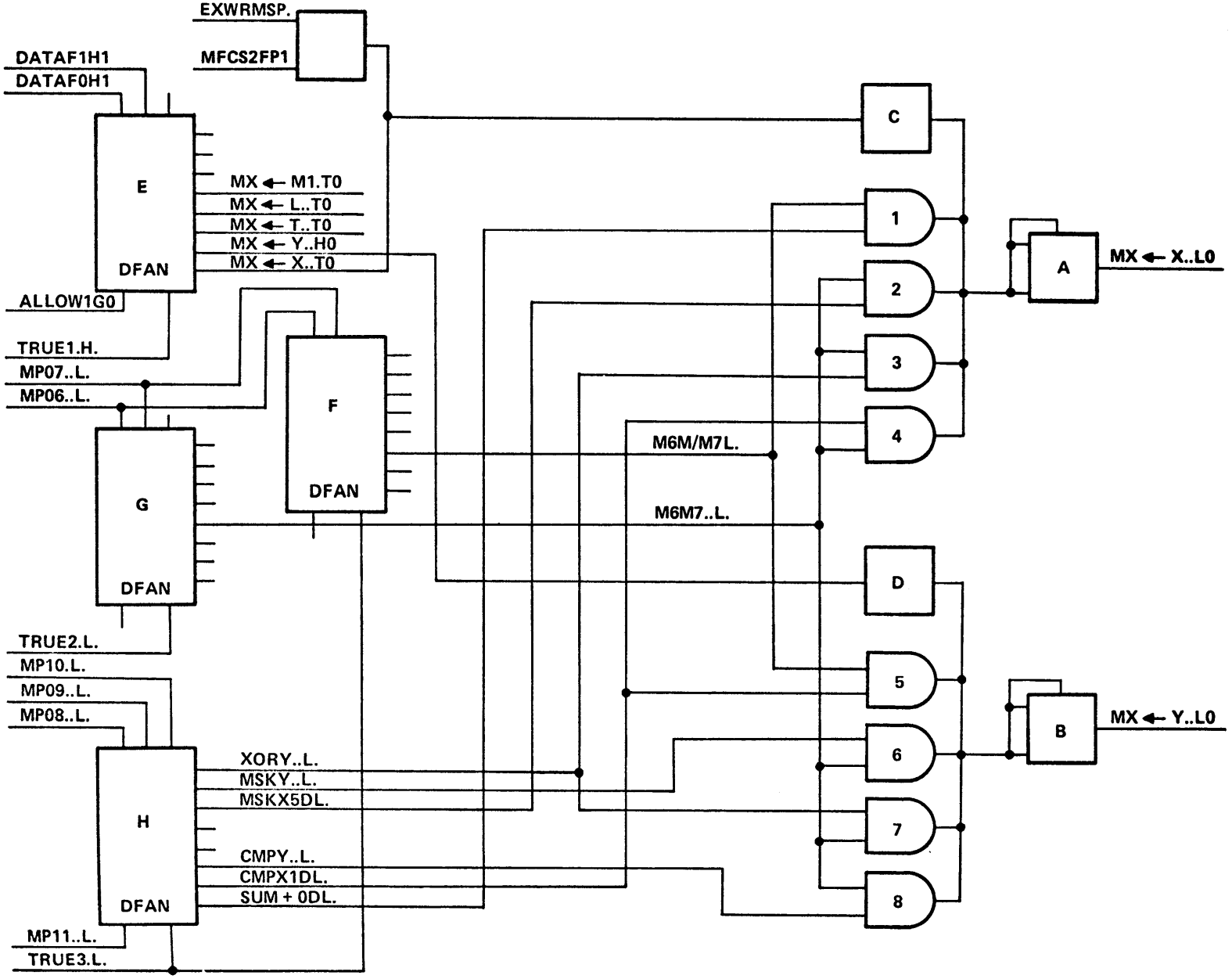


Figure 2-72. Output Multiplexor Control Lines

4-BIT OUTPUT MULTIPLEXORS

The 4-bit results from the 24-bit function box are multiplexed onto the 4-bit auxiliary bus if sourced by a micro operator. Refer to figure 2-73. Table 2-4 lists the necessary control signals used to multiplex these results to the 4-bit auxiliary bus.

Table 2-4. Multiplexing Control Signals

4SR11/F.	4SOR09F.	4SOR08F.	4 ← C1M8NO	OUTPUT
0	0	0	1	XYCN
0	0	1	1	XYST
1	1	0	1	BICN
0	1	1	1	*CPU
*Sourced for 4-bit functions				

If any of the 4-bit results are sourced, the signal SLOW4.F (through MFAN E) becomes true, indicating a slow source. This resets F1NISHP1 in the micro timing for one clock period, extending a regular one-clock micro to a 2-clock micro.

4-Bit Multiplexor Controls

The multiplexor enable signal 4 ← C1M8NO which points to select column 1, middle 8 registers is generated by two different sets of gates. Refer to figure 2-74. Gates 1, 2, 4, 6, 7 and, 8 control the MFAN's when the processor is in the run mode, and gates 3, 5, and 9 take over this control during the console state (halt-state). The address lines for the MFAN's 4SR11/NO, 4SOR09NO, and 4SOR08NO are also produced by two sets of gates. Gates 10, 12, and 14 address the MFAN's in run mode while gates 11, 13 and, 15 supply the address during the console state. For the address line assignment refer to table 2-4.

Note that unlike the multiplexor control signals for the 24-bit results, the control signals for the 4-bit sources are decoded directly from the M-register outputs under the control of the XPOS flip-flops. This is done in order to allow the availability of 4-bit results at the earliest possible time.

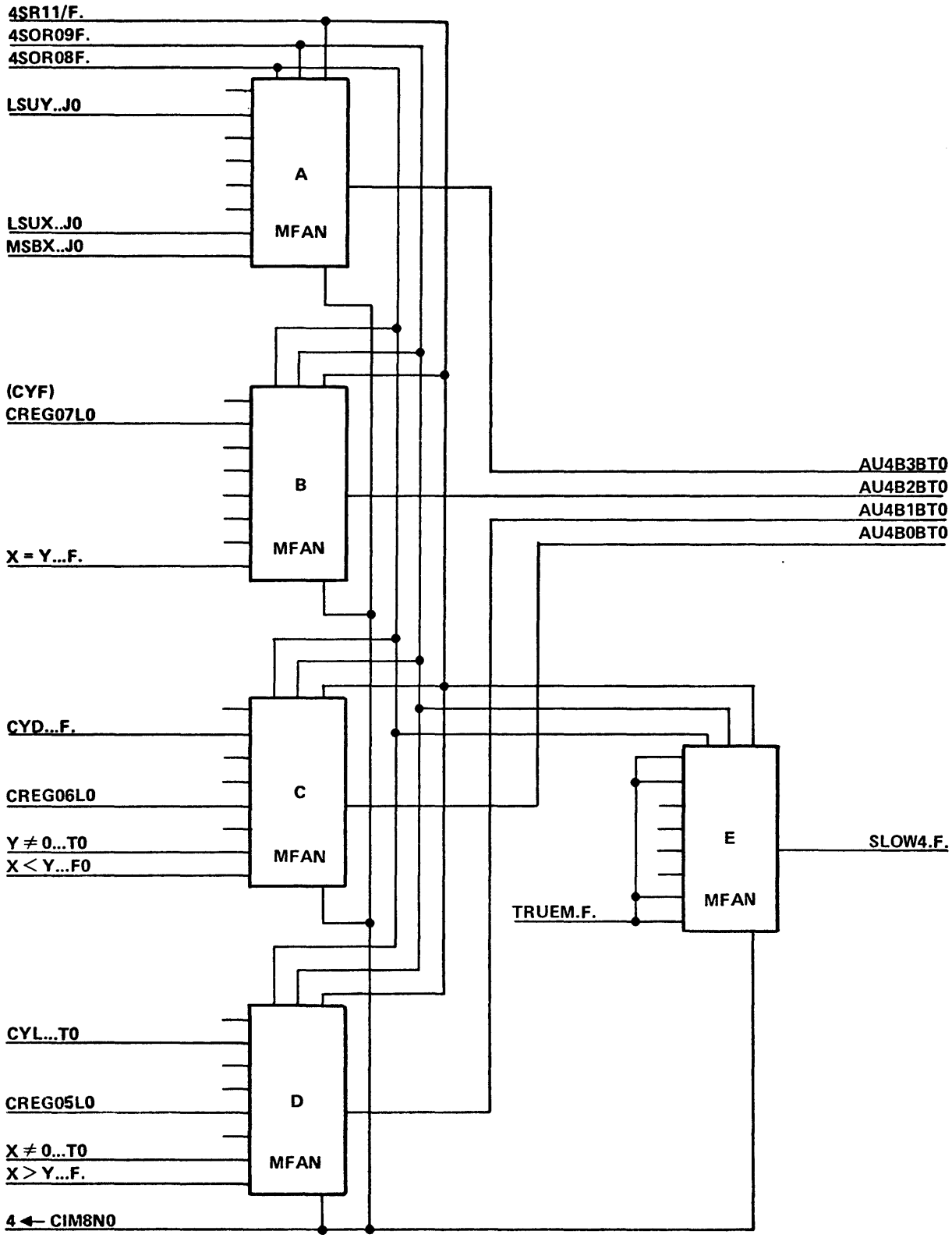


Figure 2-73. 24-Bit Box Output Multiplexors for 4-Bit Results

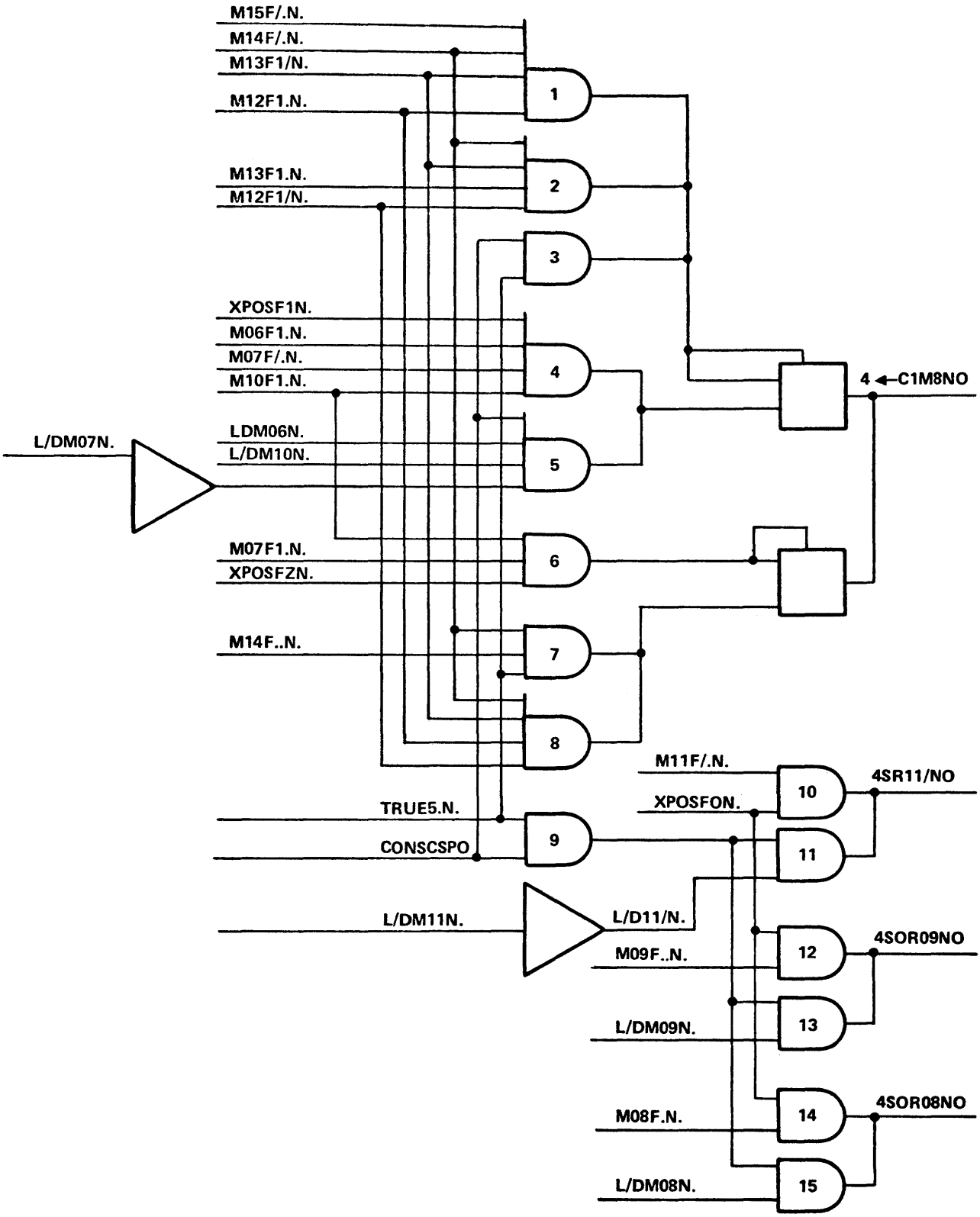


Figure 2-74. The 4-Bit Result Source Control

4-BIT FUNCTION BOX

The 4-bit function box (figure 2-75) provides logical and arithmetical functions involving the contents of the 4-bit processor registers and pseudo registers. Unlike the 24-bit function box, however, only one operand input comes from a register, and this source may be designated by the executing micro operator. Where used, the second operand input is contained within the executing micro. The 4-bit function box may be exercised by the micros in the 1C through 6C series. The functions thus provided include pass through, set, AND, OR, Exclusive OR, increment, increment and test, decrement, decrement and test, test and branch, and test and skip. (These are more fully described in the micro functions discussion. Refer to that section if further details are desired.) The result output of the 4-bit function box (modified or unmodified) is always returned by way of the 4-bit result bus to the source register. This is true except for pseudo registers, which may not serve as a sink, and register-to-register moves which do not produce a "result" as such.

In the pass-through function mentioned above, the 4-bit function box serves as a path from the 4-bit auxiliary bus to either the 4-bit result bus or four least-significant bits of the MEX. Since all 4-bit register outputs are gated to the 4-bit auxiliary bus, this bypass to the register input data paths is necessary to allow register-to-register moves.

INPUT MULTIPLEXORS

For all practical purposes, the operand input to the 4-bit function box is the contents of the 4-bit auxiliary bus. Therefore, the output multiplexors (which the T, L, FB, C, TOPM, XYST, XYCN, BICN, FLCN, and INCN registers share) may also be considered the function box input multiplexors. These multiplexors comprise three groups, serving the processor registers as follows:

Group 1: C, T, and L registers

Group 2: FB register

Group 3: FLCN, INCN and TOPM registers

A portion of each group of multiplexors is illustrated in figures 2-76, 2-77, and 2-78, respectively. In function, register selection for gating to the 4-bit auxiliary bus is controlled by the bit configuration on the 4-bit source control lines. The combination required for selecting any given register is shown in table 2-5.

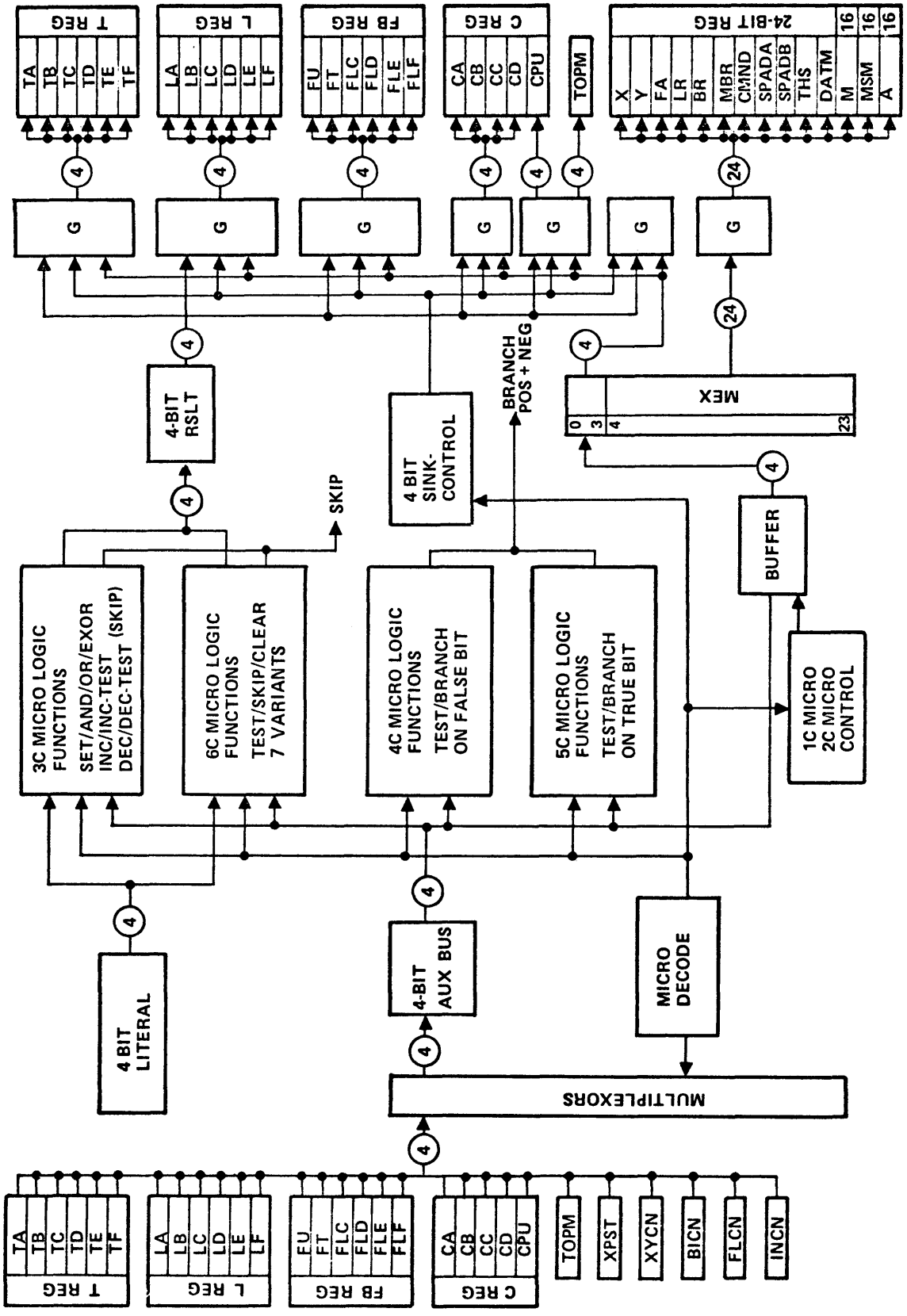


Figure 2-75. The 4-Bit Function Box Block Diagram

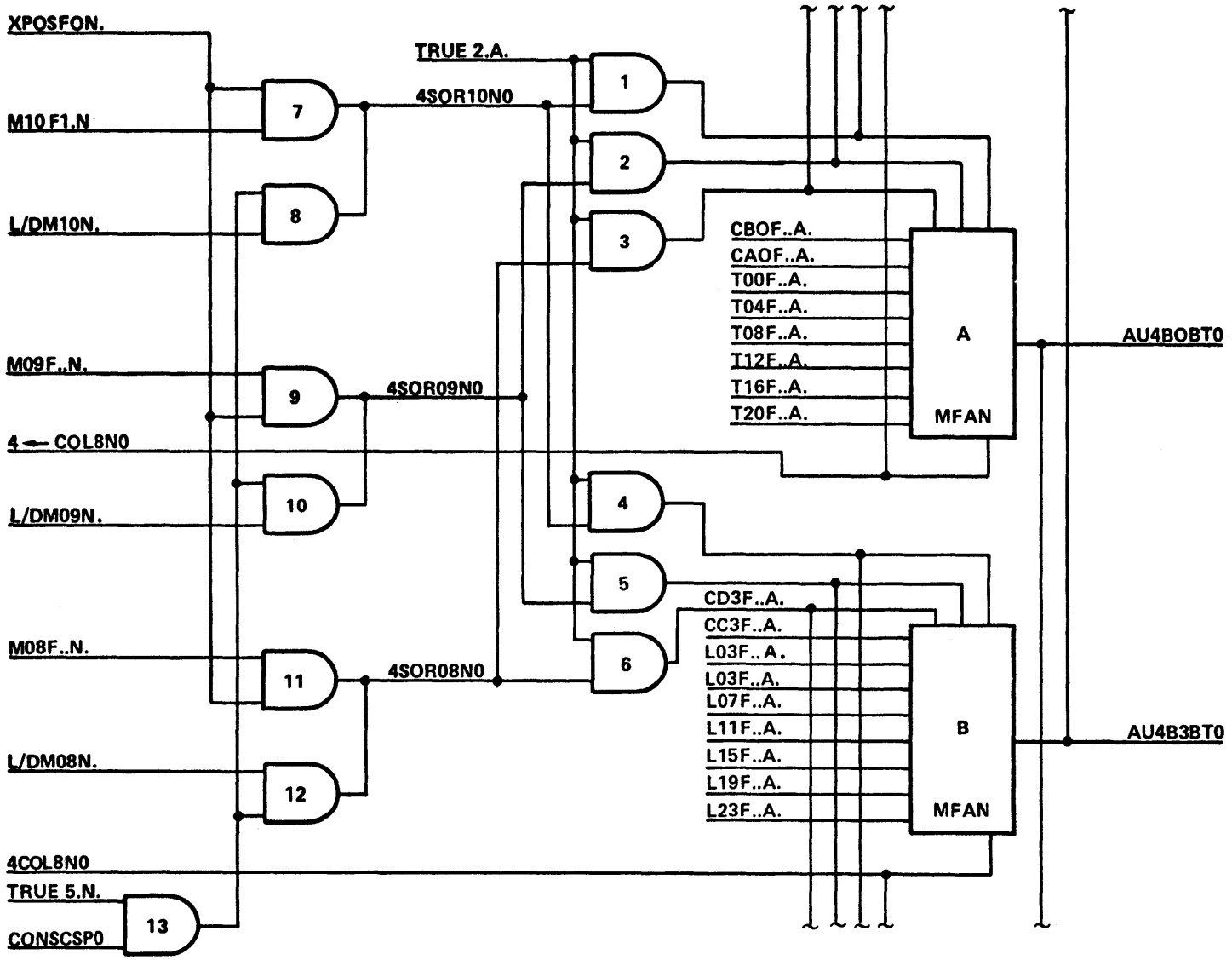


Figure 2-76. 4-Bit Box Input Multiplexors (C/T/L Registers)

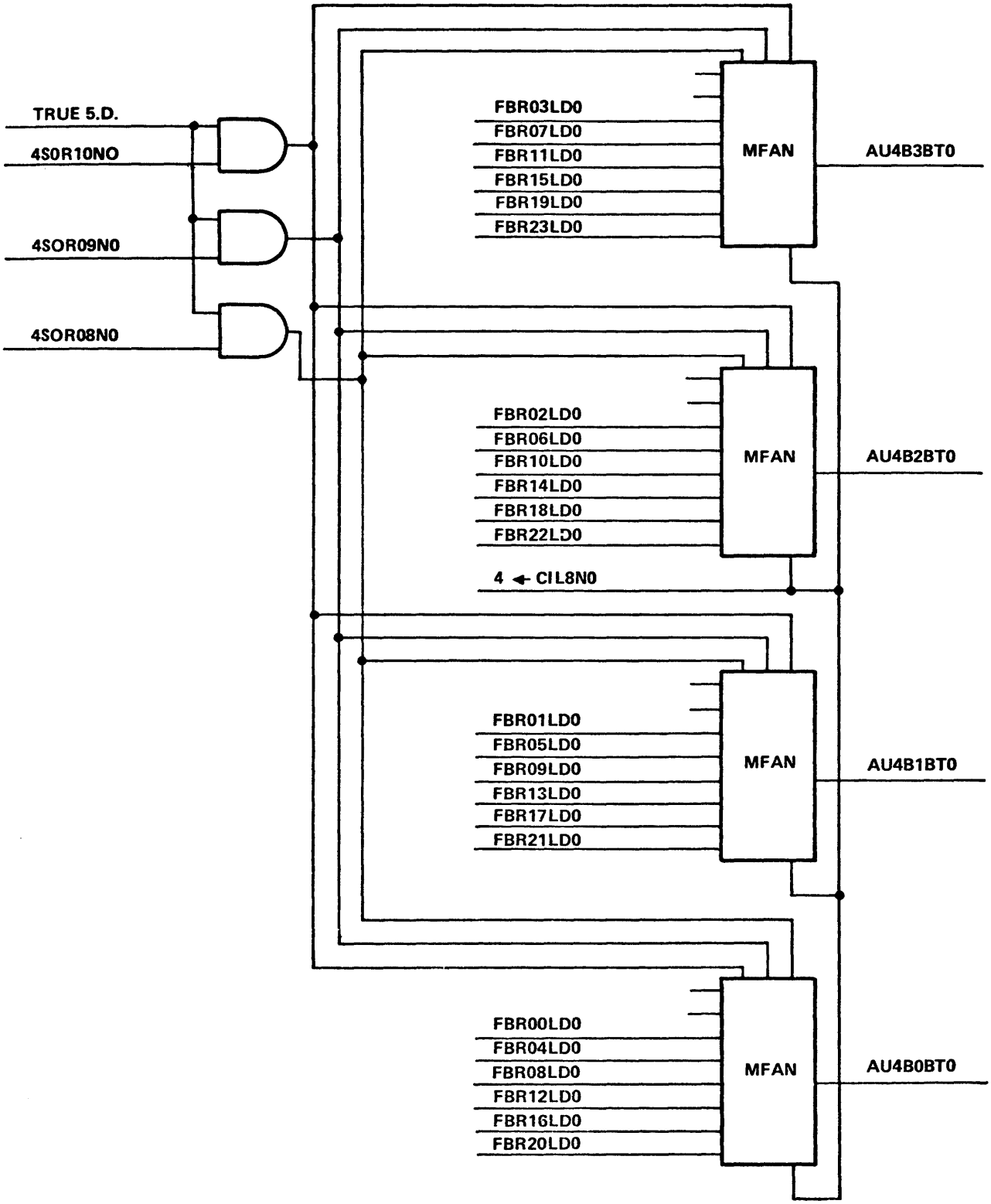


Figure 2-77. 4-Bit Box Input Multiplexors (FB Register)

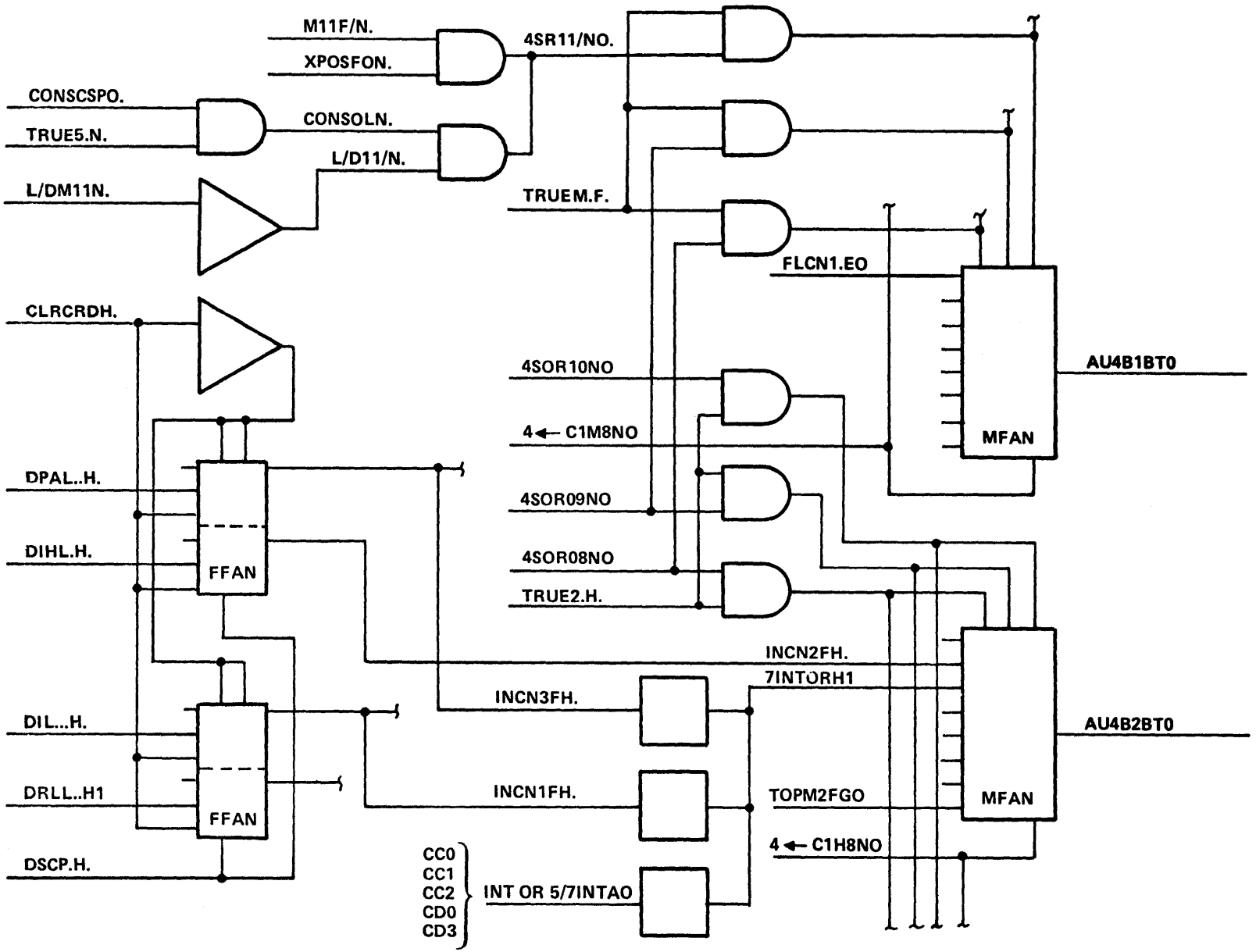


Figure 2-78. 4-Bit Box Input Multiplexors (FLCN/INCN/INT/TOPM Registers)

Table 2-5. The 4-Bit Function Box Enabling Signals

4-BIT MFAN ADDRESS LINES			4-BIT MFAN ENABLE LINES					
4SOR10N0	4SOR09N0	4SOR08N0				4←COL8N0	4←COH8N0	4←CIL8N0	4←CIM8N0	4←CIH8N0	4SR11/N0	INPUTS
0	0	0				1						TA
0	0	1				1						TB
0	1	0				1						TC
0	1	1				1						TD
1	0	0				1						TE
1	0	1				1						TF
1	1	0				1						CA
1	1	1				1						CB
0	0	0					1					LA
0	0	1					1					LB
0	1	0					1					LC
0	1	1					1					LD
1	0	0					1					LE
1	0	1					1					LF
1	1	0					1					CC
1	1	1					1					CD
0	0	0						1				FU
0	0	1						1				FT
0	1	0						1				FLC
0	1	1						1				FLD
1	0	0						1				FLE
1	0	1						1				FLF
1	0	1	0	1	0							BICN
1	0	1	1	1	0							FLCN
0	0	0	0	0	1							TOPM
0	0	0	0	1	0							XYCN
0	0	0	1	1	0							XYST
0	1	1	0	0	1							INCN
0	0	1	1	1	0							CPU
0	1	0	1	0	1							INT*

*THE INTERRUPT BIT IS MULTIPLEXED AT THE SAME TIME AS XYST

Multiplexor Controls

The 4-bit source control lines (actually address and enable lines for the MFANs) are derived directly from the contents of the M-register, under the control of the XPOS flip-flop. Refer to figure 2-79. Such direct control is employed in order to make the contents of the 4-bit registers available on the 4-bit auxiliary bus at the earliest time following initiation of the micro. In the console state, the M-register outputs are replaced by the load/display lines, which are derived from the settings of the register group and REGISTER select switches on the console.

4-BIT MICRO OPERATOR CONTROLS

The micros which exercise the 4-bit function box are decoded by the logic shown in figure 2-80. DFAN A develops a control signal for each of the micros in the 1C through 6C series (these signals are known as 4B0X1CF through 4B0X6CF). Additional control signals are developed as follows:

Gate 1: R ← 3C+−F. is decoded from the 3C micro when MOP line 06 is true. This signal enables gating the 4-bit sum or difference to the 4-bit result bus.

Gate 2: S ← 3C+−F. is decoded from the 3C micro when MOP lines 04 and 06 are both true. Enables the skip function when an overflow or underflow occurs as a result of the arithmetical operation.

Gate 3: 3CT-M.F. is decoded from MOP lines 05 and 06, and defines the add or subtract mode for the 3C as well as the 6C micro.

Gates 8-11: Pass the contents of MOP lines 00 through 03 to the function box as the literal operand input.

Gates 4, 12, and 13: 4X ← 4BBF. gates the contents of the 4-bit auxiliary bus to the MEX. This creates a direct path through the 4-bit function box for register-to-register moves (1C and 2C micros).

Gates 14 and 15: 4R ← 36CF. enables 8-bit function box output multiplexors to transfer results from a 3C or 6C micro to the 4-bit result bus.

MICROS

The above signals are employed in various combinations by the different micro operators. Since the micro functions are quite different, each is described individually.

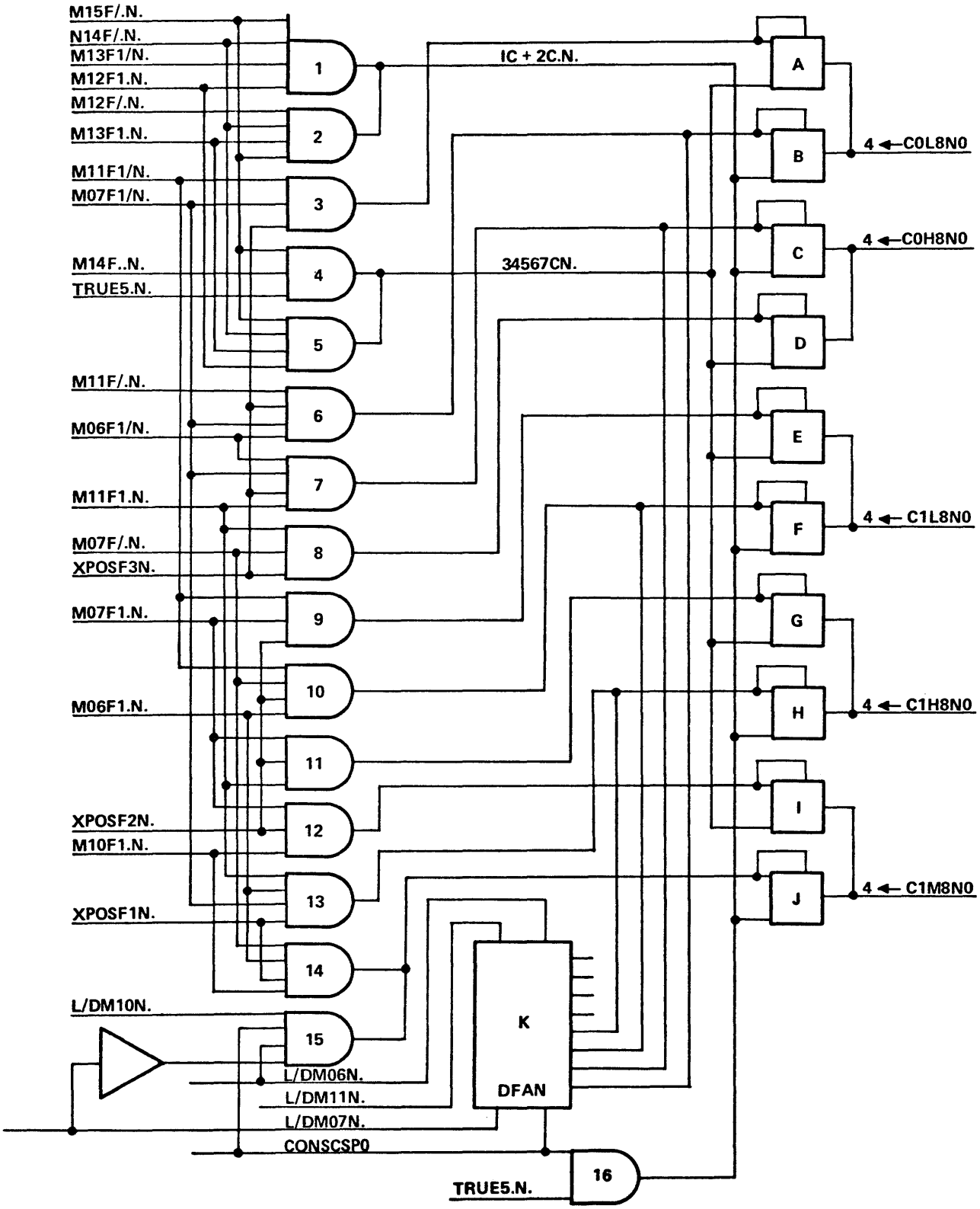


Figure 2-79. 4-Bit Input Multiplexor Control Logic

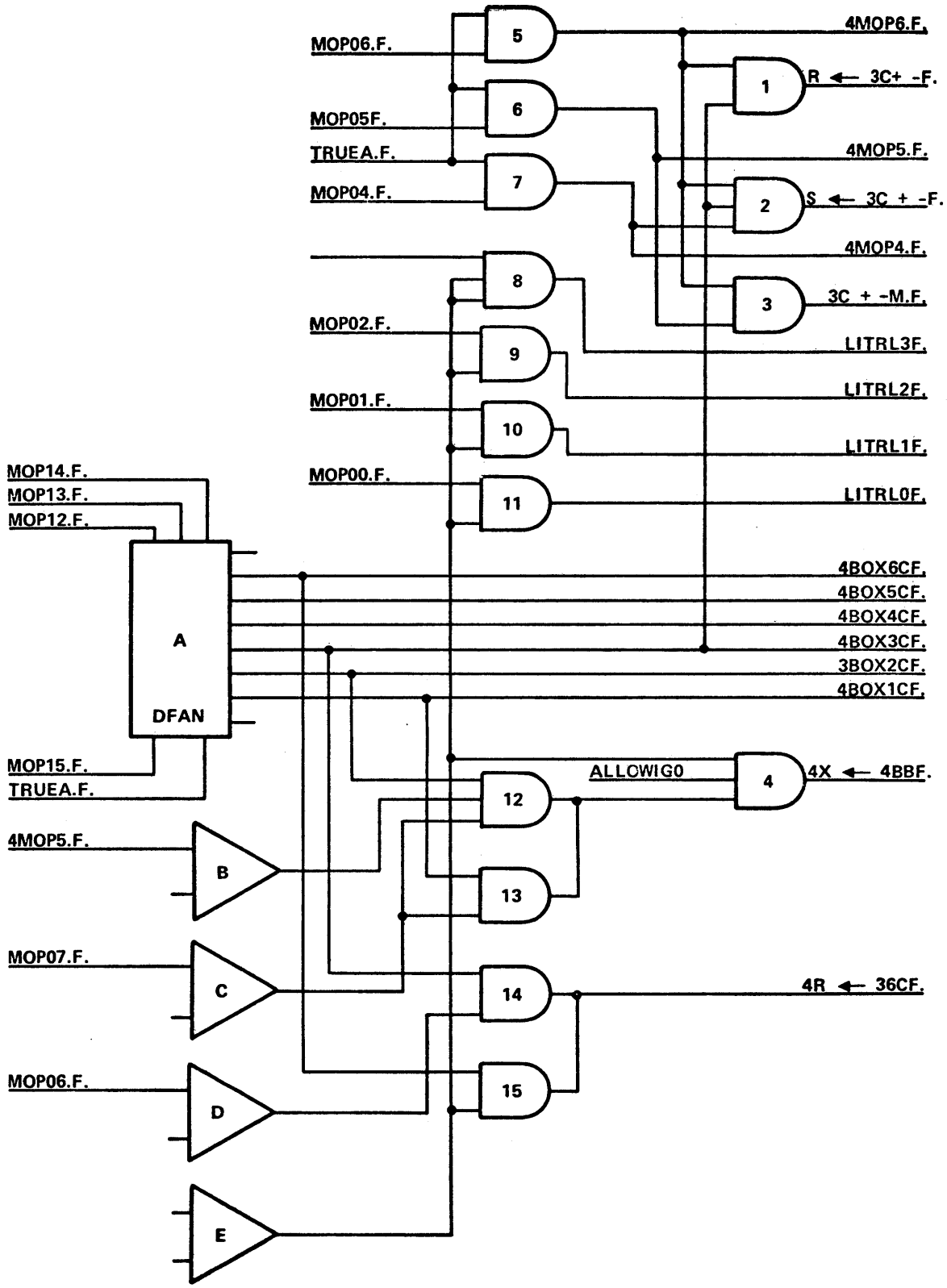


Figure 2-80. 4-Bit Manipulate Micro Control Logic

3C Micro Functions

The 3C micro performs various arithmetical and combinatorial functions between the selected 4-bit source register and the literal as specified by the micro variants. Refer to figures 2-81 and 2-82. The result is returned to the same register, except if the sourced 4-bit register is a pseudo register. Table 2-6 illustrates the different variants and the pertinent control signals.

Table 2-6. Variants and Control Signals

Variant	MOP 6	MOP 5	MOP 4	R ← 3C + - F.	S ← 3C + - F.	3C + - M.F.
SET	0	0	0	0	0	0
AND	0	0	1	0	0	0
OR	0	1	0	0	0	0
EOR	0	1	1	0	0	0
INC	1	0	0	1	0	0
INC-T	1	0	1	1	1	0
DEC	1	1	0	1	0	1
DEC-T	1	1	1	1	1	1

SET Variant

Set the 4-bit register to the value of the literal and gate the result to the 4-bit result bus (4 RSLT-bus). The signal 4R← 36CF. enables MFANs I, J, K, and L.

AND Variant

The 4-bit register and the literal are ANDed through DFANs A, B, C, and D. The output 3GENn.F. represents the result and is put through MFANs I, J, K, and L to the 4-bit result bus (4 RSLT bus). MOP-04 addresses the MFANs.

OR Variant

The 4-bit register and the literal are ORed at the outputs of gates 1-8, and the result is gated to the 4 RSLT bus. MOP-05 addresses the MFANs that select the 3CORn.F. inputs.

EOR Variant

The 4-bit register and the literal are exclusive ORed through DFANs A, B, C, and D. The outputs BPROPnF. represent the result and are put onto the 4-bit result bus (4 RSLT bus) via MFANs I, J, K, and L. MOP-04 and MOP-05 address the MFANs.

INC Variant

The 4-bit register and the literal are binarily added by using two sets of DFANs (A and H) and a CFBN. The sum represents the result and is the output of DFANs E, F, G, and H going directly onto the 4-bit result bus (4RSLT bus). R← 3C + - F. enables the DFANs.

INC and TEST (INC-T) Variant

An INC function is performed and the overflow (carry) is monitored. If the CFBN generates the output 3CO/USF., a skip function is initiated. S← 3C + - F. enables gate 1.

DEC Variant

The literal is binarily subtracted from the 4-bit register by using DFANs A through H and the CFBN. The operation of DFANs A through D is governed by the subtract term 3C + - M.F. The difference represents the result and is the output of DFANs E through H going directly onto the 4-bit result bus (4 RSLT bus). R ← 3C + - F. enables the DFANs.

DEC and TEST (DEC-T) Variant

A DEC function is performed and the underflow (borrow) is monitored. If the CFBN generates the output 3CO/USF., a skip function is initiated. S ← 3C + - F. enables gate 1.

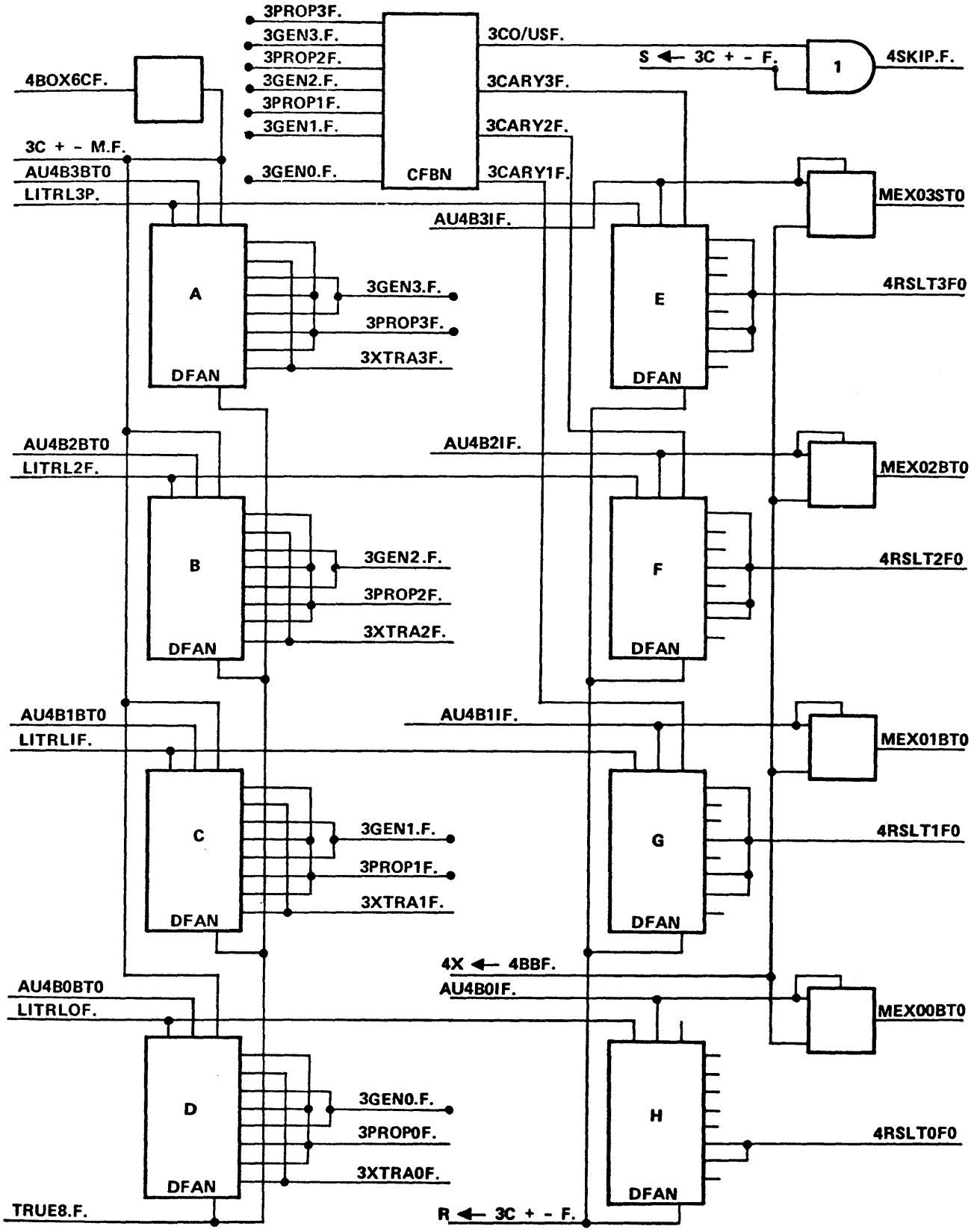


Figure 2-81. The 3C Micro Function Adder and Skip Logic

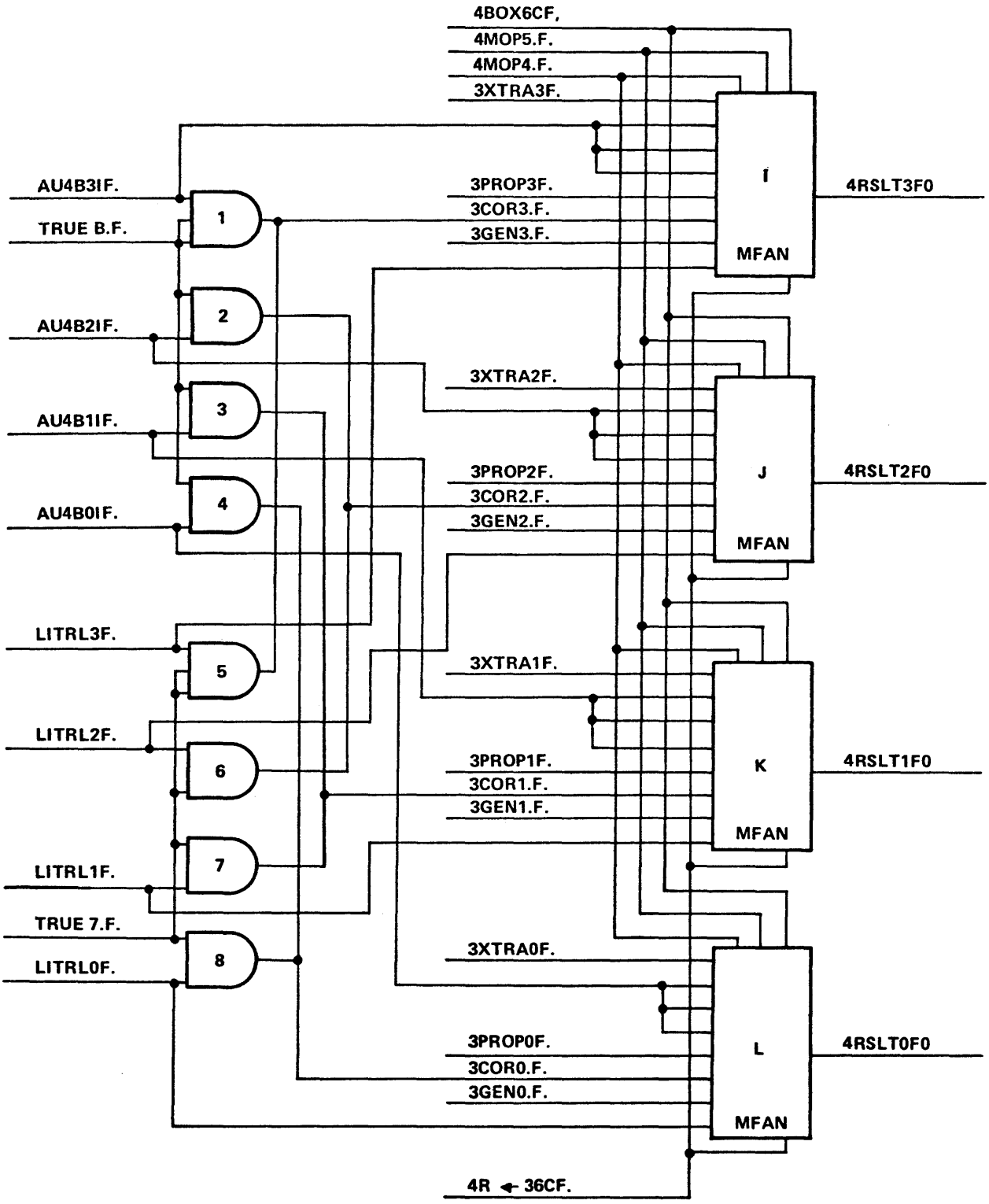


Figure 2-82. The 4-Bit Box Output Multiplexors to the RSLT Bus

6C Micro Functions

The 6C micro performs various bit test functions as specified by the micro's variants. The literal is used as a mask. Only the bits in the 4-bit register which are referenced by the 1 bits in the mask are tested and also cleared if specified. Refer to figure 3-83. All other bits are ignored, except if a compare is made equal to mask. The 4-bit register, modified or unmodified, is gated to the 4-bit result bus (4 RSLT bus) and back to the sourced register, except if the sourced 4-bit register is a pseudo register. Table 2-7 illustrates the various variants of the test function.

Table 2-7. 6C Micro Variants

MOP 6	MOP 5	MOP 4	Test Function
0	0	0	Skip if any referenced bit is true
0	0	1	Skip if all referenced bits are true
0	1	0	Skip if register equal to mask
0	1	1	Skip and clear if all referenced bits are true
1	0	0	No skip if any referenced bit is true*
1	0	1	No skip if all referenced bits are true*
1	1	0	No skip if register equal to mask*
1	1	1	No skip and clear if all referenced bits are true*

* The skip is taken if the test condition is not met.

Figure 2-83 illustrates the bit test logic for the 6C micro. Whenever any of the gates 1, 2, 3, or 4 have a true output, a skip function is initiated. At the same time, the 4-bit register bits (referenced by the literal mask) are being tested. They are also gated back to the RSLT bus either 1) directly (via the multiplexors(or 2) if a clear variant is selected, they are put through the first stage of the 3C adder logic to perform the clear function and then sent (via multiplexors) to the RSLT bus. Referring back to figure 2-84, the MFAN addresses are additionally governed by the signal 4BOX6CF. which multiplexes the fourth, fifth, and sixth input (used for the non-clearing 6C variant functions) to the RSLT bus. If a clear variant is selected, the MFAN's are addressed for the seventh input 3XTRAnF. which is derived from the 3C adder logic. Referring back to figure 2-83, DFANs A through D are controlled by the signal 4BOX6CF. Only bits referenced by a 1 mask are cleared, i.e., whenever a register bit and a mask bit are true simultaneously, the output 3XTRA3F. is false.

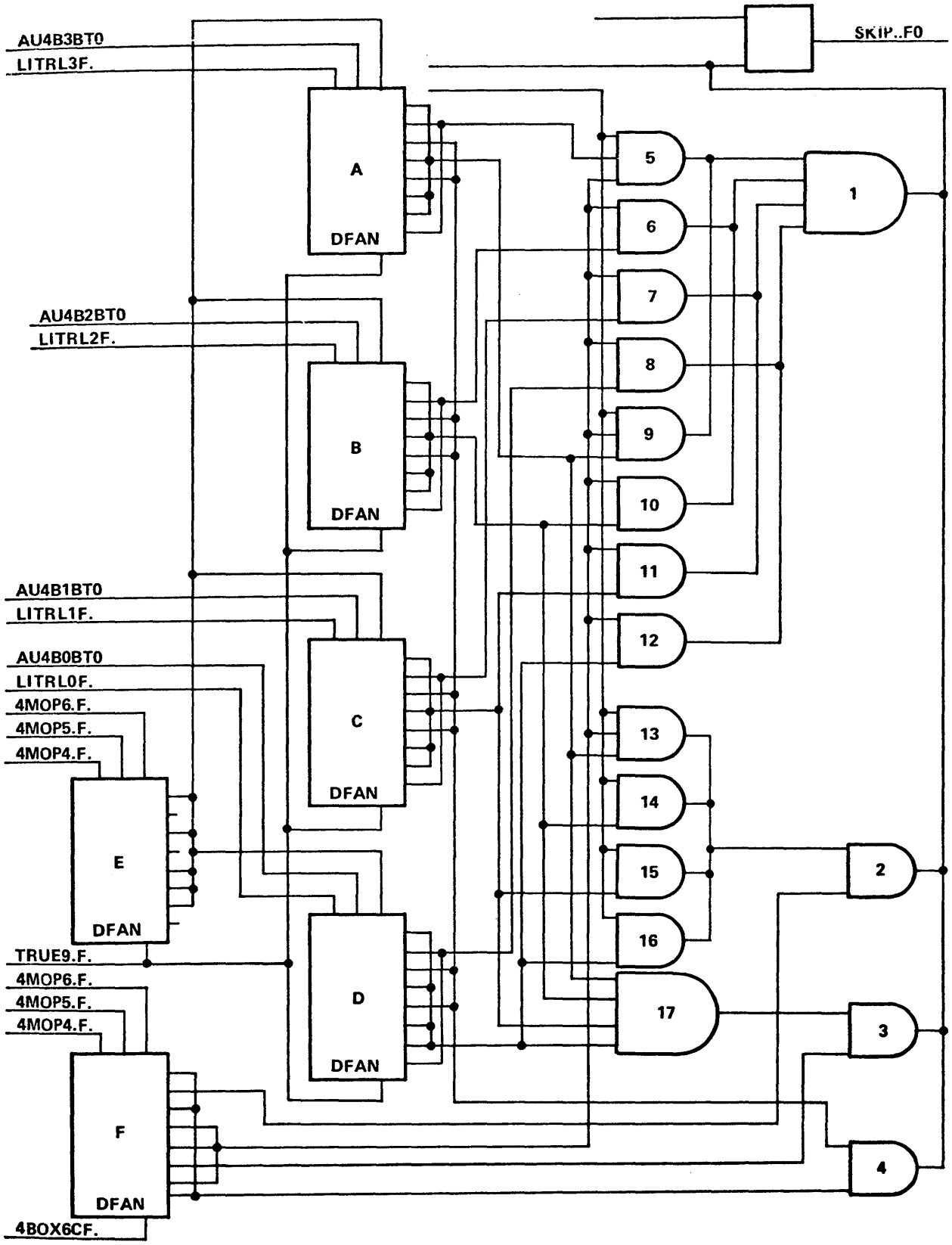


Figure 2-83. The 6C Micro Function (Skip) Logic

4C/5C Micro Functions

The 4C and 5C micros test a specified bit of the selected 4-bit register, and, depending upon the result, can branch (relative branch) a specified displacement to the next micro instruction. Refer to figure 2-84. The branch is taken if the bit is 0 for a 4C micro or 1 for a 5C micro. If the test is not met, the next in-line micro is executed.

Referring to figure 2-84, a DFAN decodes the specified bit from MOP lines 05 and 06. The selected bit is then gated with the appropriate 4-bit register bit from the auxiliary 4-bit bus. Testing for a false bit during the 4C Micro, and for a true bit during the 5C Micro generates the signal BRANCHFO which, in turn, develops JUMP.FO..

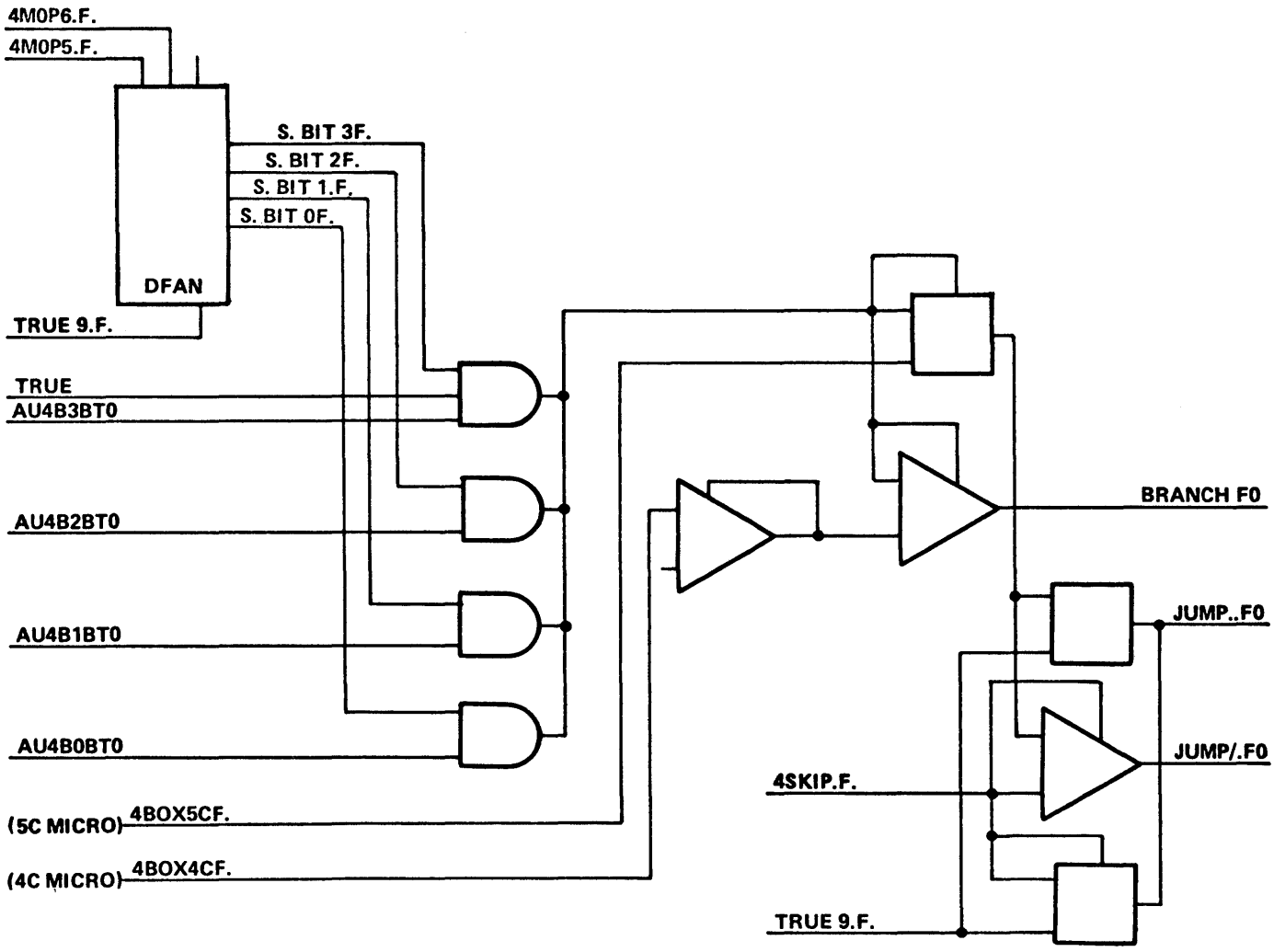


Figure 2-84. 4C/5C Micro Function Branch Logic

4-BIT REGISTER SINK CONTROLS

Except for the 4-bit register move operations performed within a 1C or 2C micro, the 4-bit register used as a sink is always the same as the one used as the source. Refer to figure 2-85. This is true except when the source is a pseudo register, in which case only the test function is relevant, and the information on the 4-bit auxiliary bus is lost.

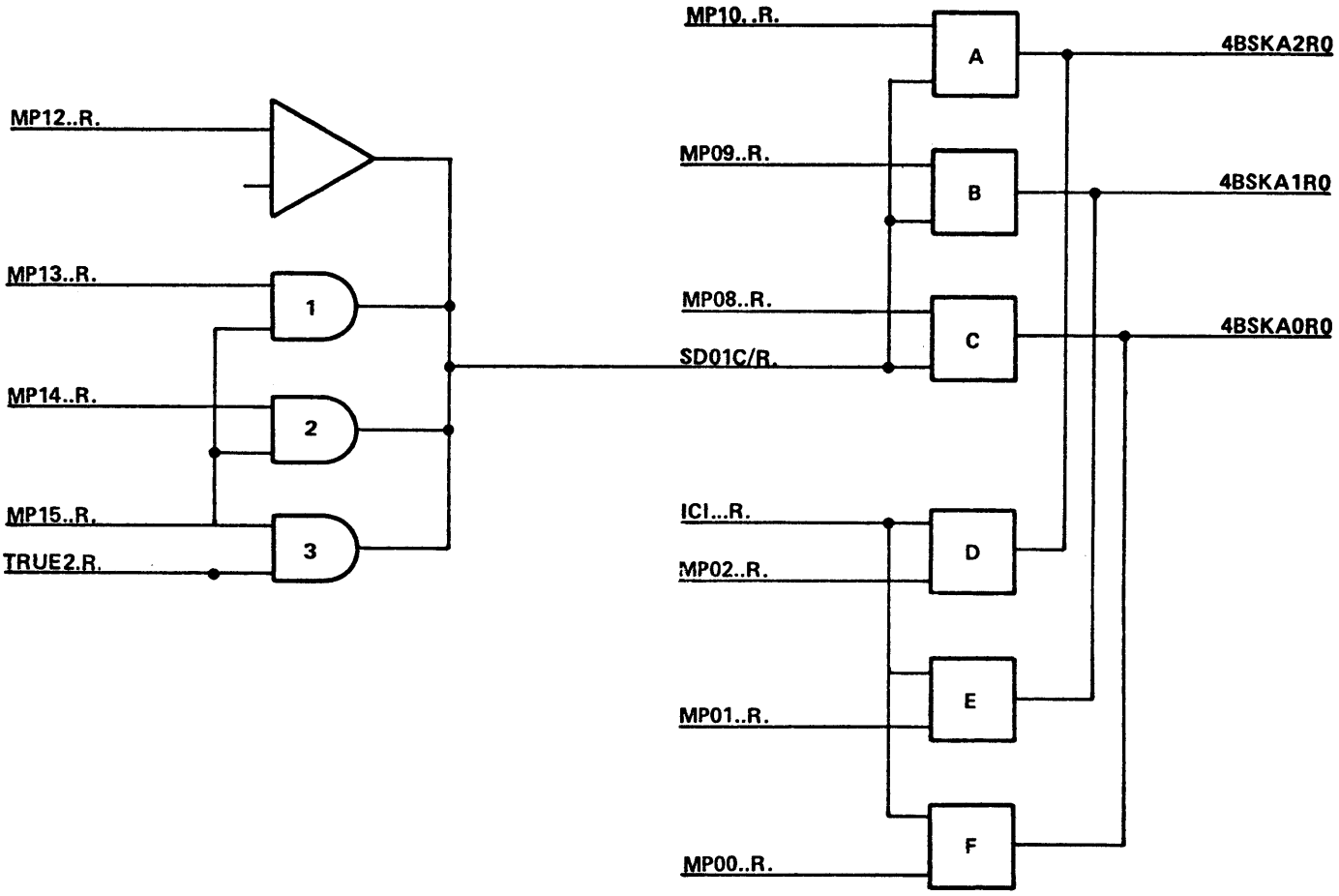


Figure 2-85. 4-Bit Register Sink Controls

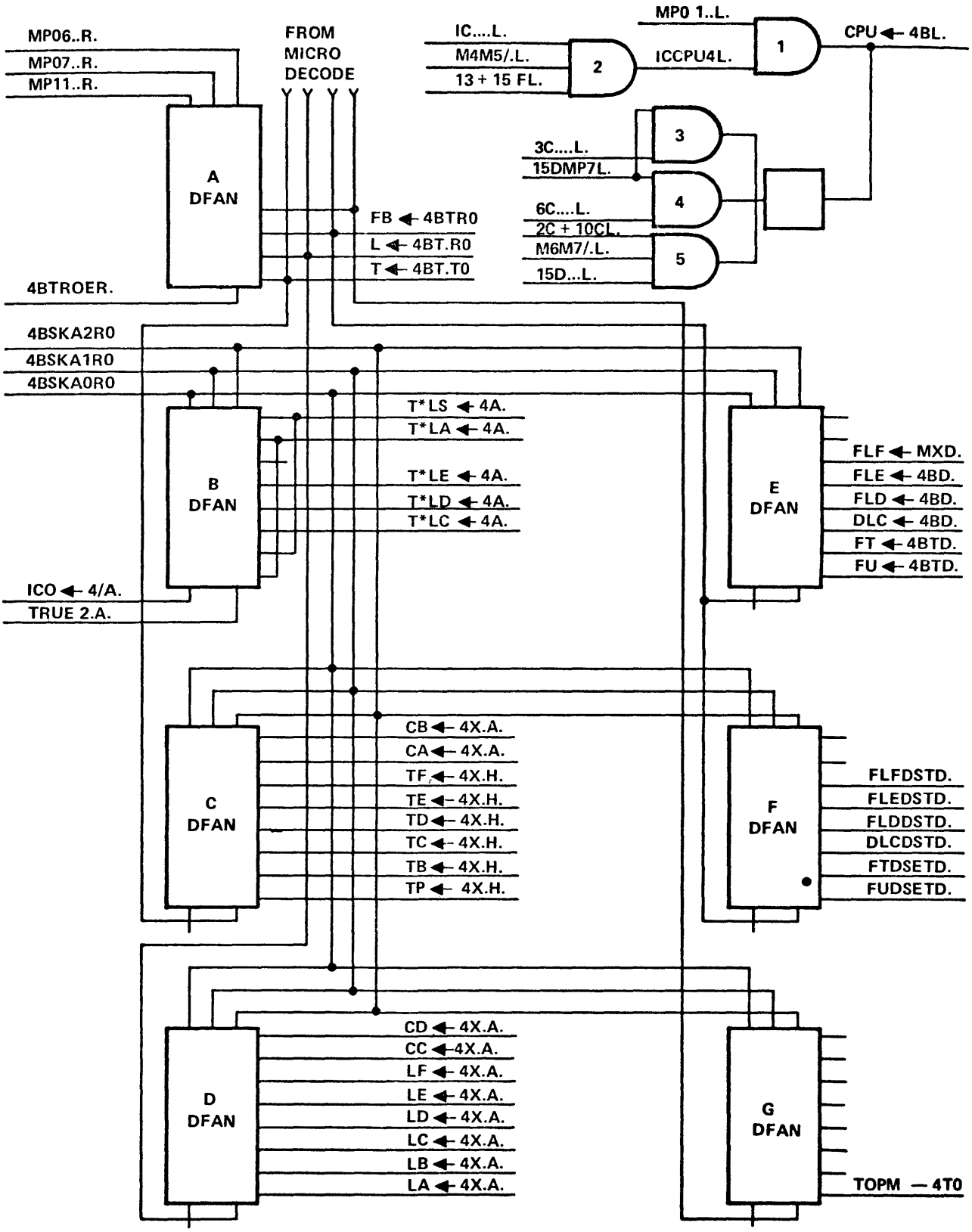


Figure 2-86. 4-Bit Register Sink Control Decoding

The three 4-bit sink-control signals (4BSKAORO through 4BSKA2RO) are generated from the MOP lines. Referring to figure 2-85, buffers A, B, and C provide the controls for the 2C, 3C, 4C, 5C, and 6C micros, while buffers D, E, and F provide controls for the 1C micro. Using these three sink signals, the logic shown in figure 2-86 generates (via DFAN's B through G) the various enable lines for the 4-bit register to be sinked. DFAN-A plus additional micro decode logic (not shown) provides the enable lines for the DFAN's. If the CPU register is being used as a sink, gates 1 through 5 provide the enable signal as follows:

Gates 1 and 2: True if CPU is used as a sink by the 1C micro.

Gate 3: True if CPU is used as a sink by the 3C micro.

Gate 4: True if CPU is used as a sink by the 6C micro.

Gate 5: True if CPU is used as a sink by the 2C or 10C micro.

4-BIT RESULT GATING

A 4-bit result is basically gated to all the different 4-bit registers in a similar manner. Gates 1 through 4 are the result drivers. Gates 5 through 12 are the inputs from the main exchange for the TE and TF or LE and LF registers. They are kept inactive by the signals 3CO ← 4/A and 3 + 6C/.FO which are false during a 4-bit result transfer. However, during a 1C or a 2C micro (if a 4-bit register is used as a sink), gates 9 through 12 are enabled and the four least-significant MEX bits are transferred to TF or LF, respectively. Gates 13 through 24 are the sink gates for the 4-bit result controlled by the appropriate enable lines. Figure 2-87 shows the 4-bit sink gating controls, and figure 2-88 illustrates the gating of the result to the TD, FE, and TF registers or the LD, LE, and LF registers.

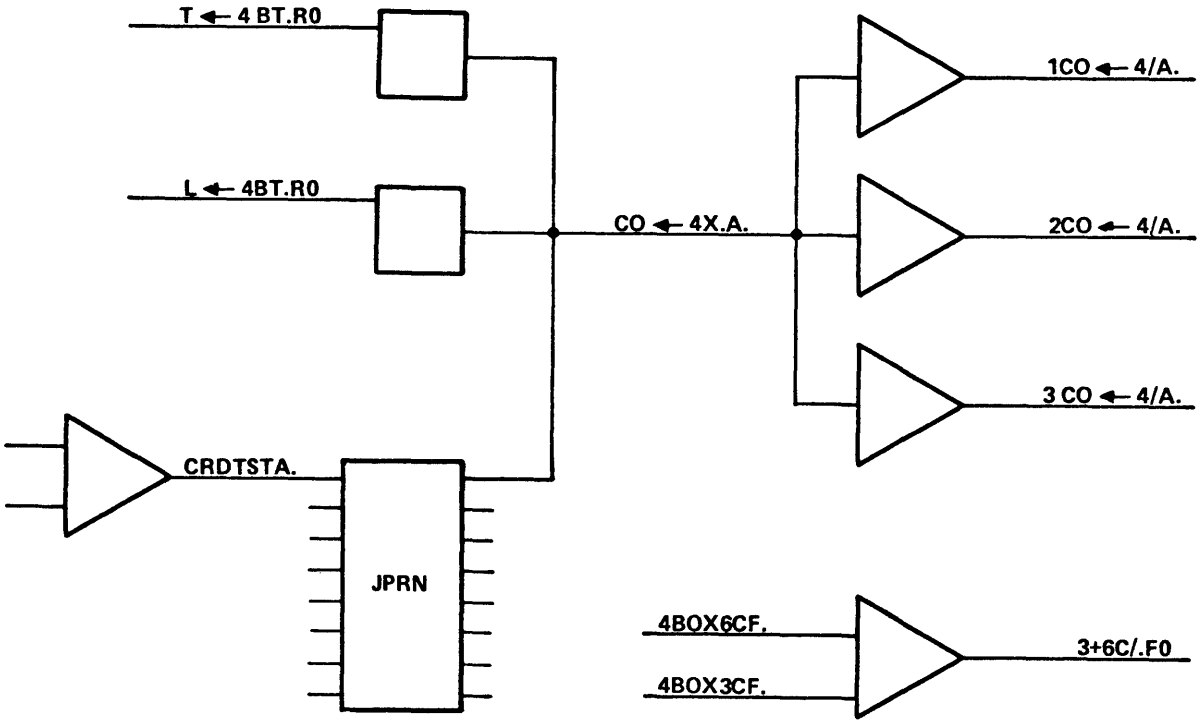


Figure 2-87. 4-Bit Sink Gating Controls

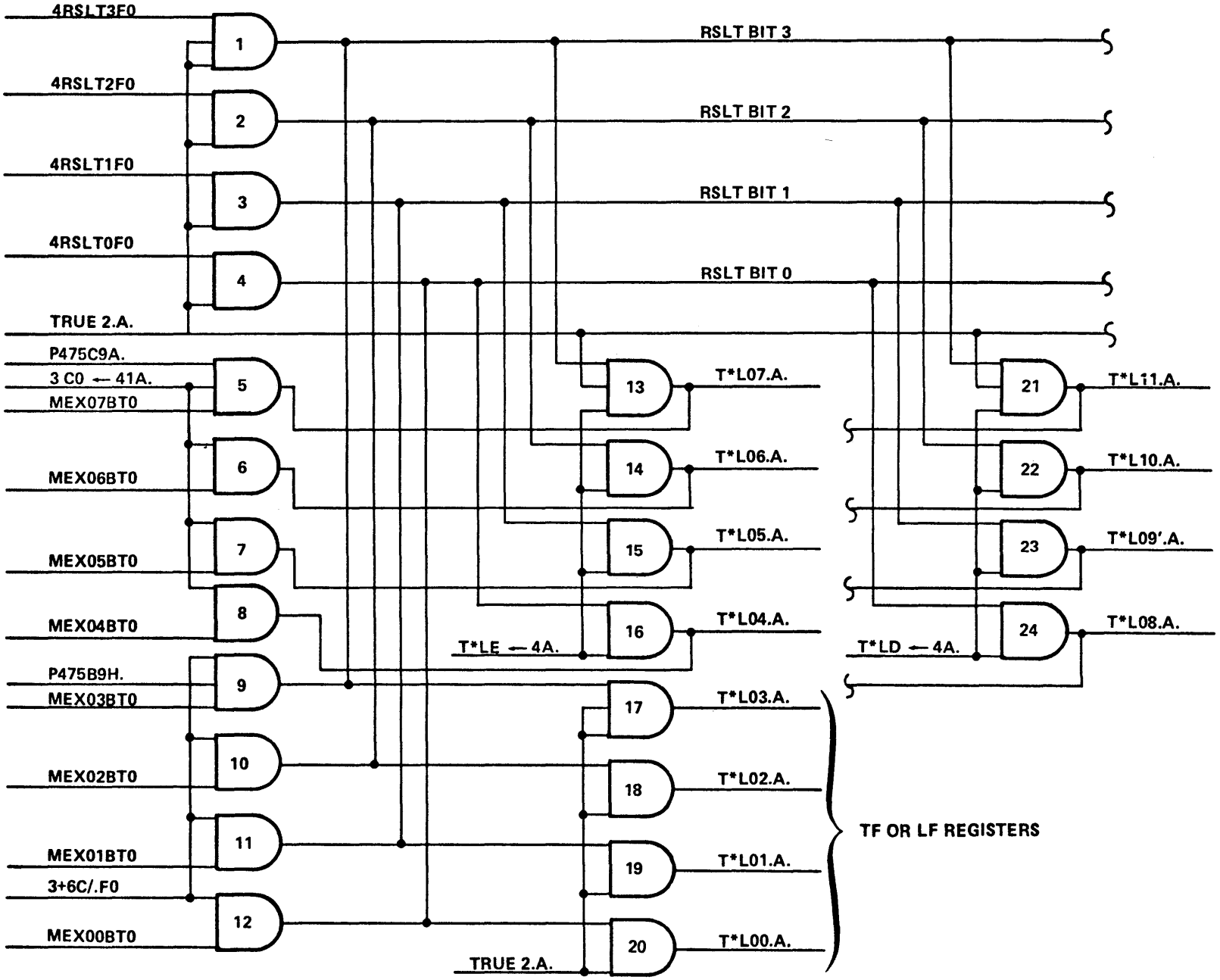


Figure 2-88. 4-Bit Sink Gating Controls

M-STRING MEMORY

M-string memory is a 16-bit-wide modular memory used exclusively for the storage of micro operators. MSM is addressed by the contents of the A-register, and is arranged to gate the addressed word location directly to the M-register for execution. MSM may also be addressed as a source or sink, with access to or from the MEX. This allows micro strings to be moved into and out of MSM as required. The M-string memory is much faster than S-memory, contributing to the greater speed of the M-memory processor.

MSM DATA STORAGE

M-string memory is composed of storage cards, each capable of storing 2K bytes, or 1024 16-bit words. The storage cards are made up of an array of 64 256 x 1-bit RFDN random access memory chips (figure 2-89). The 64 chips are placed in four rows of 16 chips each. These rows are known as groups A, B, C, and D. To achieve the 16-bit width of MSM, one entire group of 16 chips is enabled at a time. The bit locations within each of the 16 chips are addressed sequentially until exhausted, after which the process is repeated in the next successive group. Likewise, the memory cards are enabled in succession to create an electrically-continuous memory. Up to four memory cards may be employed for an overall storage capacity of 4096 16-bit words.

MSM ADDRESSING

To achieve the addressing scheme described above, a number of internal control signals are required. These signals are listed below, along with their derivations and significance.

Signal(s)	Derivation	Significance
ADR0 through ADR7	A-register bits 00 through 07	Select bit locations 00 through 255 within RFDN memory chips
SLCT00 through SLCT03	A-register bits 08 and 09	Select chip groups A, B, C, and D
MSCA0 and MSCA1	A-register bits 10 and 11	Select memory cards

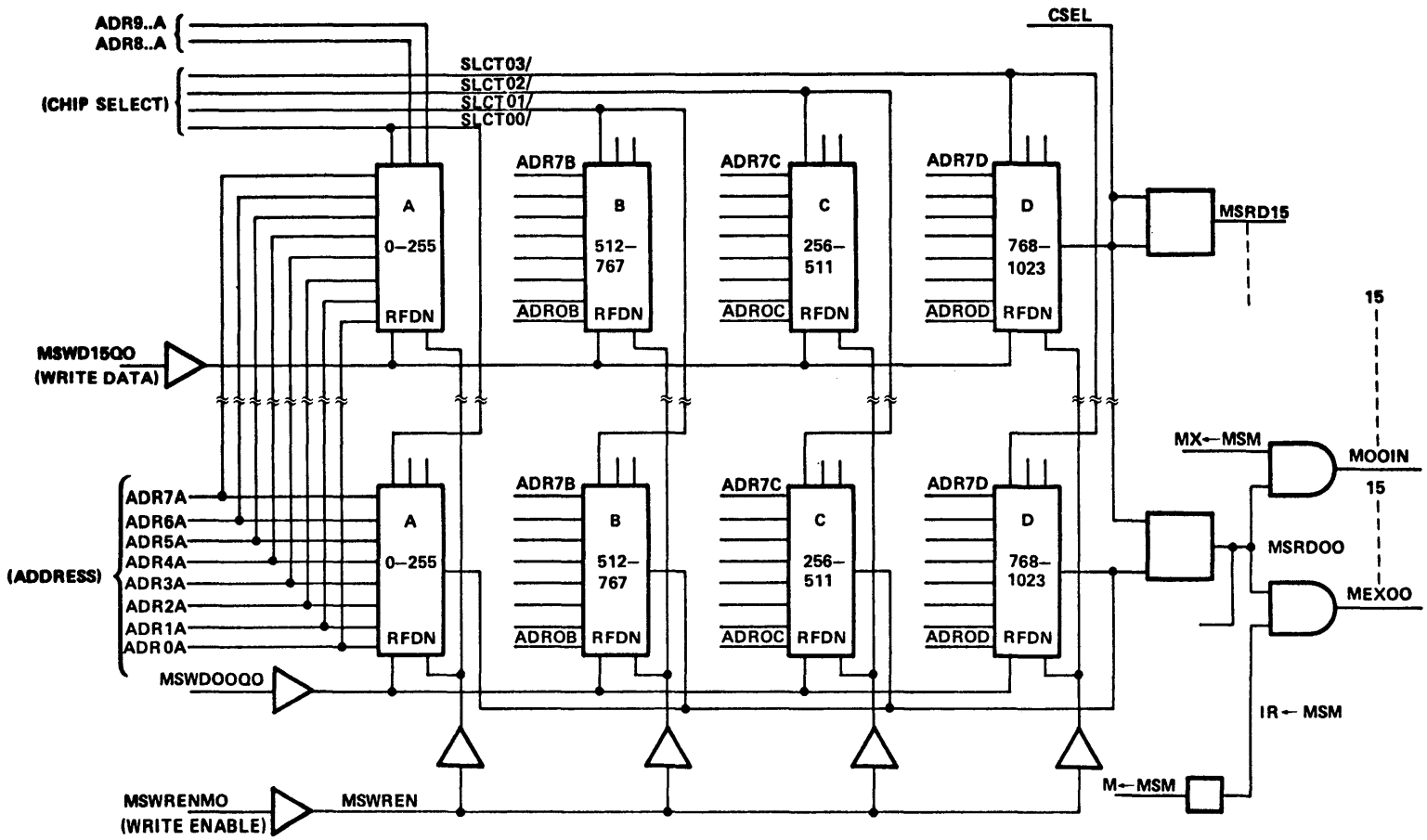


Figure 2-89. M-String Memory Card Logic Block Diagram

All chip groups receive the eight least-significant address lines (ADR0 through ADR7) in parallel. Since only one chip group is enabled at any given time, receipt of the same address by the other goups is of no significance. Likewise, all four groups on a card have data output lines connected in parallel. These output data lines are identified as DATA00 through DATA15. Each 1K storage card has output gating which enables the data output when the level CSEL is true. The data outputs of all storage cards are ORed at the backplane.

The control lines SLCT00/, SLCT01/, SLCT02/, and SLCT03/ are derived from A08F/ and A09F/, by the circuit shown in figure 2-90. The signal development and significance are as shown in table 2-8.

Table 2-8. MSM Addressing

Word Number	A09F	A08F	MSR11/	MSR10/	SLCT ##/ SEL01/	SEL00/	= False	Group
0 through 255	0	0	1	1	1	1	00	A
256 through 511	0	1	1	0	0	1	02	C
512 through 767	1	0	0	1	1	0	01	B
768 through 1023	1	1	0	0	0	0	03	D

In like manner, card selection is derived from A10F and A11F. These signals are used within each card to produce the level CSEL, which enables the gates (A and B in figure 2-90) controlling data output to the backplane. Within the card, the A10F and A11F flip-flop set and reset output lines are known as MSCA0, MSCA0/, MSCA1, and MSCA1/. In order that only one storage card be enabled for each of the four possible binary configurations of A10F and A11F, the backplane connections for these signals are different at each card location. This is shown in table 2-9.

Table 2-9. MSM Card Addressing

Card Number	Word Number	Address Bit States A11F	A10F	Backplane Connections Gate A	Gate B
Card 1	0 through 1023	0	0	MSCA1	MSCA0
Card 2	1024 through 2047	0	1	MSCA1	MSCA0/
Card 3	2048 through 3071	1	0	MSCA1/	MSCA0
Card 4	3072 through 4095	1	1	MSCA1/	MSCA0/

Selection of an individual card occurs when both its MSCA# inputs are false. Note that CSEL/, when true, forces all SLCT lines within a card true, effectively disabling all memory chips on it (in addition to disabling the output gating). This is significant during write operations, because the incoming write data is available simultaneously at all cards and chip groups.

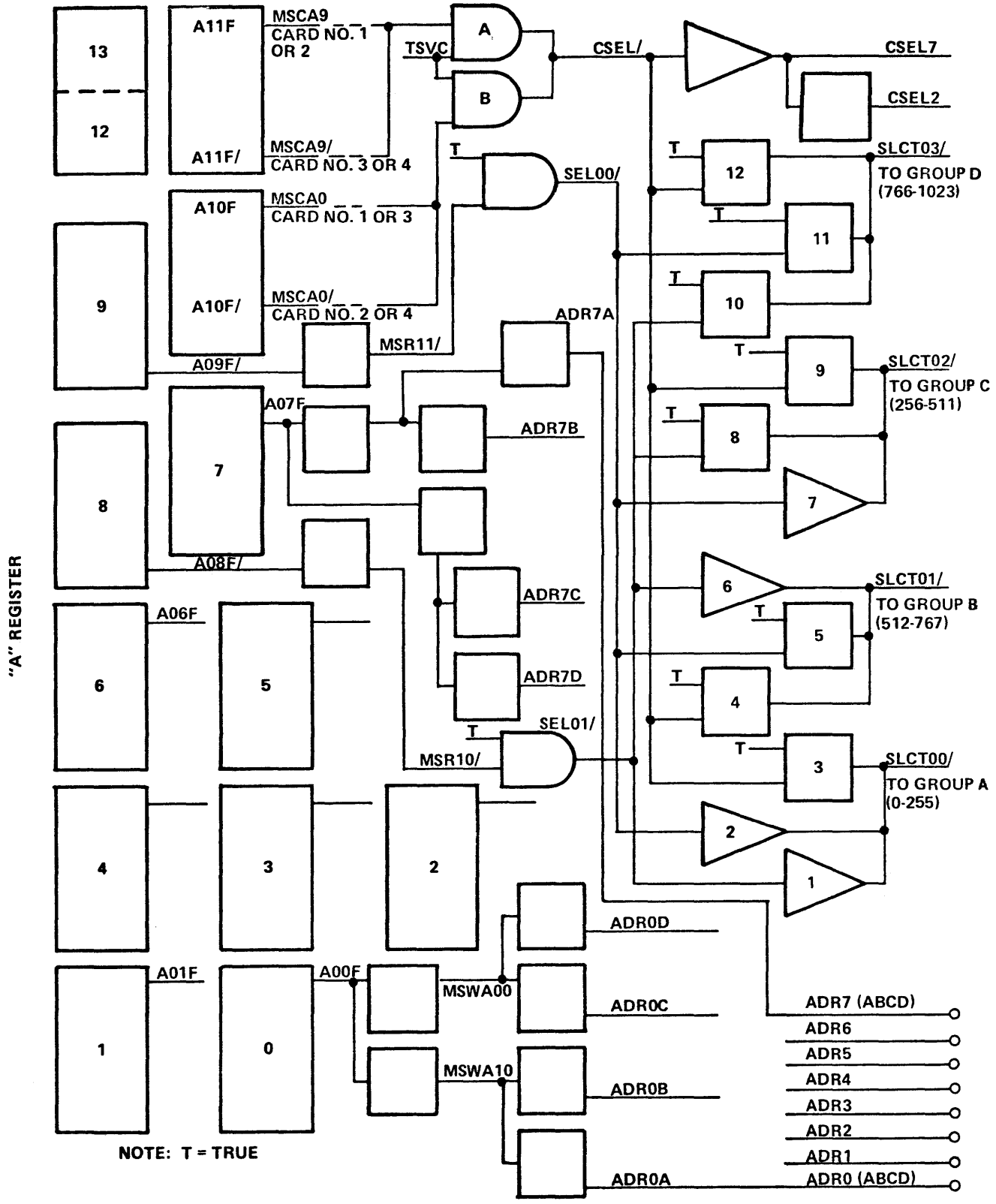


Figure 2-90. The M-String Memory Addressing Logic

MSM DATA INPUT

MSM has one data input source: the MEX. The MEX input is gated directly to all portions of MSM, with the write data not subject to further gating once past the input. Write data may enter MSM under three conditions: during an overlay M-string operation (2F micro), during a write MSM operation (7E micro), or during execution of any other micro which selects MSM as a sink (1C, 2C, 8C, 9C, or 10C). The enabled data is known as MSWD00Q0 through MSWD150O. The MSM data input gating is shown in figure 2-91.

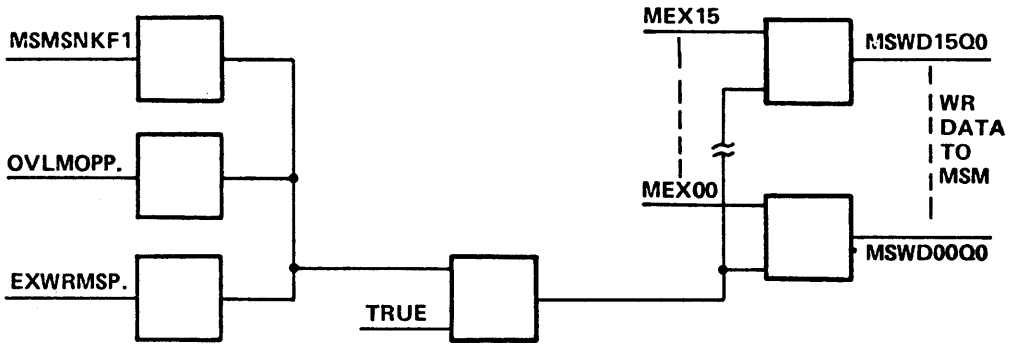


Figure 2-91. MSM Data Input Gating

WRITE OPERATION

Writing in M-string is a 2-clock pulse operation. Any of the conditions mentioned above generate the level WRITMSPO. This level causes, by way of the logic shown in figure 2-92, the generation of an 80 nanosecond false pulse, called MSWPENMO, during the second clock pulse. The latter pulse is distributed to all units of MSM as M-string write enable.

READ DATA

Read data from MSM may be gated to either the M-register or the MEX, with M being the usual destination. When the currently executing micro operator selects MSM as a source and a register other than M as a sink, the level OR ← MSM enables gating the read data to the MEX. Micro fetching from MSM occurs when M ← MSM enables gating the read data to the M-register.

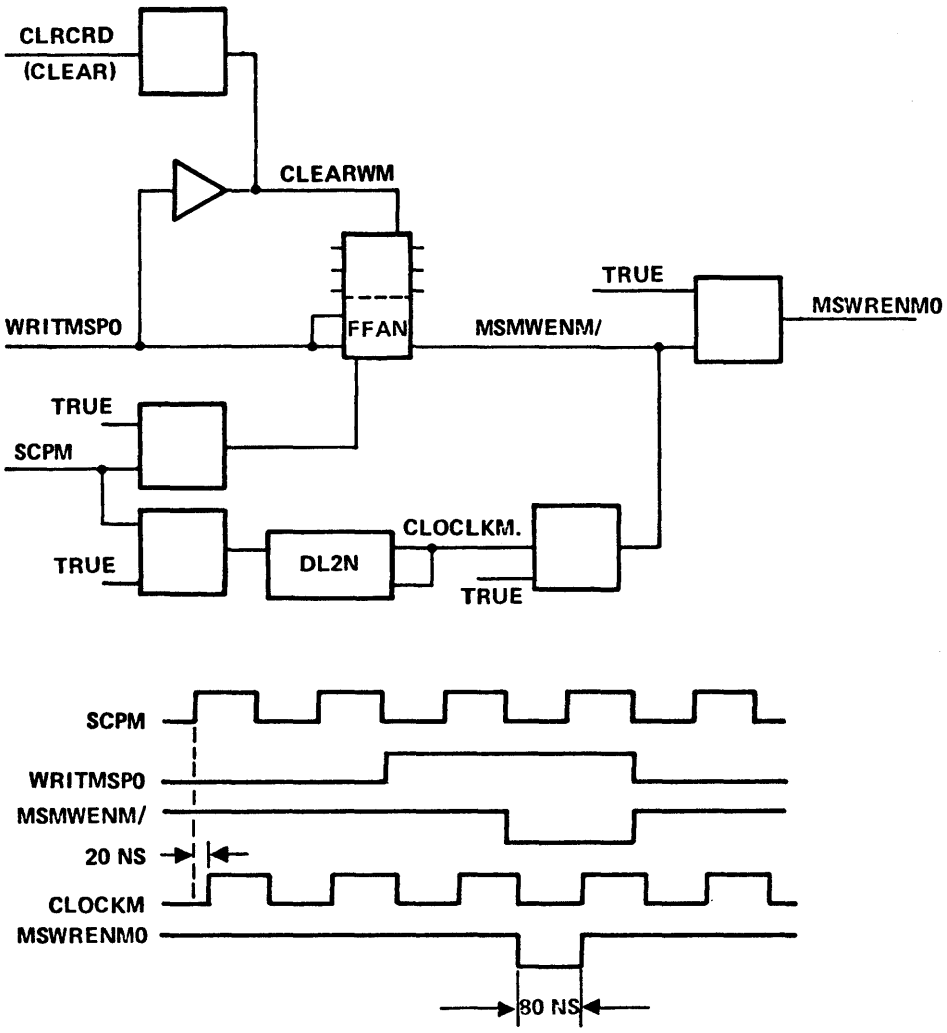


Figure 2-92. MSM Write Enable Logic and Timing

MICRO EXECUTION AND TIMING

The B 1720 Series Central System Processor is always in one of five defined current states which gives rise to the term "finite state machine." These five states are:

Term	Definition
CONSCSP.	Console current state
STRTCSP.	Start current state
EXECCSP.	Execute current state
FTCHCSP.	Fetch current state
STOPCSP.	Stop current state

Each of these states, when active, allows certain operations to take place within the processor as follows:

Term	Description
CONSCSP.	Load/display functions are enabled.
STRTCSP.	True for one system clock pulse and is entered from CONSCSP. by the START pushbutton. In MTR mode, the cassette drive is started.
EXECCSP.	This is the normal operating state of the machine in which the fetching (from MSM) and execution of micros takes place. This state is entered from STRTCS or FTCHCS and may exit to FTCHCS or STOPCS.
FTCHCS.	This state is entered from EXECCS when it is necessary to fetch a micro from some source (S-memory or U-register) other than M-memory.
STOPCSP.	True for one system clock and stops the cassette drive if running and enters CONSCS.

CURRENT STATE LOGIC

The current state of the machine is stored in the three flip-flops: BMAC1F (24), BMAC2F (23), and BMAC4F (22) which are the basic micro and address control flip-flops. Refer to figures 2-93 and 2-94. All three flip-flops operate in the D-set mode when CHGBCDP1 (change basic control state) is true.

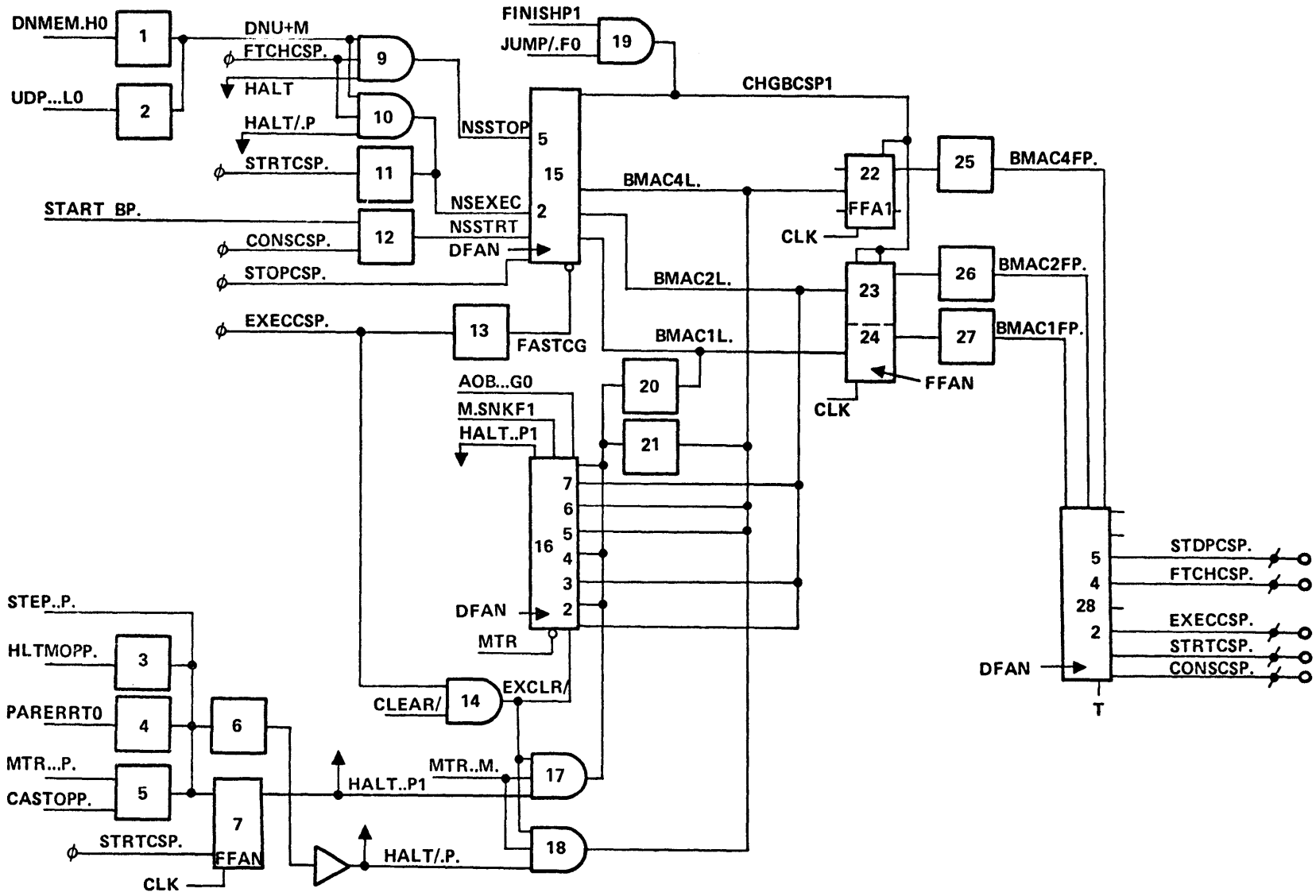


Figure 2-93. Current State Logic

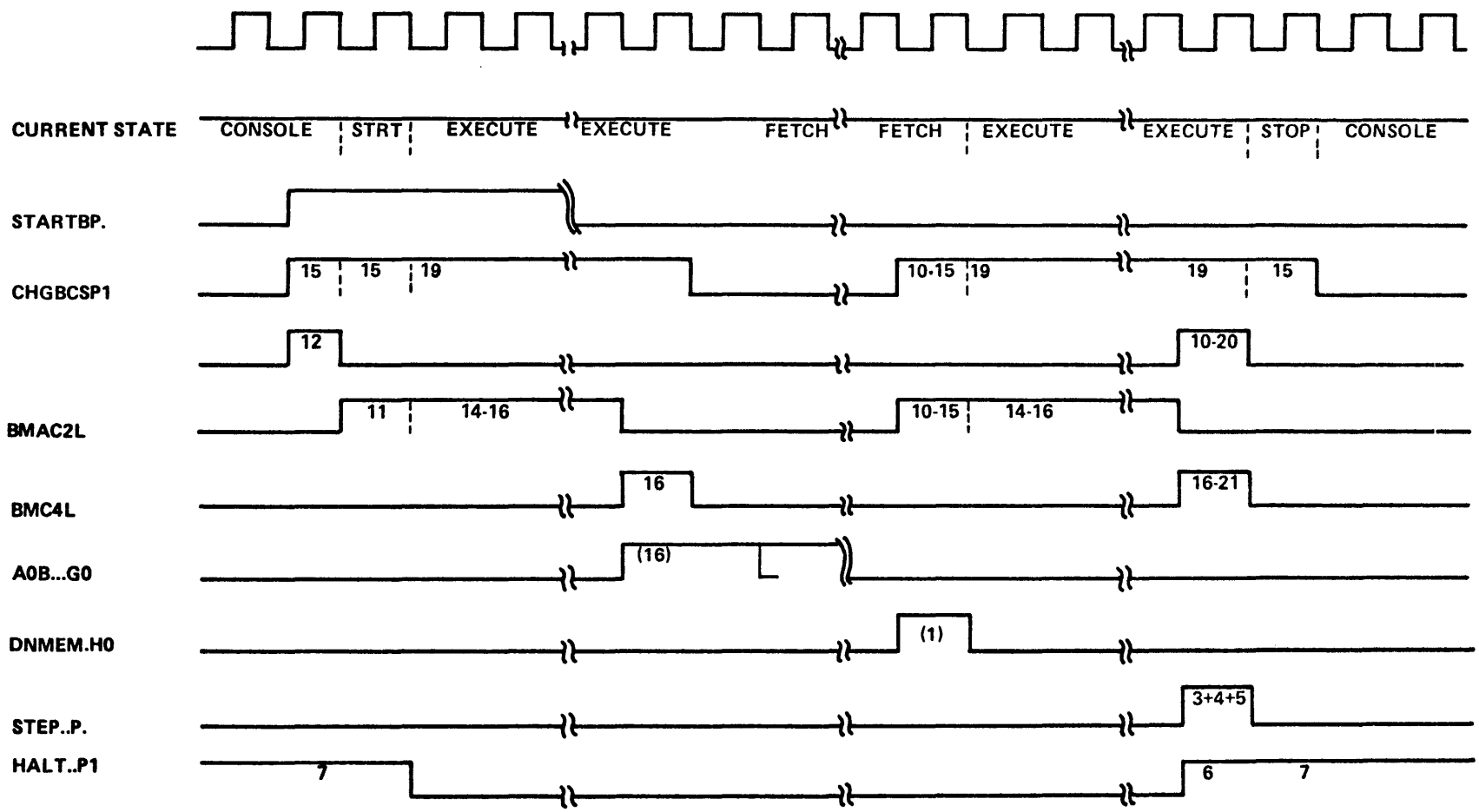


Figure 2-94. Current State Timing

After a power on, a clear, or halt the processor is in CONSCS with flip-flops 22, 23, and 24 all being reset, thereby enabling the zero output of DFAN 28. At this time EFAN 15 is enabled by the output of buffer 13 being false. Depression of the START pushbutton enables buffer 12 and makes NSSTRT (next state start) true. With EFAN 15 enabled, NSSTRT is true making CHGBCS and BMAC1L (basic micro and address control 1 level) true. The next DSCP sets flip-flop 24 which is decoded through DFAN 28 to enable STRTCS.

STRTCS is ANDed with MTR to enable CSTARTOP and DSETM so that if the processor is in MTR mode, the cassette drive is started and zeros are clocked into the M-register. STRTCS also goes to the reset input of the HALTF.F. (7) and to gate 11 to make NSEXEC (next state execute) true which is encoded through EFAN 15 to make CHGBCS and BMAC2L true. The next clock after entering STRTCS sets BMAC2F (23) to decode EXECCS through DFAN 28, and resets HALTF.F.

With EXECCS now true the output from buffer 13 is true and EFAN 15 is disabled. At the same time the output of buffer 14 is true, which enables DFAN16 if MTR is false. Provided there are no exception conditions, i.e., AOB (A out-of-bounds), M..SNK (M-register sink), or halt, the zero output of DFAN 16 is true which maintains a true on the D-input of flip-flop 23. Each time FINISHP1 is true (at the end of execution of each micro instruction) and JUMP/ is true (indicating that the next sequential micro is to be executed), CHGBCS is true from gate 19 putting flip-flops 22, 23, and 24 in D-set mode. Flip-flop 23 remains set and 22 and 24 reset, causing the processor to remain in EXECCS executing micro instructions and testing each time FINISH and JUMP/ are true for an exception condition.

Micro instructions are normally fetched from M-string memory during one system clock time while the previous micro instruction is being executed. If the A-register address is greater than TOPM then AOB is true (A out-of-bounds), and the next micro to be executed must be fetched from S-memory. AOB true enables output number 4 of DFAN16 making the level BMAC4L true. With the finish level true at the completion of execution of the current micro in the M-register, BMAC4F (22) sets, BMAC2F resets, and BMAC1F (24) remains reset so that 1) FTCHCS is decoded from DFAN28, 2) EFAN15 is enabled, and 3) DFAN16 is disabled. When a fetch from S-memory takes place, the following occurs: at the end of the memory cycle DNMEM.HO (done memory) comes true, and through gates 1 and 10 makes NSEXEC (next state execute) true. As EFAN15 is now enabled the next DSCP sets BMAC2F (23) and resets BMAC4F (22) to decode EXECCS and allow execution of the micro instruction which has just been fetched from S-memory.

If the Processor is in MTR mode, then DFAN16 is not enabled when entering EXECCS from STRTCS. With EXECCS and MTR true, gate 18 is enabled, forcing BMAC4L true. Since zeros were gated into M-register as the processor went from STRTCS to EXECCS, A one-clock no-op is executed. Finish and Jump/ are true at this time so that BMAC4F (22) is set and BMAC2F is reset on the next clock pulse (after entering EXECCS in MTR mode). This changes the output of DFAN 28 to FTCHCS.

The processor now waits while a micro instruction is read from the cassette tape and assembled in the U- Register. When a complete micro has been read into the U Register UDP...LO (U data present) comes true. The micro instruction is gated through the MEX and into the M-register and simultaneously UDP...LO true makes NSEXEC true from gates 2 and 10. The next DSCP sets BMAC2F (23) and resets BMAC4F (22), to decode EXECCS from DFAN 16. The micro read from tape (and now in M-register) is executed. At the completion of execution FINISH is true, again putting flip-flops 22, 23, and 24 in D-set mode. Gate 18 forces BMAC4L true and BMAC4F sets and BMAC2F resets. The processor has now returned to FTCHCS and waits for the next micro instruction to be read into the U-register.

HALT LOGIC

The signal HALT..P1 may come true from several sources:

Source	Description
HLTMOPP.	Halt micro decode
PARERRTO	Parity error
MTR·CASTOPP.	MTR mode and cassette stop micro
SNGMOD	Single step mode
HALTB.P.	HALT push button
A=CSW·NULLREG·EXECCSP	A-register address equals console switches and null register and execute current state.

Any of the above conditions brings STEP..P. true (figure 2-93) which, through gate 6, makes HALT..P1 true and forces a true on the set output of HALTF..F. From this point, the halt process depends on the operating mode of the processor. Refer to figure 2-95 for Halt timing.

In RUN mode and EXECCS, gates 17 and 18 are disabled and DFAN 16 is enabled. HALT..P1 true is decoded by the DFAN16 to make the #1 output true. Through buffers 20 and 21, BMAC1L and BMAC4L are true. After execution of the current micro, FINISH and JUMP/ at gate 19 allow BMAC1F (24) and BMAC4F (22) to be set and BMAC2F (23) to reset. STOPCS is decoded from DFAN28. Stop current state forces CSTPAGPO true (card P, page 1) to stop the cassette drive at the next gap. EFAN15 is now enabled and, with STOPCS on the zero input, CHGBCS is true and all the BMACnL signals are false. The next DSCP resets BMAC1F, BMAC2F, and BMAC4F to put the processor in the console current state and enable the load/dispaly functions. The processor can only return to EXECCS by depression of the start push button at gate 12. If the processor had been in FTCHCS (when HALT..P1 came true) then the fetch would continue until DNU+M came true. Through gate 9 NSSTOP comes true and the processor goes from FTCHCS to STOPCS to CONSCS with the next in-line micro to be executed in the M-register.

In MTR mode, DFAN16 is disabled and is bypassed by gate 17 to force a similar function as when in the run mode. If, in the run mode, EXECCS, FINISH, and HALT..P1 are true but JUMP/ is false, this would imply that a branch or skip micro is currently being executed and the jump is to be taken. In this case, one extra clock pulse is required before halting to allow the new micro instruction at the "branch to" address to be fetched and the "branch to" address in A-register to be incremented by one to address the next in-line micro. Jump/ is true at the extra clock pulse and the halt logic takes the processor to the console state (as before) with the next micro to be executed in M-register, and A-register addressing the next micro to be fetched.

Halt - Console Switches Equal A-Register

When null register is selected by the console dial settings and the processor is in run mode and executing micro instructions, the processor halts whenever the contents of A-register match the console switch setting (starting with the fifth switch position from the right and ignoring the leftmost six positions). See figure 2-96. Card M (page 6) contains three CFAN's where a comparison is made: compare console switches CSW04 through CSW17 to A-register bits A00F through A13F.

When an equal comparison is found, A=CSW.M0 is true. This signal goes to card P (page 8) where it is ANDed with EXECCSP. and NULREGP. to give STEP..P. true and bring the processor to a halt.

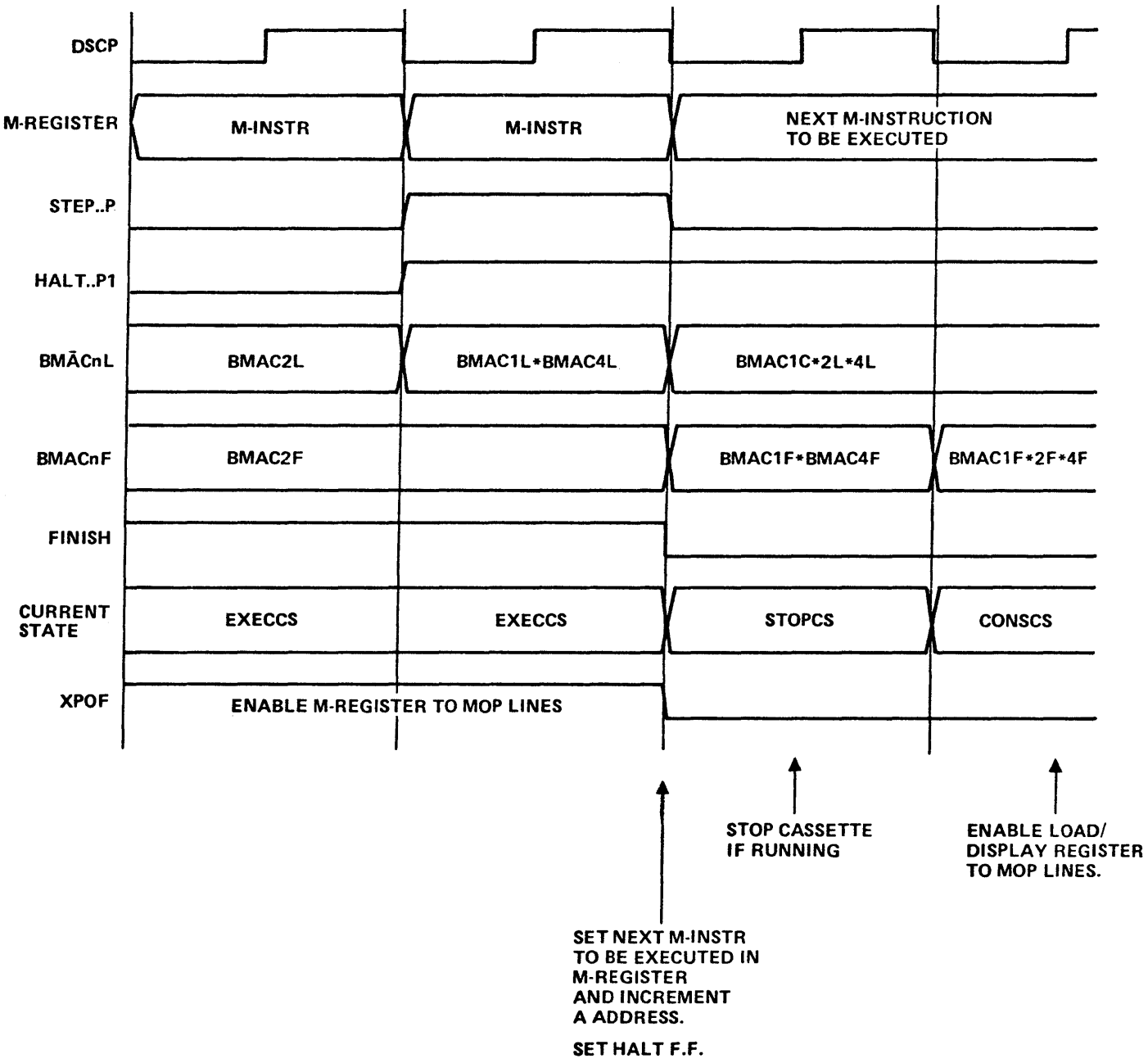


Figure 2-95. Halt Logic Timing

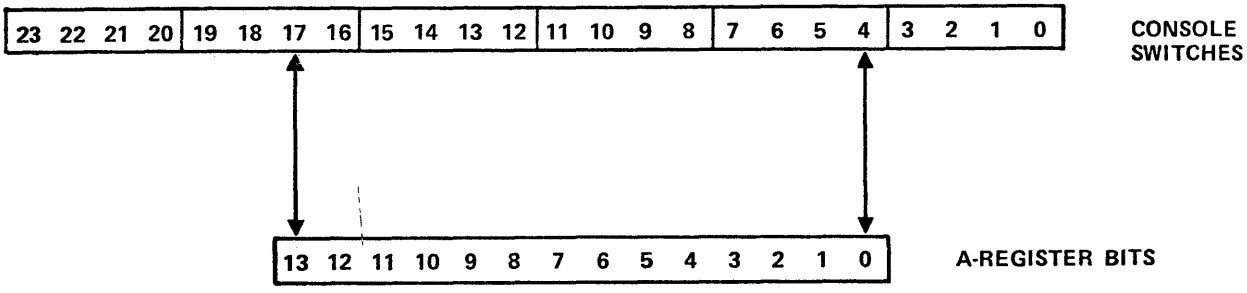


Figure 2-96. Console Switches/A-Register Alignment

MICRO EXECUTION - M-REGISTER AND A-REGISTER

The micro instructions used in the B 1720 Processor may be subdivided into the three following types. Refer to figure 2-97.

1. Micro instructions normally requiring one clock period for execution, but possibly requiring up to four clocks when exception conditions arise. Included in this group are: 1C, 2C, 3C, 45C, 6C, 8C, 9C (3-clock micro), 10C, 11C, 123C (2-clock micro), 145C (2-clock micro), 3D, 6D, 7D, 8D (2-clock micro), 9D, 2E, 3E, 4E, 5E, 6E, 1F, and no-op micros.

2. a. Multiple function micros including: 7E (exercise MSM), 2F (overlay M-string), and 4F (bind).

   b. Variable length micros including: 4D and 5D (shift/rotate X and/or Y) and 3F (normalize X).

3. Basic 3 micros including: 7C (read), 7C (write), 2D (swap), and the 1E (dispatch).

In type 1 and 2b(above), execution time is determined by FINISHP1 and EXRFINP1 (run mode) or EXMFINP1 (tape mode).

In type 2a (above), the multiple function current states control execution time.

In type 3 (above), the concurrency logic makes the level FORCEFP1 true (force finish) when the operation is complete.

The following sections are primarily concerned with execution timing of micros in the type 1 category. Multiple function current states and concurrency are covered in later sections.

Any micro instruction to be executed is placed in the M-register, from where it is gated to the MOP lines and decoded. In the M-memory processor, micro instructions are normally stored in M-string memory from where they are read at addresses determined by A-register. From MSM, each fetched micro is gated directly into the M-register at the completion of the previous instruction. When required, micro instructions may also be fetched from S-memory or from the U-register.

Basic micro execution time is one system clock. In this time the micro instruction is decoded, the data source is addressed or enabled, the data is manipulated and/or gated to the destination, and, on the trailing edge of the system clock, it is latched into its destination. Simultaneously as the micro is being executed the next in-line micro is being addressed by A-register and appears as MSM read data. This read data is gated to the M-register D-inputs by the level M ← MSM. With DSETM1P0 and DSETM2P0 true, the same clock pulse latches a new micro instruction into M-register, and also increments A-register to address the next in-line micro.

If a micro instruction takes more than one system clock to execute, then DSETM and INCA (increment A-register) is disabled. This allows the M-register to hold the currently-executing micro until execution is complete, and A-register continues to address the next in-line micro.

When A-register is used as a destination (as in 1C move to A-register), or when A-register is modified (as in a branch or skip), then an additional clock must be allowed for the micro at the new A-register address to be fetched from M-string. This is achieved by disabling M ← MSM and having DSETM1P0 and DSETM2P0 true. Zeros are latched into the M-register and a one-clock no-op is executed during which time the micro at the new A-register address is fetched.

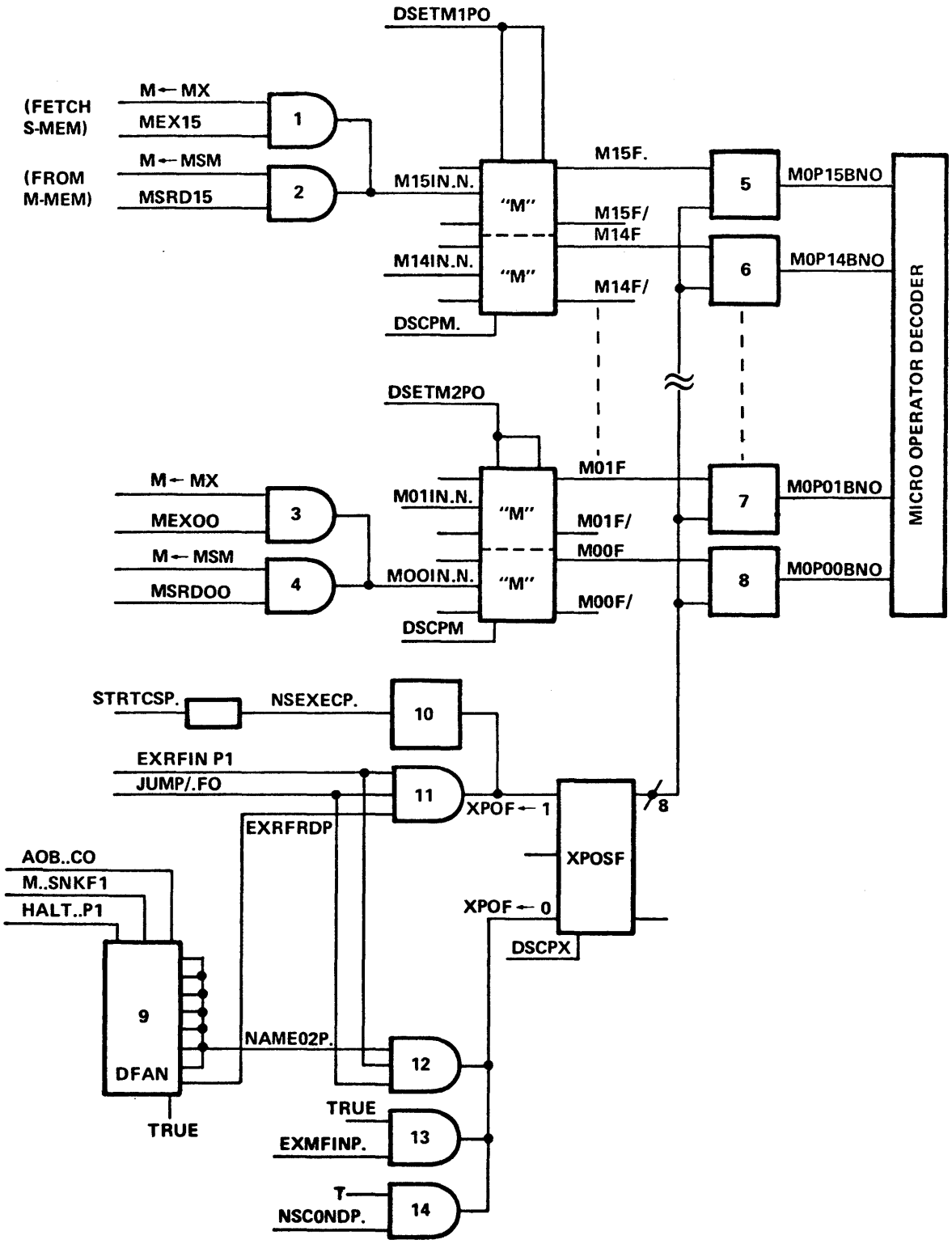


Figure 2-97. M-Register and Expose Flip-Flop

A no-op may also be forced by resetting the Expose flip-flop. XPOSF is set when the processor enters the execute current state and normally remains set to allow the M-register outputs onto the MOP lines. If an exception condition, A out-of-bounds, M-register sink or halt arises, then XPOSF is reset and no-op's executed until it is again set.

The XPOS flip-flop is also reset in the tape mode, this occurring after the execution of each micro. Repeated no-op's are then executed until the next in-line micro is read into the U-register (from the cassette), and gated to M.

If, during a concurrency check, the micro being checked is not found to be a concurrency micro, then NSCONDP. is true and the XPOSF is reset to prevent the micro from being decoded.

FINISH LOGIC - M-REGISTER AND A-REGISTER CONTROL

Consider the processor to be in run mode and console current state with the HALTF.F. set. Refer to figures 2-98, 2-99, and 2-100. The M-register contains the next micro instruction to be executed and the A-register contains the address of the next micro to be fetched. With depression of the START pushbutton the processor enters STRTCS. With the next clock pulse, XPOSF (expose flip-flop) is set (gate 10, figure 2-97) and the processor enters the execute current state (EXECCSP1 is true). With EXECCSP1 and RUN...P. true, DFAN17 and DFAN14 are enabled. Also with XPOSF set, the M-register is gated to the MOP lines and decoded. The inputs to DFAN's 20, 17, 14 are false, unless:

a. SUPFINP1 is true if the micro instruction is a type 2a multiple function, 2b variable length micro, or type 3, basic 3 or dispatch.

b. SORHOP1 (source holdover) is true if a type 1 micro instruction decodes an exception condition such as slow source.

c. SNKHOP1 (sink holdover) is true if a type 1 micro instrucion decodes an exception condition such as a slow sink register.

Assume the three signals above to be false. With DFAN's 17 and 14 enabled, FINISHP1 and EXRFINP1 are both true. Provided a branch or a skip is not made and AOB/ (A-address out-of-bounds not) is true, then gates 3, 7, and 13 are enabled. This makes INC A (increment A-register), DSETM (M-register D-set mode control) and M ← MSM (M-string memory read data to M-register) all true. The next clock pulse D-sets a new micro into M-register and increment A-register to address the next micro to be fetched. EXRFINP1 and FINISHP1 normally remain true, enabling gates 3, 7, and 13. Under these conditions micro instructions are fetched and executed in one clock period until an exception condition arises.

The following paragraphs detail the various exception conditions experienced in the finish logic operations. These include:

a. Branch

b. Skip

c. A Out-of-Bounds

d. Slow Source including:

   1. Slow 24-Bit

   2. Slow 4-Bit

   3. Slow Scratchpad Read After Write

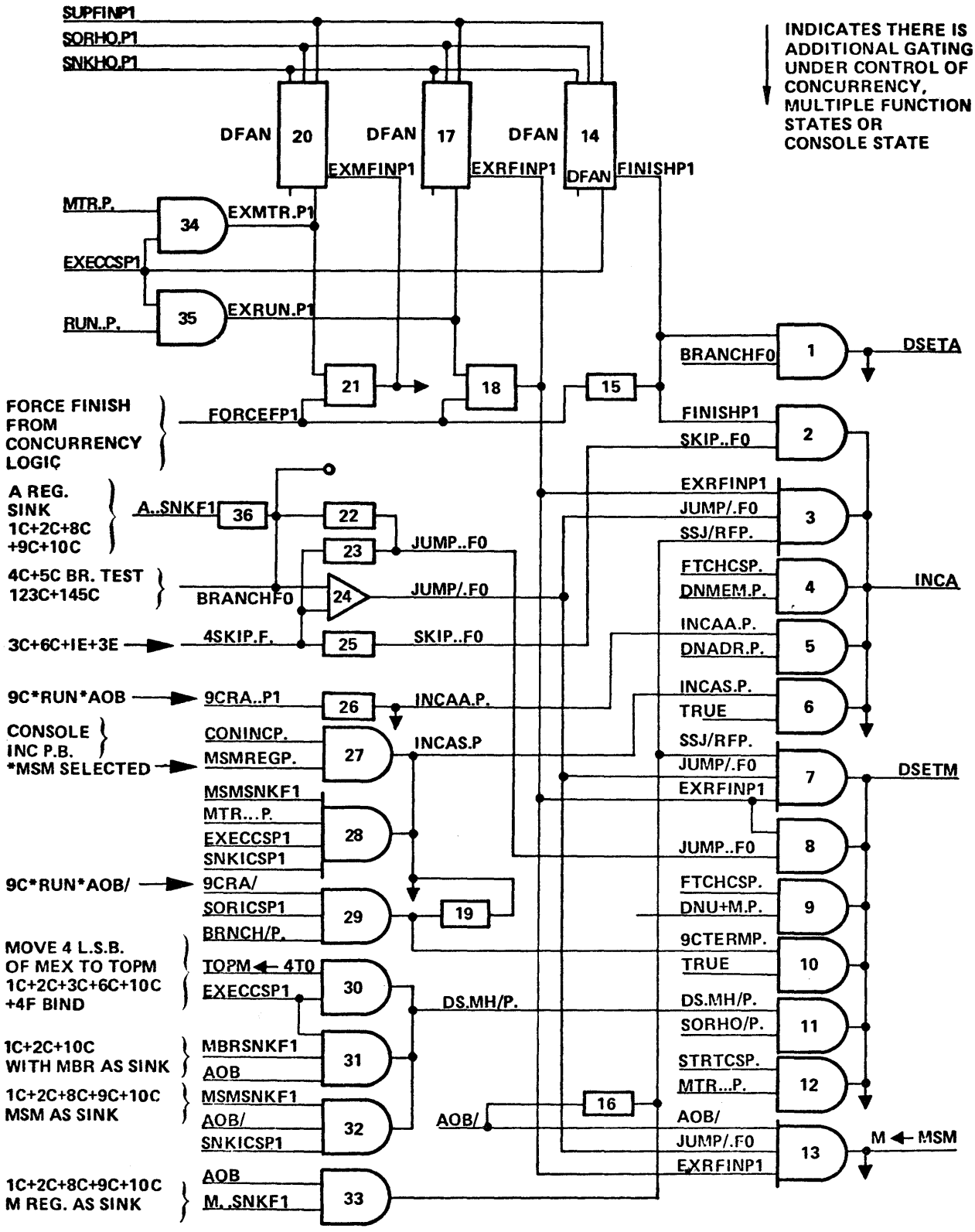


Figure 2-98. Finish Logic: M-Register and A-Register Controls

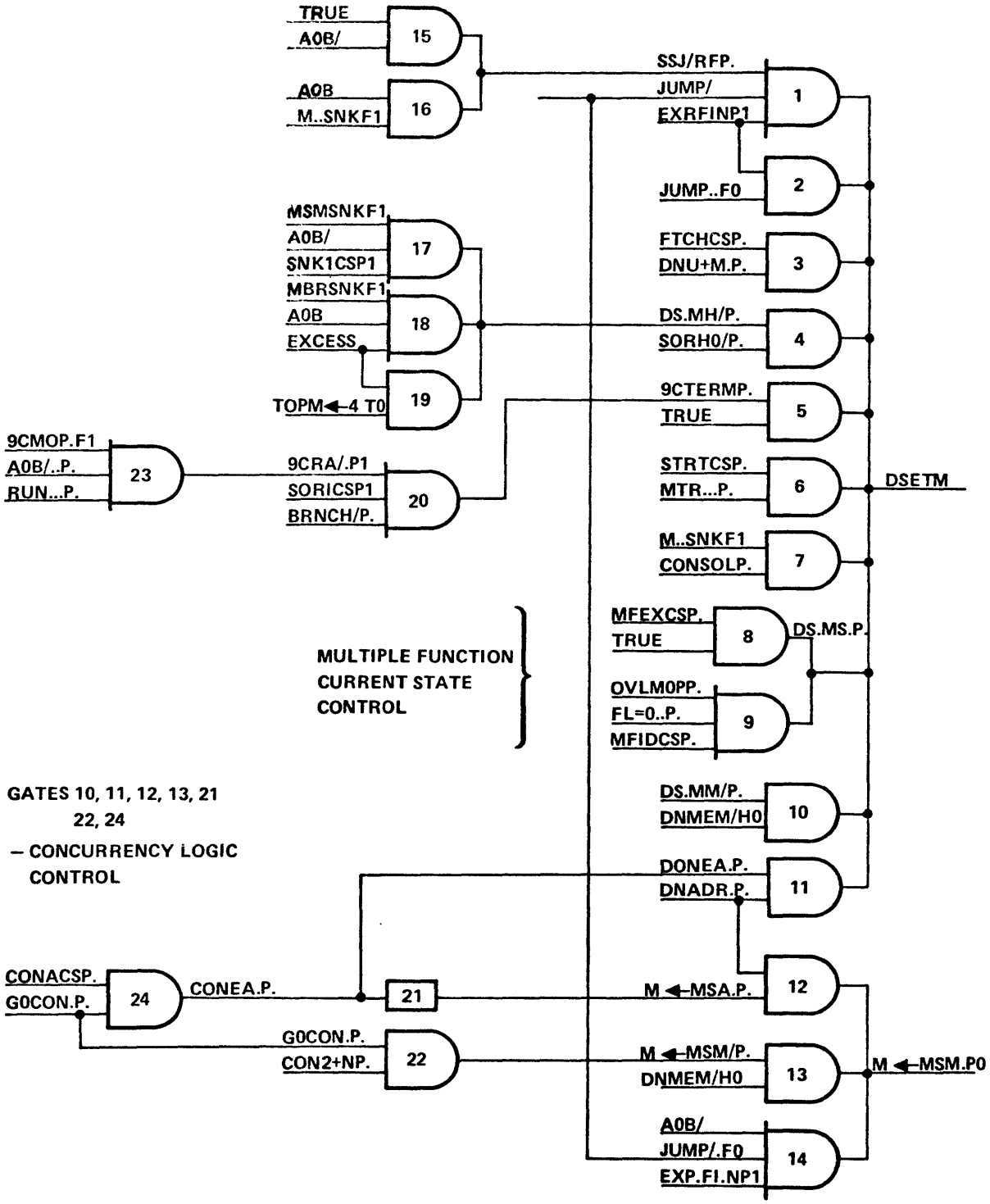


Figure 2-99. M-Register Controls: D Set M, M MSM

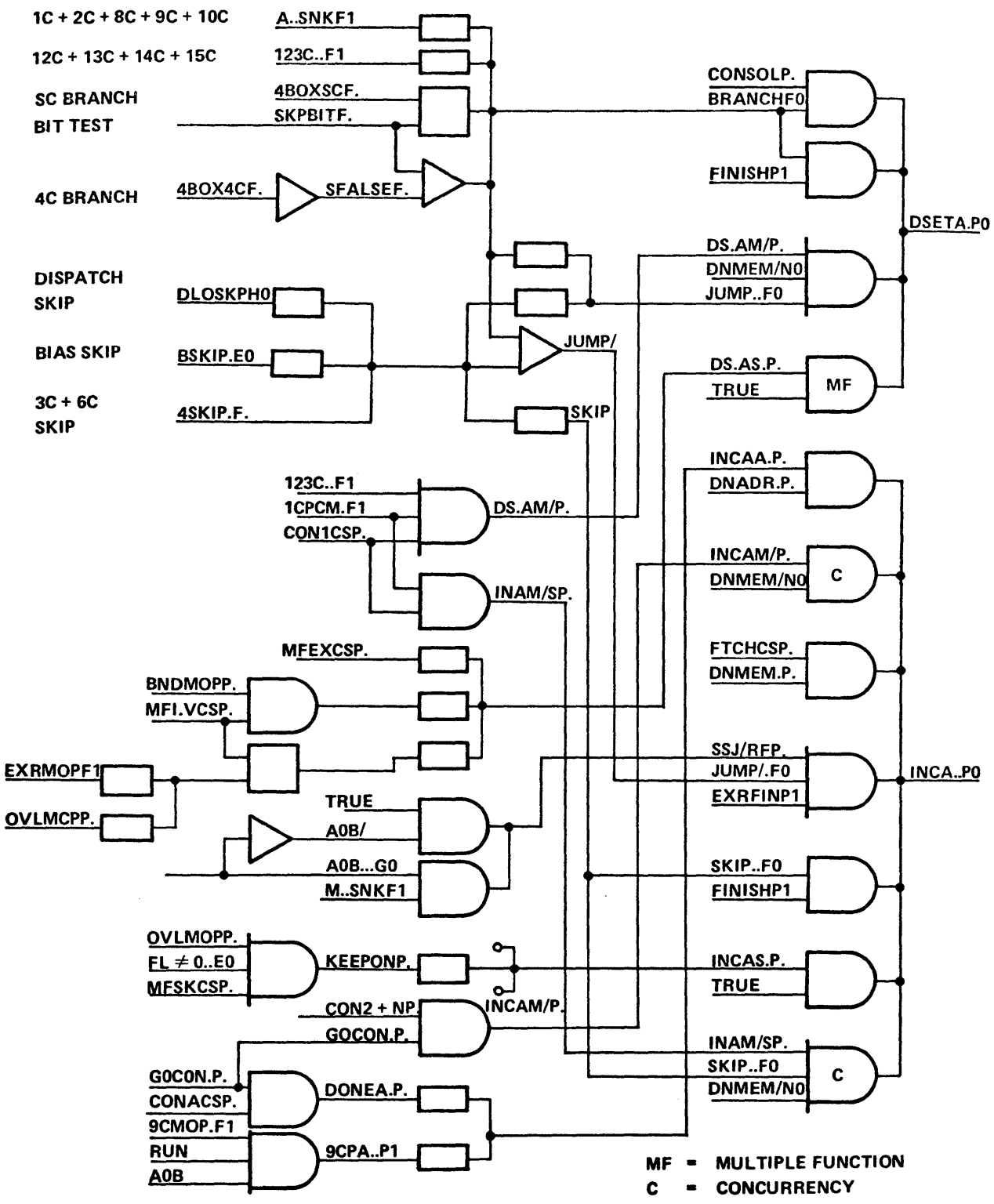


Figure 2-100. A-Register Controls: DSETA.P0 , INCA..P0

e. Suppress Finish, Source Holdover or Sink Holdover including:

   1. Suppress Finish

   2. Source and Sink Holdover

f. M-String Memory as Sink and AOB/

Branch

This may occur during execution of a 4C or 5C branch test micro, when the branch is to be made. In this case the term BRANCHFO is generated. The 12C or 13C branch relative and 14C or 15C call relative micros may also generate BRANCHFO, as well A-register being the destination on a 1C, 2C, 8C, 9C, or 10C. (See logic schematics card F, page 3 for BRANCHFO generation.)

With BRANCHFO true, JUMP..FO is true and JUMP/.FO false through gates 22 and 24. Gates 3 and 13 are disabled and gate 1 enabled, giving D-SETA true (A-register D-set mode control) and DSETM true. The next clock sets the "branch to" address in A-register and sets zeros in M-register. With zeros in M-register, BRANCHFO goes false and a one clock no-op is executed during which time the new micro is fetched from the "branch to" address. At the next clock pulse with gates 3, 7, and 13 again enabled, the new micro is set in M-register and A-register is incremented. Executing the branch micro instruction (or A-register as a destination) takes one clock plus a one clock no-op to fetch the micro at the new address.

Skip

This is true when the operation indicates that the skip is to be taken on a 3C 4-bit manipulate, 6C skip when, 1E dispatch, or 3E bias. 4 SKIP.F. true makes JUMP..FO and SKIP..FO true and JUMP/.FO false. Gates 3 and 13 are disabled and gate 2 enabled to make INC A and DSETM true. The next clock increments A-register and sets zeros into M-register. A one clock no-op is executed while the micro at the "skip to" address is fetched. Thus, the micro has been executed in one clock (except dispatch which is a multiple clock micro). If the skip is to be taken, the next in-line micro is replaced by a no-op and the following micro is fetched.

A Out-Of-Bounds (AOB)

AOB true indicates that the next micro instruction to be executed is not stored in M-string memory but must be fetched from S-memory. AOB true disables gates 3, 7, and 13. DSETA, INCA, DSETM, and M ← MSM are all false. With finish of the currently executing micro, the clock pulse resets the XPOSF.F. (figure 2-101, gate 6), and the processor enters FTCHCS (fetch current state, figure 2-93). DFAN's 17 and 14 are disabled and EXRFINP1 and FINISH go false. A memory cycle is initiated to fetch the next micro from S-memory at the address determined by A+MBR. When the memory cycle is complete and the new micro is on the main exchange, the level DNMEM..HO (done memory) comes true. Gates 4 and 9 (figure 2-98) are enabled, as are gates 1, 9 and, 4 (figure 2-101). Thus, with DNMEM.HO coming true the fetch current state, INCA, DSETM, M←MX, XPOF , and NSEXEC (next state execute) all come true. The next clock sets the micro on the main exchange into M-register, increments A-register, sets the expose flip-flop, and the processor enters the execute current state.

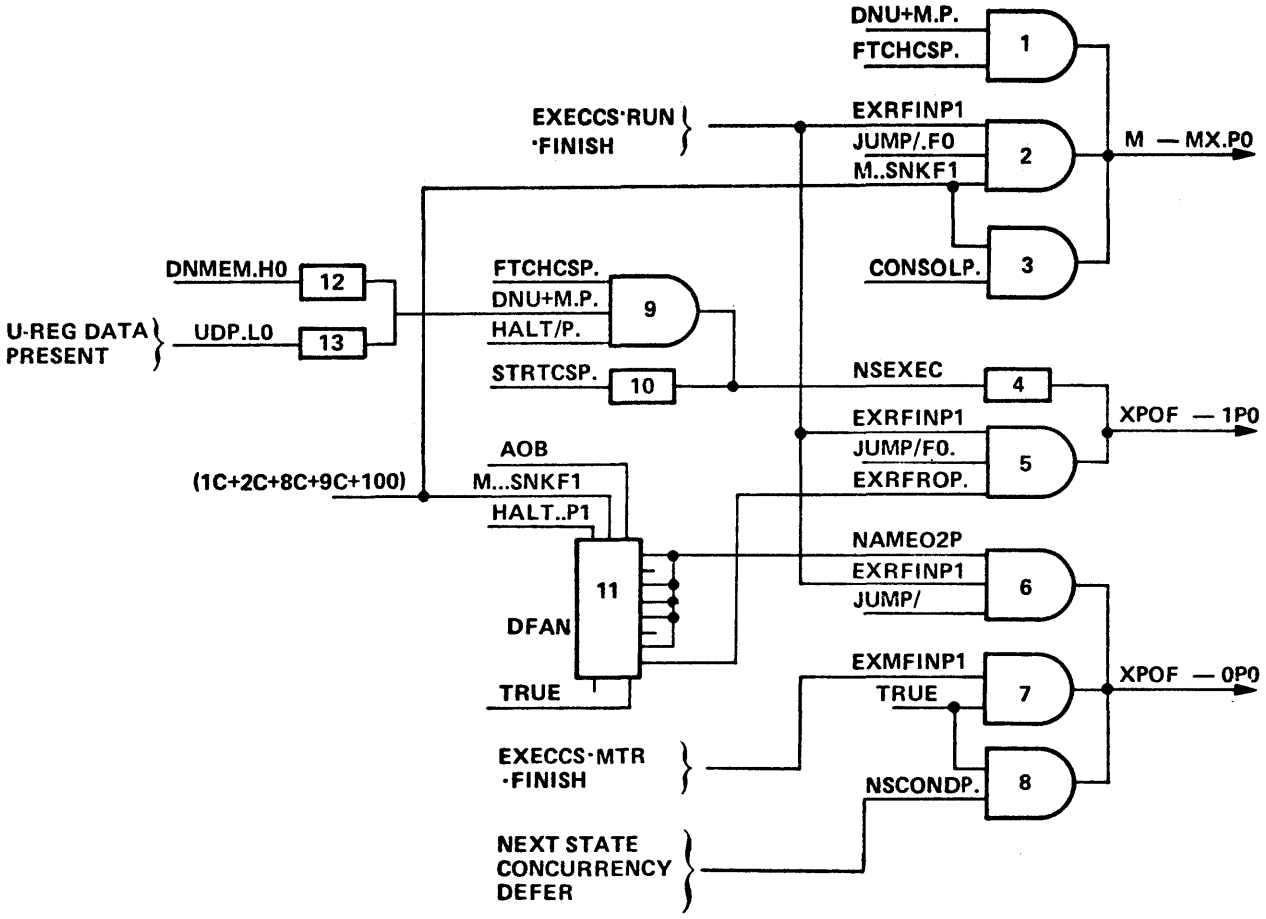


Figure 2-101. M◄MX..P0 ,XPOF◄1P0 , XPOF◄0P0

SLOW SOURCE

Because of the propagation delays inherent in certain electronic circuits, several 24-bit and 4-bit register sources require more than one system clock (164 nanoseconds at 6 megahertz) to produce a stable output and perform the function according to the micro being executed. Refer to figure 2-102. Such registers are SUM and DIFF (binary or decimal), BICN, FLCN, XYCN, XYST, and scratchpad read after write. As the basic micro execution time is one system clock, gating is provided to produce the signal SLOSORF1 when any of these registers are used as a source, and extend micro execution time by one clock period. In addition, the 8D scratchpad relate FA micro, which is always a two clock micro, utilizes the slow source logic to obtain the second clock.

Slow 24-Bit (Card L, Page 3)

The output from gate 16 or 17, SUMDIFL., is true any time SUM or DIFF registers are decoded as a source from MOP lines 6 through 11. SUMDIFL. is gated with CREGO5L1 (C-register, bit 5) at gate 6 and with CUBOOOL. at gate 5, so that regardless of whether the operation is binary (CUBOOOL. true) or decimal (CREGO5L1 true), I+-HLDL. is true if SUM or DIFF is used as a source. At gate 4, MX← C03L. is true if the micro being executed is a 1C move, or a 2C move register to scratchpad (MOPO5-false). The output of gate 4 is SL024.LO which goes to card F (page 5) and through buffer 2 gives SLOSORF1.

If an 8D micro is decoded, then 2CPCM.F1 is true (two clock pulse concurrency micro). Through gate 17, SL024.LO is true and execution time extends by one clock period. On the first clock, FFAN 18 sets (disabling gate 17), and on the second clock resets as a new micro is set in the M-register.

Slow 4-Bit (Card F, Page 6)

SLOW4.F. is decoded from the outputs of the M-register flip-flops, rather than the MOP lines. This is to give faster decoding of 4-bit functions, which generally require more levels of gating than 24-bit functions. SLOW4.F. is true anytime the source register is BICN, FLCN, XYCN, or XYST on a 1C, 2C, 3C, 4C, 5C or, 6C micro. At gate 3, SLOW4.F. is ANDed with MC3 + 6.F. which is true for a 3C (4-bit manipulate) or a 6C (skip when) micro instruction. Consequently SLOSORF1 is true and execution time extended by one clock when BICN, FLCN, XYCN, or XYST are used as sources on a 3C or 6C micro.

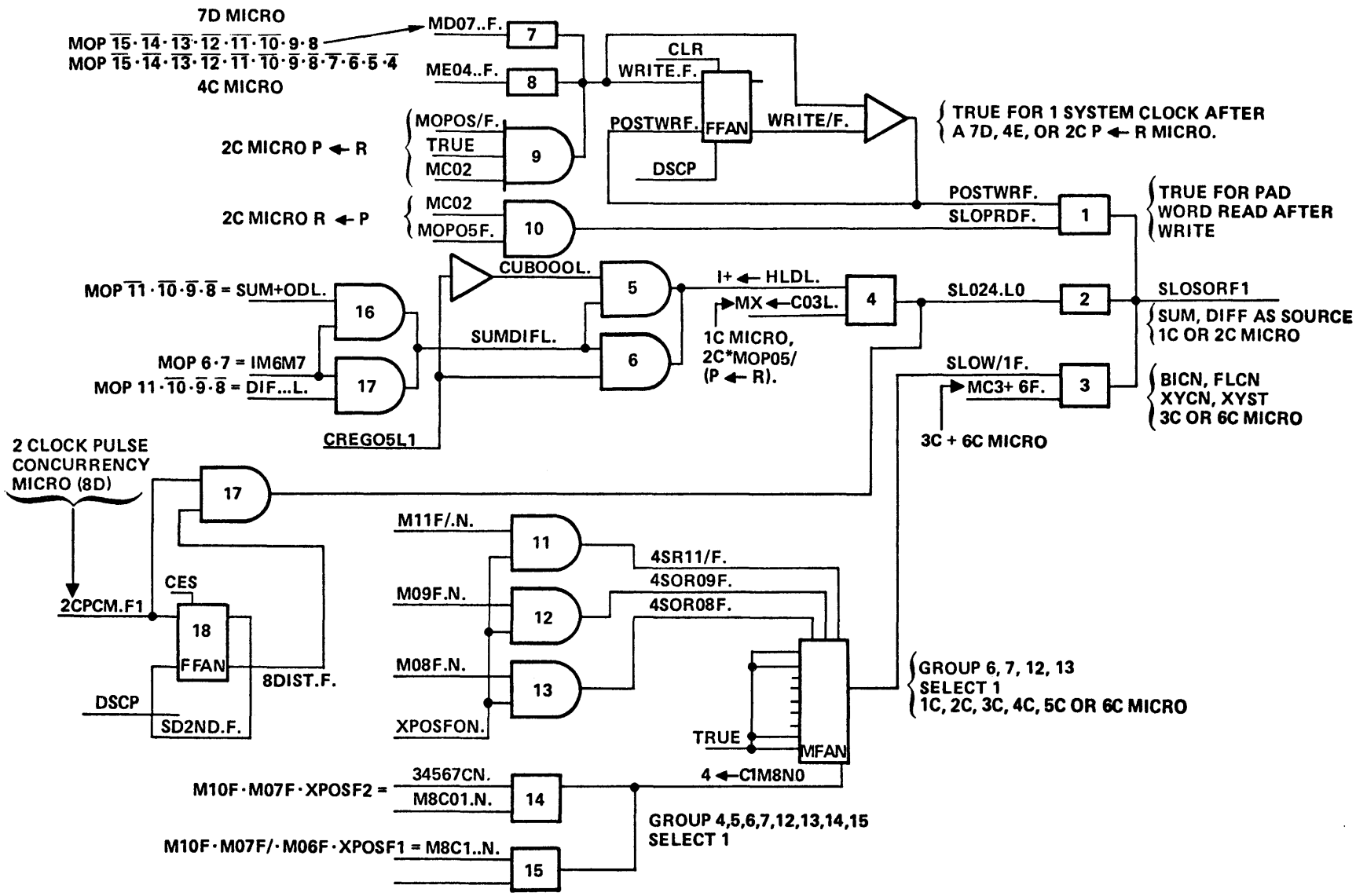


Figure 2-102. Slow Source Logic

Slow Scratchpad Read After Write (Card F, Page 5)

Scratchpad write micros (2C P←R, 7D and 4E) normally take one clock to execute. On the trailing edge of the clock the write data is latched into a holding register and the sink scratchpad word address is stored. At this time the processor is released to execute the next micro; however, the actual writing of the data stored in the holding register does not take place until approximately halfway through the next clock period. In the event that the next micro to be executed is a scratchpad read (2C R←P), execution time must be extended by one clock to give sufficient time to give sufficient time to write the data from the preceding micro into the scratchpad word, then read data from a new scratchpad address.

Anytime a 2C P←R, 7D or 4E micro is executed, gate 7, 8, or 9 makes WRITE.F. true which puts a true on the set input of the FFAN. The flip-flop sets with the next clock pulse, and simultaneously a new micro is latched in M-register and WRITE.F. goes false. Two false signals into the inverter make POSTWRF. (post write) true. If the next micro is a 2C move padword to register, then SLOPRDF. (slow scratchpad read) is true from gate 10, and gate 1 makes SLOSORF1 true. The FFAN remains set for one clock, and SLOSORF1 true extends micro execution time by one clock.

There are two other micro instructions which do a scratchpad read, but they do not require an additional clock period for execution after a scratchpad write. These are a 7D swap F with DPW and 5E load F from DPW. In both cases there is a direct path from the scratchpad to the F register. Scratchpad read data, therefore, does not have to pass through the main exchange. Thus, one clock period gives sufficient time to write the data from the previous micro, address the source scratchpad address on a 7D or 5E, and have stable inputs to F-register before the trailing edge of the clock.

SUPPRESS FINISH, SOURCE HOLDOVER OR SINK HOLDOVER

The three signals SUPFINP1 (suppress finish), SORHO.P1 (source holdover), and SNKHO.P1 (sink holdover) are used to disable the finish level and, therefore, extend the micro execution time from the normal one clock period to as many clock periods as are necessary to complete execution. Refer to figure 2-103.

Suppress Finish

Any time that one of the inputs of EFAN S0 is true, then SUPFINP1 is true. All the inputs except SUPFONP. are decoded from multiple clock micro operators as follows:

PICKUPPO	True when a basic 3 (7C read or write, 2D swap) is decoded or when concurrency is not in idle state.
NORMALP.	3F NORMALIZE·FL0..E0·MSBX≠1P.
DISP5.F1	1E dispatch. True for all variants except V = 5, port absent.
5+6S/RNO	4D + 5D shift/rotate X and/or Y·S/R count not equal to zero. Finish is suppressed until the S/R count = 0.
EXRMOPF1	7E exercise MSM
BNDMOPP.	4F bind
OVLMOPP.	2F overlay MSM

The above micros, except 4F bind (which is a three-clock micro), are all of variable length.

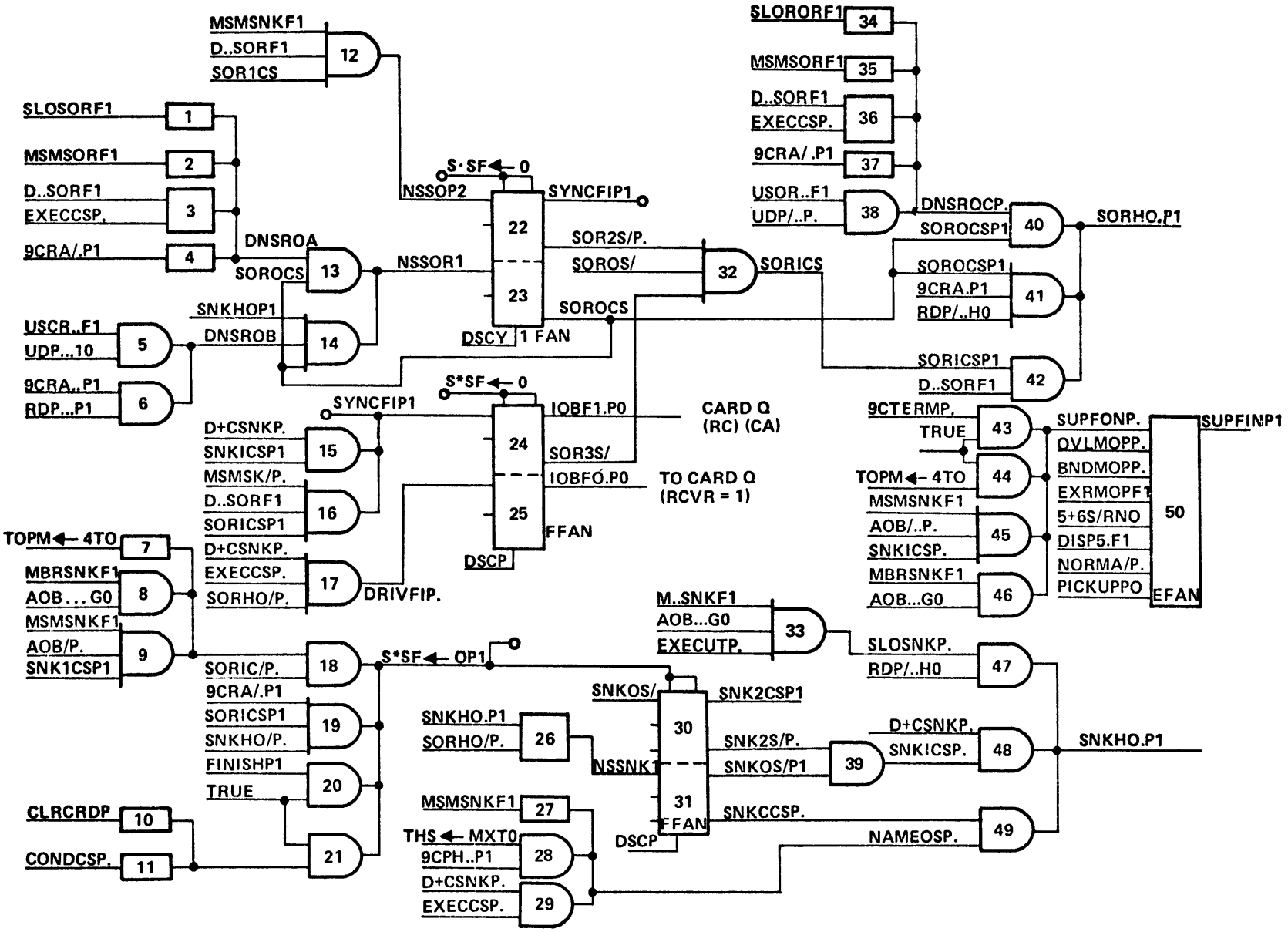


Figure 2-103. Suppress Finish, Source, and Sink Holdover Logic

Source and Sink Holdover

These two levels are primarily used to extend the micro execution time of basic one-clock micros, when certain exception conditions arise. Control is by four flip-flops (22, 23, 30, and 31). The flip-flops 22 and 23 make up the source sequencer. Their outputs, when gated together, give the timing levels SOROCS (source 0, current state), SOR1CS (source 1, current state), and SYNCF1P1 (synchronization flip-flop).

Flip-flops 30 and 31 make up the sink sequencer, the outputs of which give the timing levels SNKOCS (sink 0, current state), SNK1CS (sink 1, current state), and SNK2CS (sink 2, current state).

There are two additional flip-flops IOBF0 (24), input/output bus flip-flop zero; and IOBF1 (25), input/output bus flip-flop one. On any move to or from DATA or to CMND, IOBF0 synchronizes the I/O bus drivers/receivers and IOBF1 enables the command active or response complete signals (CA/RC) on the last clock period of the move.

When a micro operator completes execution (FINISHP1 true), gate 20 is enabled making S·SF←OP1 (zero-to-source and sink flip-flops) true and putting all the FFAN's in D-set mode. With the next clock pulse all the FFAN's are reset, and a new micro instruction is set in M-register. The source and sink zero current state levels SOROCS (23) and SNKOCS (31) are also true at this time. See table 2-10 for applicability of the various suppress/holdover control signals.

Table 2-10. Suppress Finish and Source/Sink Holdover Logic Signals

Mnemonic	Name	Applicability
A..SNKF1	A-register sink	1C + 2C (R ← P) + 8C + 9C + 10C
C..SNKF1	CMND sink	1C + 2C (R ← P) + 10C
D..SNKF1	Data sink	1C + 2C (R ← P) + 10C
D..SORF1	Data source	1C + 2C (P ← R)
D+CSNKP.	Data or CMND sink	
EXECCSP.	Execute Current State	
MBRSNKF1	MBR sink	1C + 2C (R ← P) + 10C
MSMSNKF1	M-string memory sink	1C + 2C (R ← P) + 8C + 9C + 10C
MSMSORF1	M-string memory source	1C + 2C (P ← R)
M..SNKF1	M-register sink	1C + 2C (R ← P) + 8C +9C + 10C
SLOSORF1	Slow source	SUM or DIFF as source on 1C +2C. BICN, FLCN, XYCN, XYST as source on 3C + 6C. 2C padword read after 2C, 7D or 4E padword write. 8D scratchpad relate FA.
UDP...LO	U-data present	True after 16 bits have been read from tape into U-register.
USOR..F1	U-register source	1C + 2C.
TOPM ← 4TO	Move 4 least significant bits of MEX to TOPM	1C + 2C + 3C + 6C + 10C where TOPM is sink (card R), 4F bind (card P).

Example 1: Source Holdover (see figure 2-104)

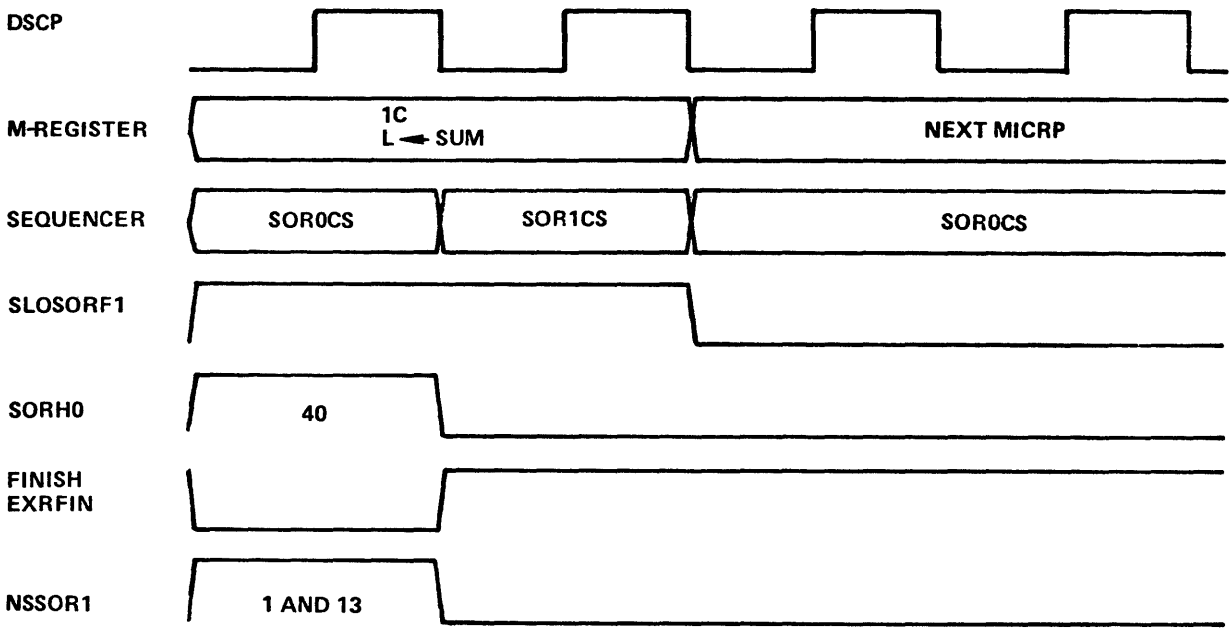


Figure 2-104. Micro Timing: Slow Source

Consider the new micro to be a 1C move sum to L. SLOSORF1 is true, which disables FINISHP1 and EXRFINP1 (assuming the processor is in run mode). Through gates 1 and 13, NSSOR1 is true (next state source one) and the first clock pulse sets FFAN 23. Gate 32 makes SOR1CS true and SNKOCS is still true. Gate 40 is now disabled and SORHO.P1 is false, allowing FINISHP1 and EXRFINP1 to come true. The second clock pulse now sets a new micro in M-register, and through gate 20, resets FFAN 23. The other FFAN's are already reset. Thus SLOSORF1 coming true has extended micro execution time by one clock period.

Example 2: Sink Holdover

Consider a 1C, 2C, or 10C micro instruction where the destination is DATA (or CMND). Refer to figure 2-105. D..SNKF1 is decoded from the micro and makes the level D+CSNKP. true (DATA or CMND Sink). Assuming execute current state and run mode, gates 29 and 49 are enabled, making SNKH0.P1 true, which disables FINISHP1 and EXRFINP1. SORHO.P1 is false. Gate 26 and gate 17 put a true on the set inputs of FFANs 25 and 31, respectively. The first clock pulse sets 25 and 31. SNKOCS goes false and gate 39 makes SNK1CS true. Gate 48 holds SNKH0.P1 true, and finish remains disabled. The setting of IOBFO.PO (25) enables the I/O bus drivers (card Q, page 3).

The set input to FFAN 30, SNKOS/P1 is now true and gate 15 makes the set input to FFAN 24 true. The second clock pulse sets FFANs 24 and 30. SNK1CS goes false and SNK2CS goes true. SNKH0.P1 is now disabled allowing FINISHP1 and EXRFINP1 to go true. IOBF1.PO (24) being set at this time is gated by D..SNKF1 to make RC true, or with C..SNKF1 to make CA true, depending upon whether DATA or DMND is the destination (card Q, page 7).

As FINISH is now true, the third clock pulse resets all the FFANs and set a new micro in M-register. With DATA or CMND as a sink, a normally one- clock micro instruction (1C, 2C, or 10C) has been extended to three clock periods. Similarly, if DATA is a source on a 1C or 2C, the execution time is extended to three clocks by the source sequencer.

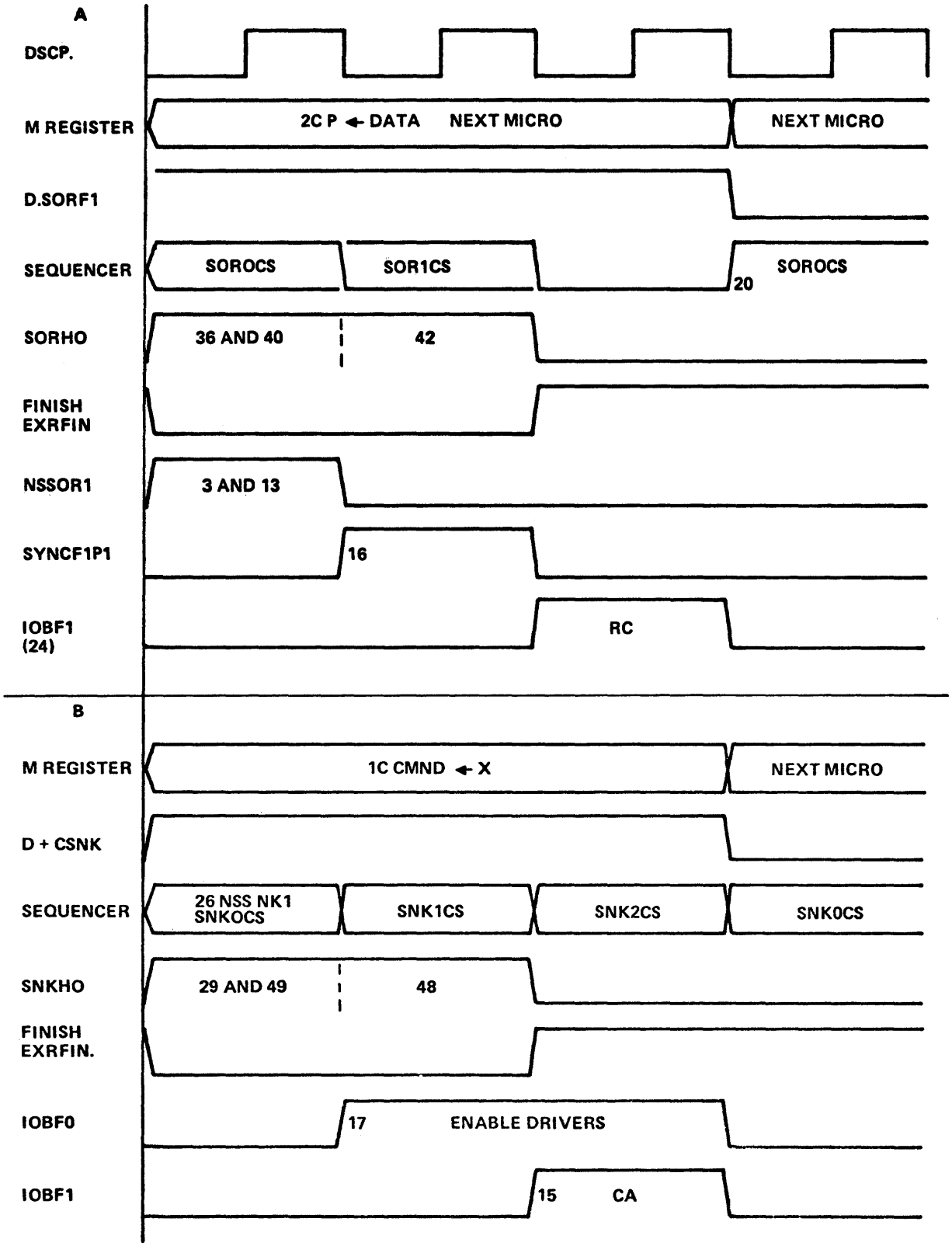


Figure 2-105. Micro Timing:-(A) Data Register as Source
(B) CMND Register as Sink

If a slow source is being moved to DATA or CMND (such as a 1C move DIFF to CMND), then execution time is extended to four clocks. Initially, gates 34 and 40 make SORHO.P1 true, and gates 29 and 49 make SNKHO.P1 true. The first clock pulse sets FFAN 23. SOR1CS and SNKOCS are now true and SORHO.P1 goes false, but SNKHO.P1 remains true. The second clock pulse sets FFANs 25 and 31. SOR1CS and SNK1CS are now true. Gate 48 holds SNKHO.P1 true. The third clock pulse sets FFANs 24 and 30. SOR1CS and SNK2CS are now true and IOBF1 (24) is set. SNKHO goes false allowing FINISH true and execution is completed with the fourth clock.

M-String Memory as Sink and AOB/

MSMSNKF1 may be decoded from a 1C, 2C, 8C, 9C, or 10C micro, where M-string memory is the sink. The source register data is written into MSM at the address indicated by A-register.

With MSMSNKF1 true and AOB/ true, gates 27 and 49 (figure 2-103) make SNKH0.P1 true, and FINISHP1 and EXRFINP1 go false. The first clock pulse sets FFAN31, and SNK1CS is true at gate 39. With SNK1CS true, gate 45 is enabled to make SUPFINP1 true so that FINISHP1 and EXRFINP1 remain false. Gates 9 and 18 are also enabled to put FFANs 22, 23, 24, 25, 30, and 31 in the D-set mode. In figure 2-98, gates 32 and 11 are enabled to make DSETM true. During the second clock period, 16 bits from the MEX are written into MSM at the address specified by A. With the second clock pulse, zeros are gated to M-register and the FFANs (figure 2-103) are reset. A one-clock no-op is executed, and during this time the next micro is fetched from MSM at the address indicated by the A-register (which is the same address to which data has just been written).

If AOB is true and MSMSNKF1 is true (as decoded from 1C, 2C, 8C, 9C, or 10C), 16 bits of data from the source register are written to MSM at the address designated by the A-register. This occurs during the second clock period. However, gates 45 and 9 (figure 2-103), and gates 32 and 11 (figure 2-98) are now disabled by AOB true. FINISHP1 is true, and with the second clock, the expose flip-flop is reset. The processor then goes into the fetch current state to fetch the next micro from S-memory. MSM as sink timing is shown in figure 2-106.

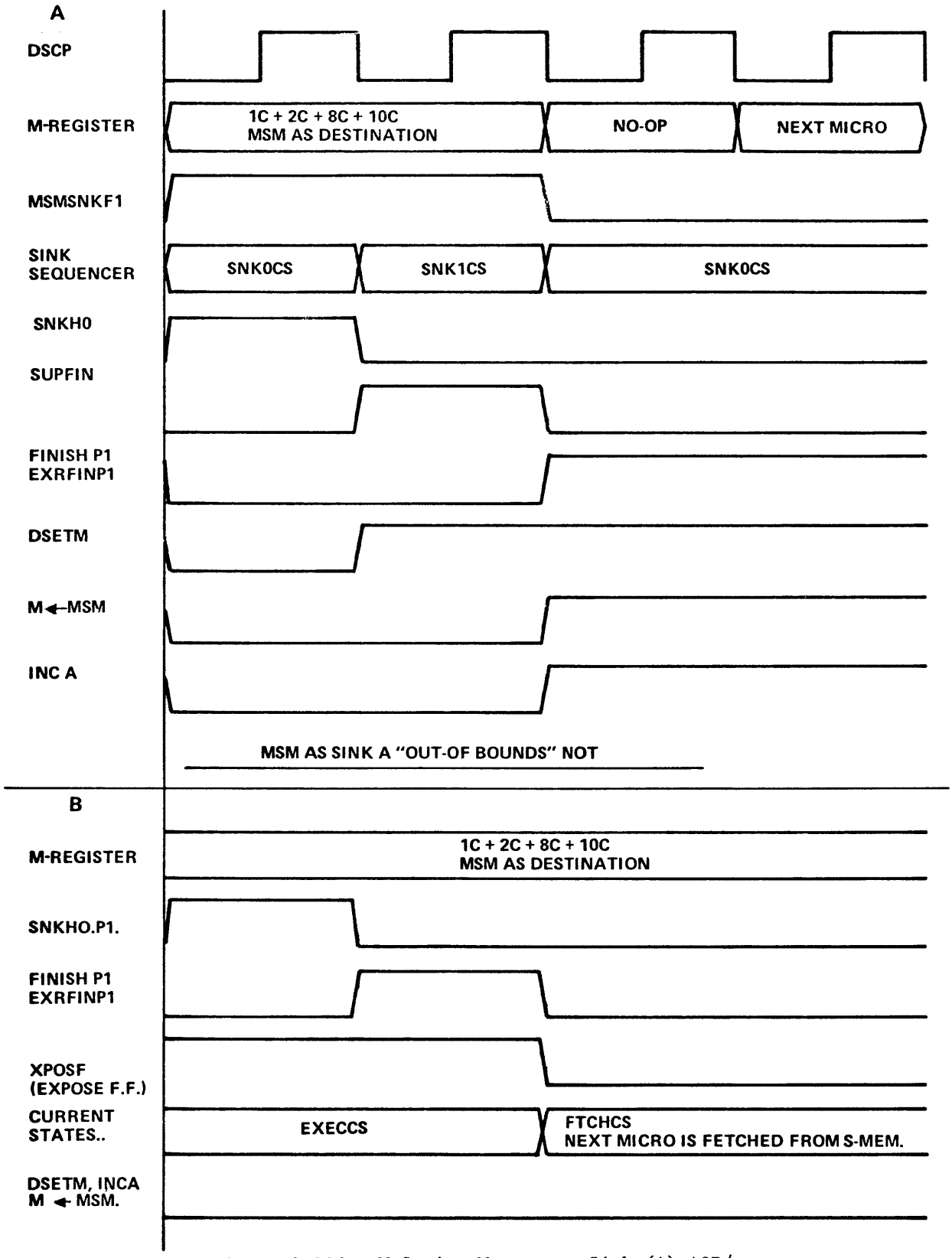


Figure 2-106. M-String Memory as Sink (A) AOB/.
(B) AOB.

EXECUTION OF A 1-CLOCK MICRO OPERATOR

Example: 8C when L-register is the sink.

This micro is executed in one clock period. After the 8C micro is set in the M-register, decoding begins and the M-register is sourced to the MEX as shown in figure 2-107.

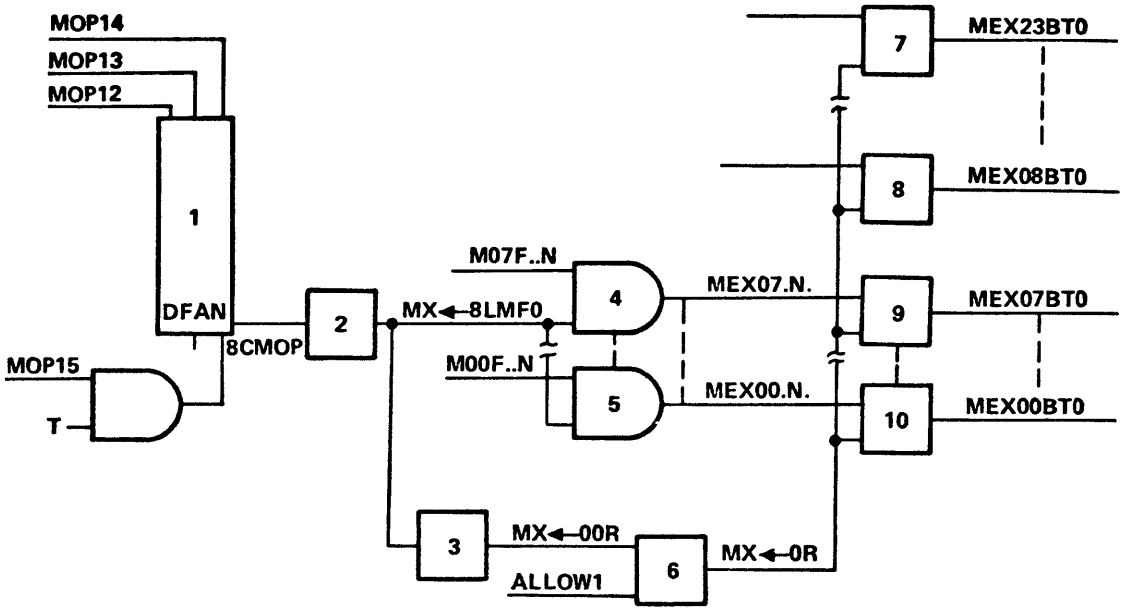


Figure 2-107. Source the M-Register to MEX

The DFAN 1 has only the MOP 15 true, which decodes 8C MOP true, and, through buffer 2 enables MX← 8LM true. This level true gates M00F through M07F by gates 4 and 5, onto lines MEX00 through MEX07. The lines MEX08 through MEX23 are not enabled by the MX← 8LM and are zeros. MEX00 through MEX07 are then gated to the main exchange by MX← 0R, generated by buffer 6 and going to all 24 MEX buffers shown (partially) as 7 through 10 in figure 2-108. At the same time, the sink register (1-register) is placed in D-set mode to receive the data from the MEX. The data is gated to L via DFANs 11 and 12 and gates 13 and 14 (refer to figure 2-108).

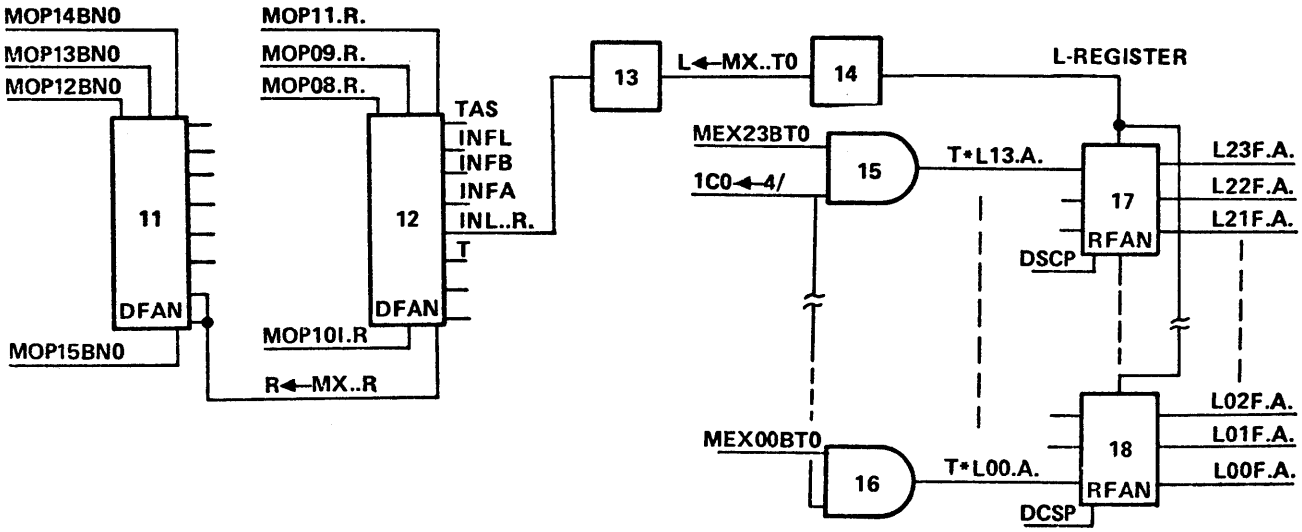


Figure 2-108. Sinking to the L-Register

Gating from the MEX to the L- and T-register inputs is normally enabled except when the registers are used as 4-bit sinks. Note that the levels FINISH, INCA, DSETM, and M ← MSM all remain true during the present clock period, and, at the trailing edge of the clock, the following actions take place:

a. The L-register receives the data.
b. The M-register receives the next micro.
c. The A-register is incremented by 1.
d. The MSM receives the address of the second micro following the one just completed.

One-clock 8C micro timing is shown in figure 2-109.

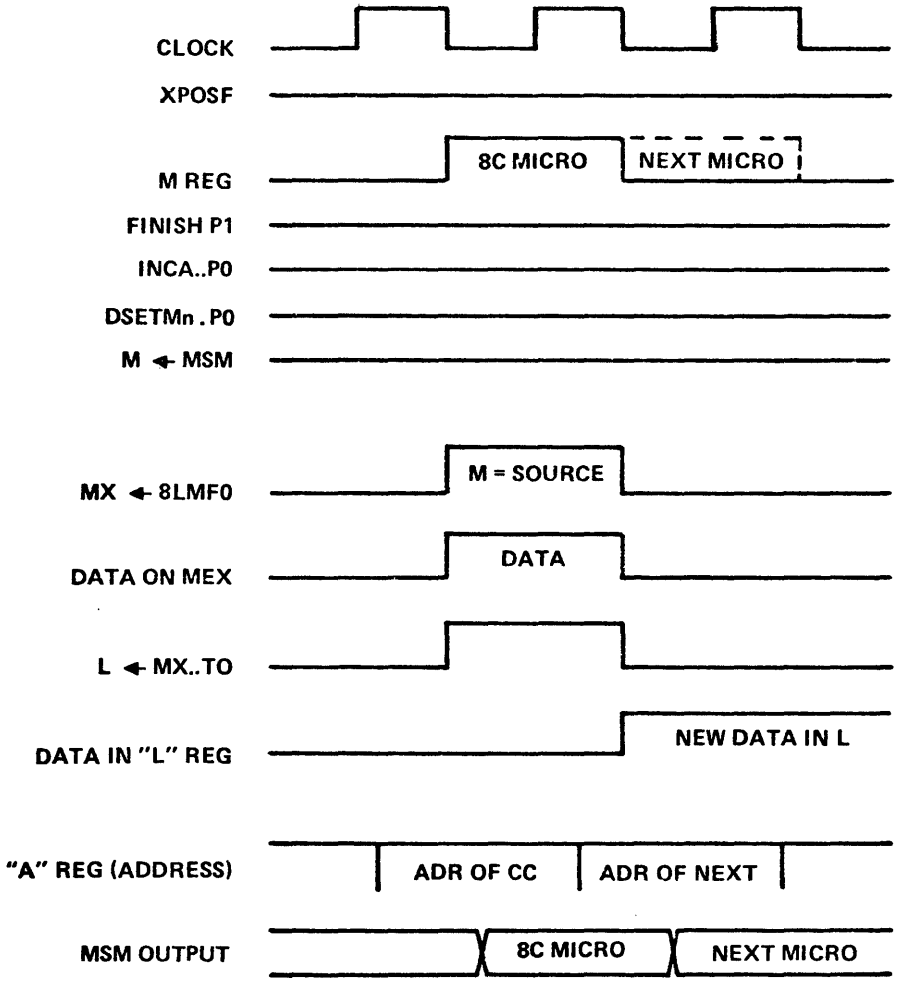


Figure 2-109. 8C Micro Timing Chart (One Clock Micro)

EXECUTION OF A 2-CLOCK MICRO OPERATOR

Example: 8C when A-register is the sink. (Refer to figures 2-110 and 2-111.)

This micro is executed in two clocks. During the second clock period a no-op is executed to allow the micro instruction designated by the new A-register address to be fetched from M-string memory. When the A-register is selected as sink by the DFANs (refer to figure 2-100) the following happens:

a. The A-register is placed in D-set mode to receive the new data (address). The level DSETA.PO comes true via gate 17, buffer 13, and buffer 11.

b. The A-register inputs (00 through 13) are enabled (to receive the MEX data) via gate 18, and buffers 14 and 10. Note that the lines MEX00 through 03 are lost, because the A-register receives only the word address (MEX lines 04 through 17).

c. The jump level is generated. When jump comes true the following happens:

1. The signal M←MSM is removed, causing the M-register inputs to be zeros.

2. The DSETM is held true, via gate 4, causing the M-register to receive zeros. A no-op is forced.

3. The level INCA is removed and the A-register is not incremented because it receives a new address.

At the trailing edge of the first clock a no-op operation starts. The M-string memory receives the new address, and 70 nanoseconds later the MSM data (new micro) is present on the M-register input. At the trailing edge of the second clock the new micro begins executing.

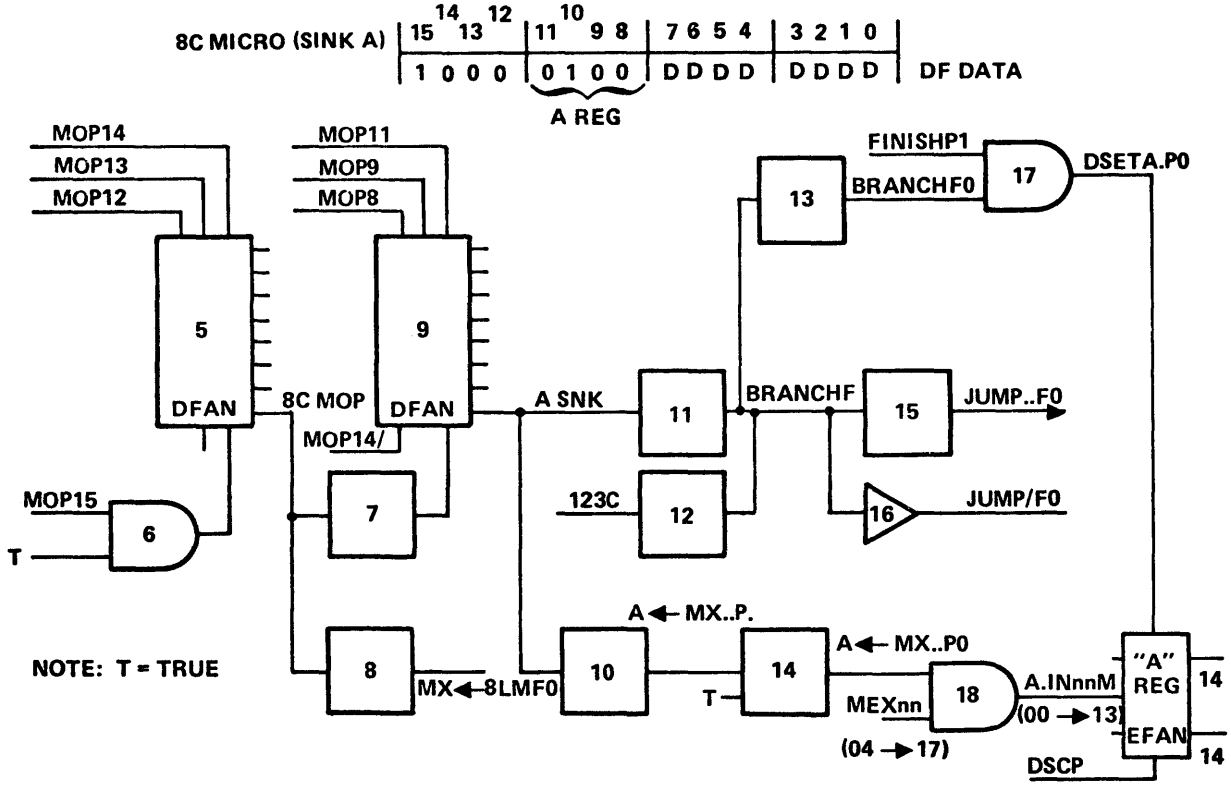


Figure 2-110. A-Register Sink Enable

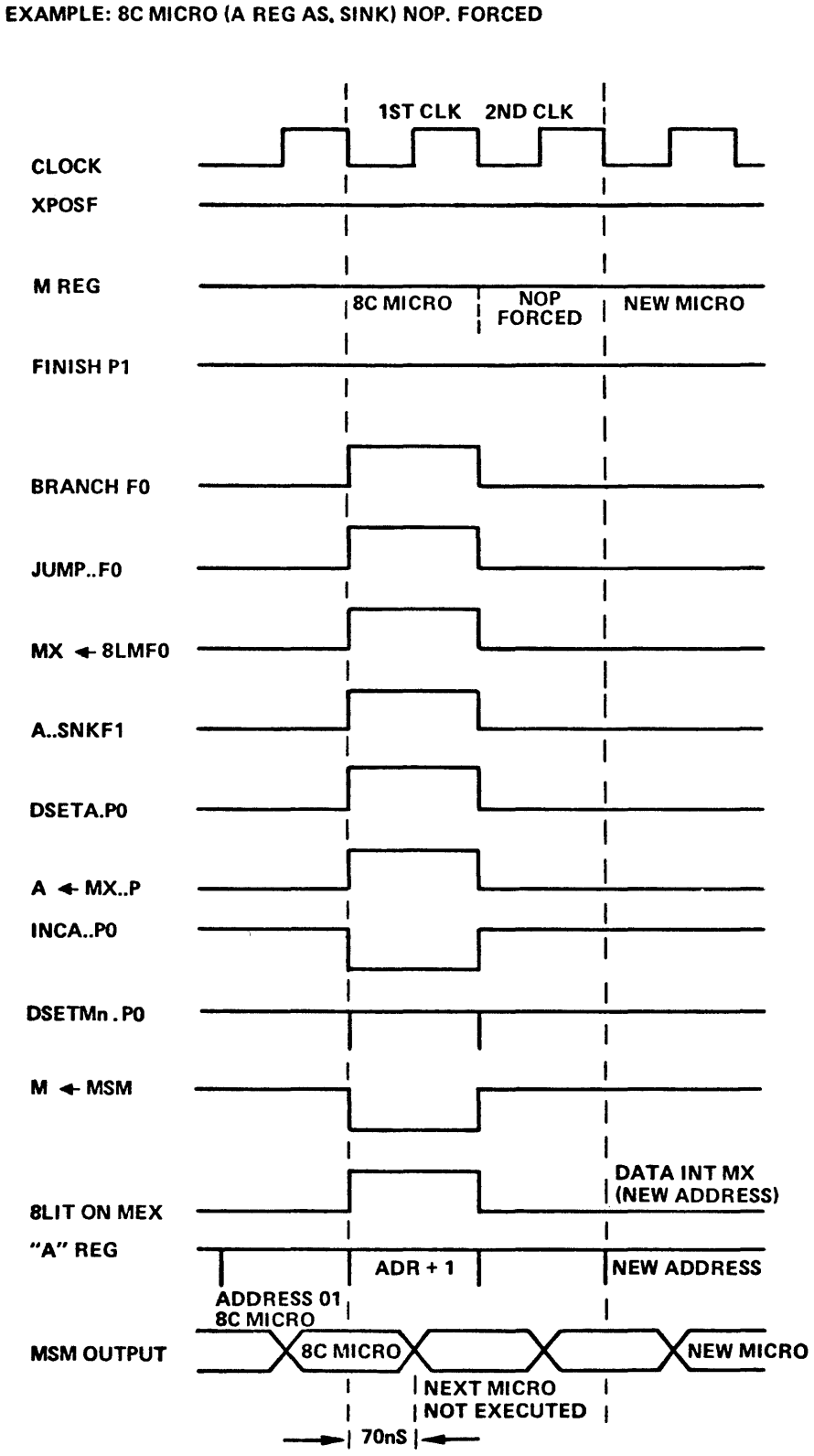


Figure 2-111. 8C Micro Timing Chart (Two Clock Micro)

EXECUTION OF AN N-CLOCK MICRO OPERATOR

Example: 3F - normalize X with FL = 3 (refer to figures 2-112 and 2-113).

This micro instruction takes 4 clock periods to execute. The 3F MOP-decode generates the signal NORMALP. through DFAN1, inverter 2, and gate 3 which enables SUPFINP1. Referring to figure 2-98, the level SUPFINP1 causes FINISHP1 and EXRFINP1 to go false through DFAN 14 and DFAN 17, respectively. As long as EXRFINP1 is false the levels DSETM, M MSM, and INCA remain false, disabling the M-register from receiving the next Micro.

At the trailing edge of each clock, the X-register is shifted left by one position and the FL-register is decremented by one. When FL = 0 the level NORMALP1 goes false, FINISHP1 and EXRFINP1 become true which, in turn, enables the levels DSETM, M←MSM, and INCA again. The next micro operator can now be transferred to the M-register and the A-register is incremented.

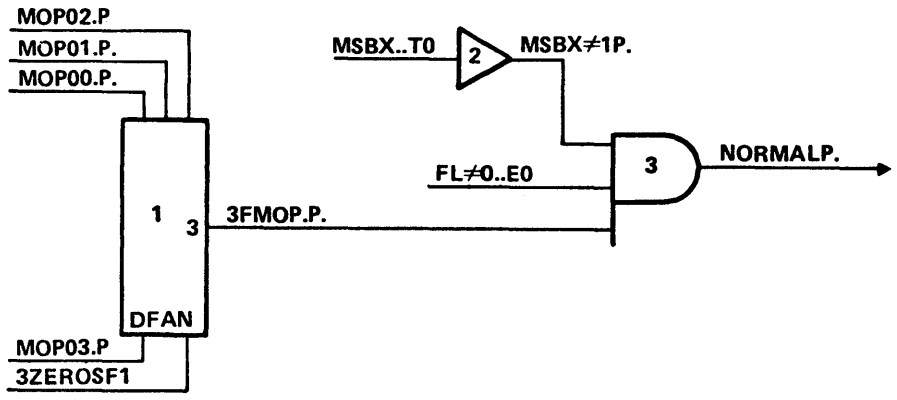


Figure 2-112. The 3F Normalize Micro Control (n CLK Micro)

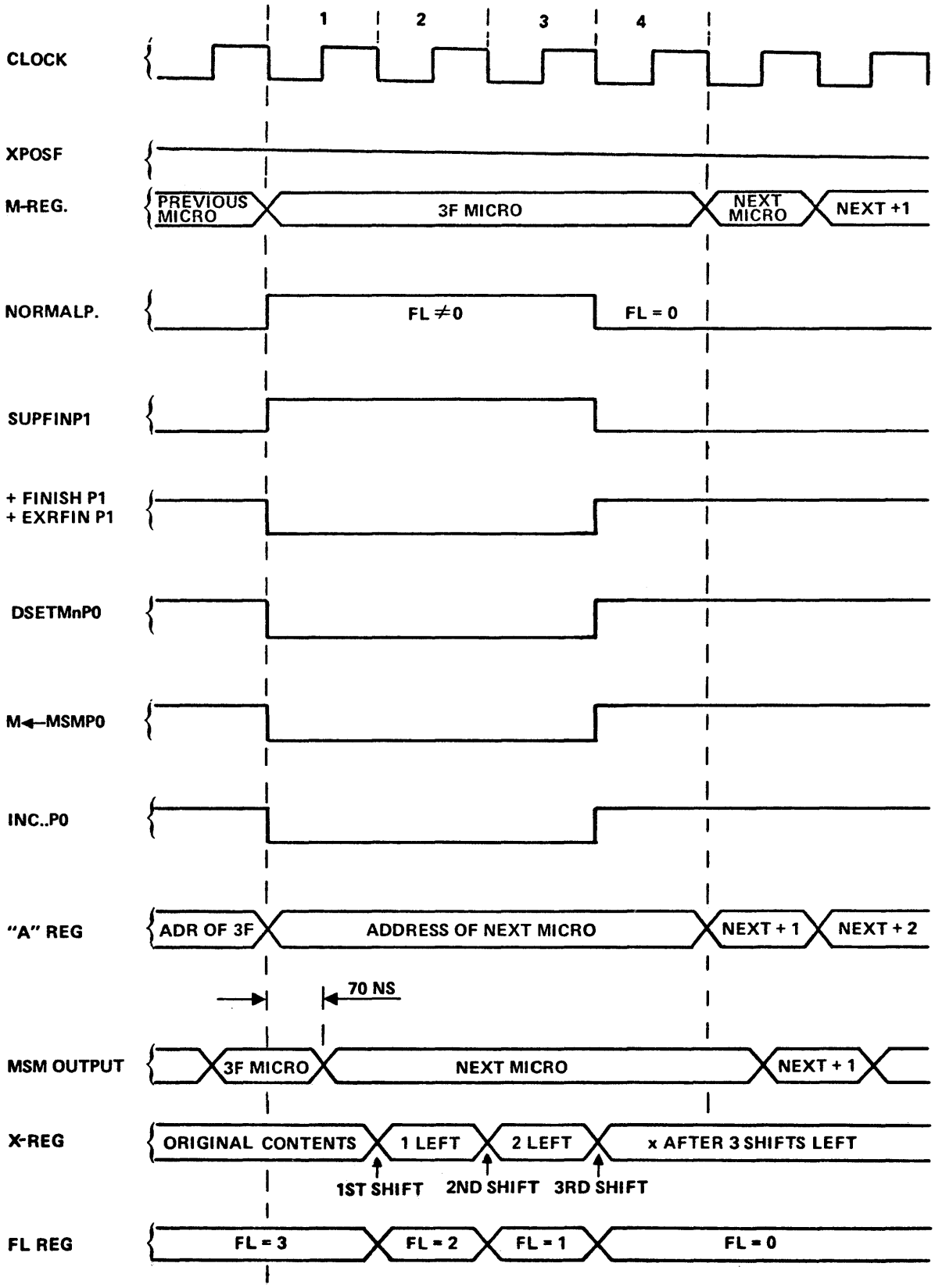


Figure 2-113. 3F Micro Timing Chart (n Clock Micro)

MULTIPLE FUNCTION MICRO EXECUTION

To accomplish certain software requirements, provisions have been made for the execution of several micro operators which either perform a number of different operations, or do repetitions of the same operation. Included in this group are the 7E exercise MSM, 2F overlay M-string, and 4F bind micros. Execution of the multiple function micros requires modification of the basic micro and address control signals, and this task is performed by a special multiple function micro sequencer circuit (figure 2-114). This sequencer overrides the normal control circuits during the execution of multiple function micros, serving as an additional means of producing the required logic activiation signals.

The sequencer consists of an encoder, three flip-flops, and a decoder, plus a number of associated gating elements. Its outputs, which are used directly and indirectly to produce significant control signals, consist of the following:

Signal	Description
MFIDCSP.	Idle current state. In this state the sequencer is waiting for a multiple function micro to be decoded, and exerts no control over micro execution.
MFINCSP.	Initial current state. Control signals are generated to condition the processor for the operation which is to follow. This may include what would normally be a single function micro operator, such as "move A to TAS" which is done as part of the 2F overlay M-string micro.
MFSRCSP.	Source current state. Causes a read to be performed from the appropriate source, as part of the executing micro.
MFSKCSP.	Sink current state. Causes a write to be performed in the appropriate sink, as part of the executing micro.
MFEXCSP.	Exit current state. Completes the executing micro (including restoration of previously existing conditions), and returns control to the normal execution logic.

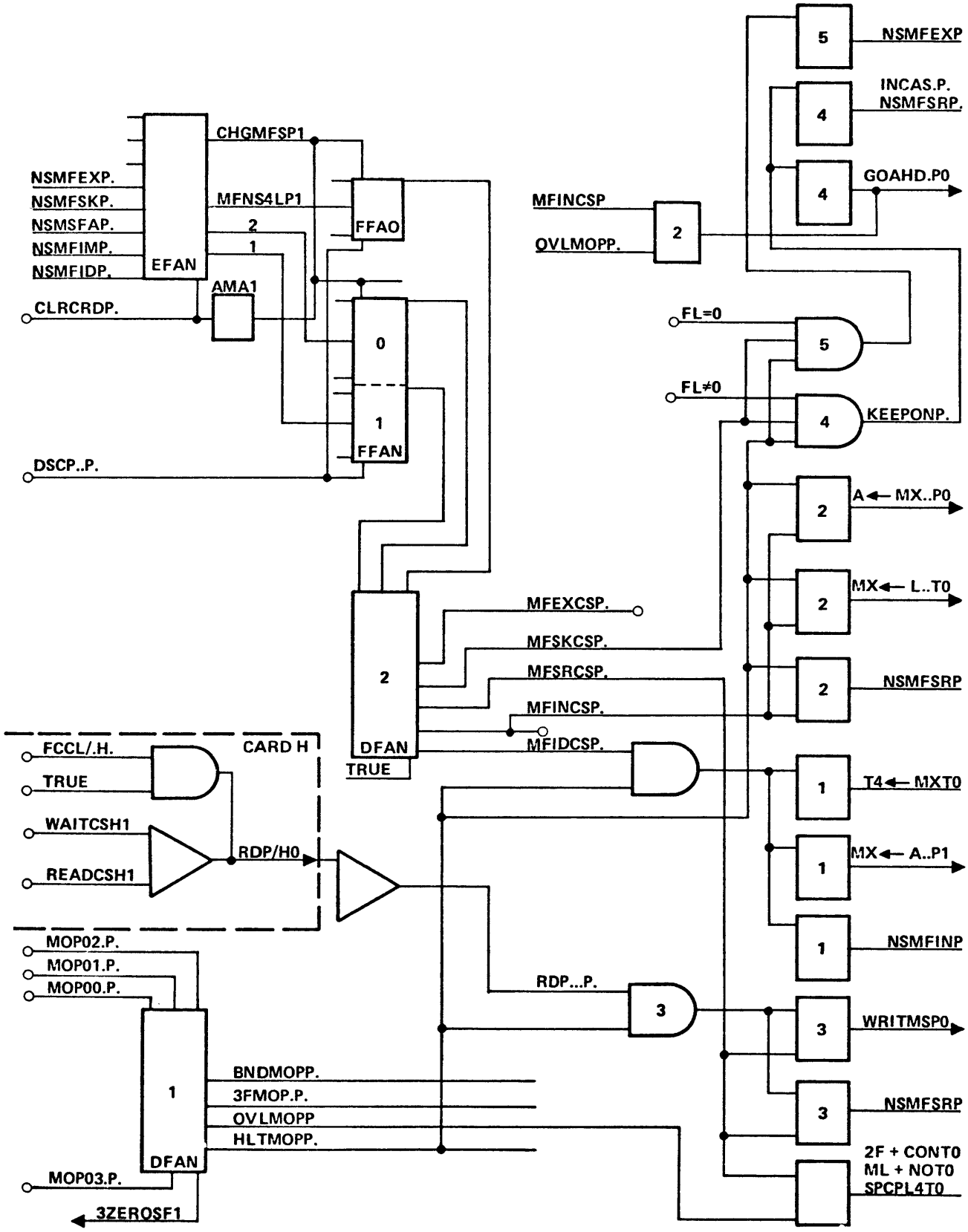


Figure 2-114. The Multiple Function Micro Sequencer

In operation the sequencer monitors the status of numerous control lines (including the outputs of micro decoding and its own outputs) to derive the "next state" signals. These are fed to the inputs of an EFAN in ascending order from NSMFIDP. (next state idle) through NSMFEXP. (next state exit). The binarily-encoded position of the most-significant next state signal which is true is set into the three flip-flops (to ensure a constant output for at least one clock pulse), and from there to a DFAN which reduces the code to the appropriate single level multi-function current state output signal.

Since the operation of the sequencer is determined in part by intermediate results, it is best explained by citing specific cases. Therefore, refer to the descussion of the 2F overlay micro (following) for a detailed operational description.

Execution of the 2F Overlay M-String Micro

The 2F micro (overlay M-string from S-memory) is used to transfer strings of micro operators from storage in S-memory to M-memory for use. Refer to figures 2-215, 2-216, and 2-217. It is a multi-clock operation which involves repeated reads from S-memory, with writes of this data into sequential addresses in M-string memory. In operation, 2F utilizes the following parameters:

The starting M-string address is taken from L.

The starting S-memory address is taken from FA.

The length of the data overlay (in bits) is taken from FL.

Execution of the 2F micro proceeds as follows:

a. Move A to TAS (save the contents of A).

b. Move L to A.

c. 16 bits are read from S-memory and written into MSM. The FL register is then decremented by 16, FA is incremented by 16, and A is incremented by 1.

d. Step c is repeated until FL is equal to zero, at which point the micro terminates by moving the contents of TAS back to A.

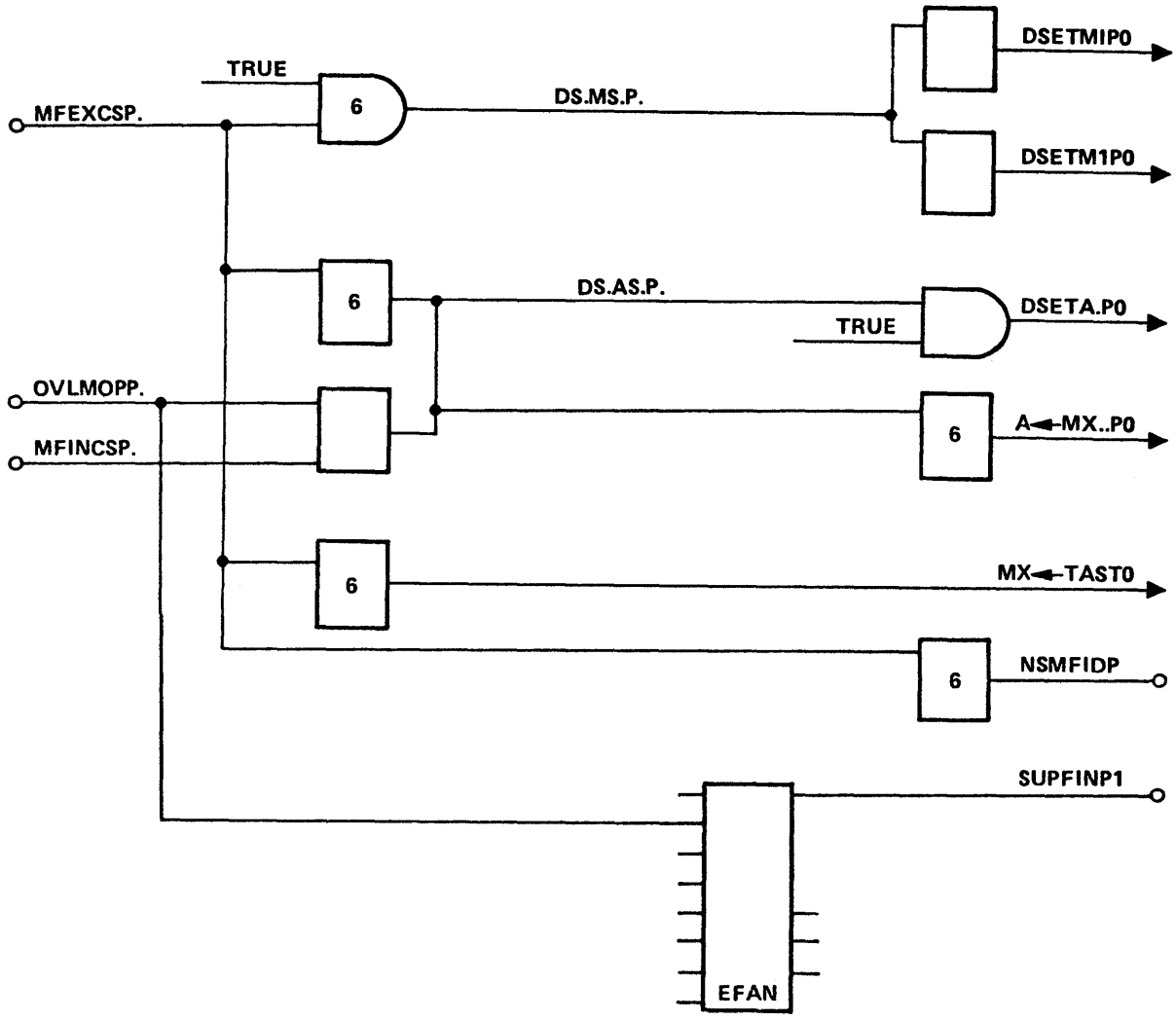


Figure 2-115. The Overlay Micro Control Levels

The multiple function sequencer (figure 2-114) is normally in the idle state (MFIDCSP. true). When the overlay micro (OVLMOPP) is decoded, the suppress finish signal (SUPFINP1) is forced true immediately to prevent completion of the micro. With MFIDCSP. and OVLMOPP. both true, all buffers identified by a 1 are enabled, causing the contents of A to be moved to TAS (A is "saved" to allow return of the processor to the previously existing condition upon completion of the overlay micro). At this time NSMF1NP. also comes true, which causes the sequencer to advance to MFINCSP. (initial state) at the trailing edge of the next system clock pulse.

During the initial state, the buffers numbered 2 are enabled, causing the contents of L to be moved to A. The level GOAHD.P0 (go ahead) is forced true at this time, and is sent to the PA/PD interface control logic where it causes a memory cycle to be initiated. NSMFSRP. comes true to cause the sequencer to advance to the source current state at the next clock. During the source state, the logic awaits the first FCCL pulse from the port interchange. When this signal is received, the contents of FA are gated to the MEX and then to S-memory to serve as the address. FCCL also causes the PA/PD interface logic to generate DNB3P.T0, which enables counting FA up and FL down. (The levels 2F+CONT0 and FL+-MOT0 are true at this time; see card P.)

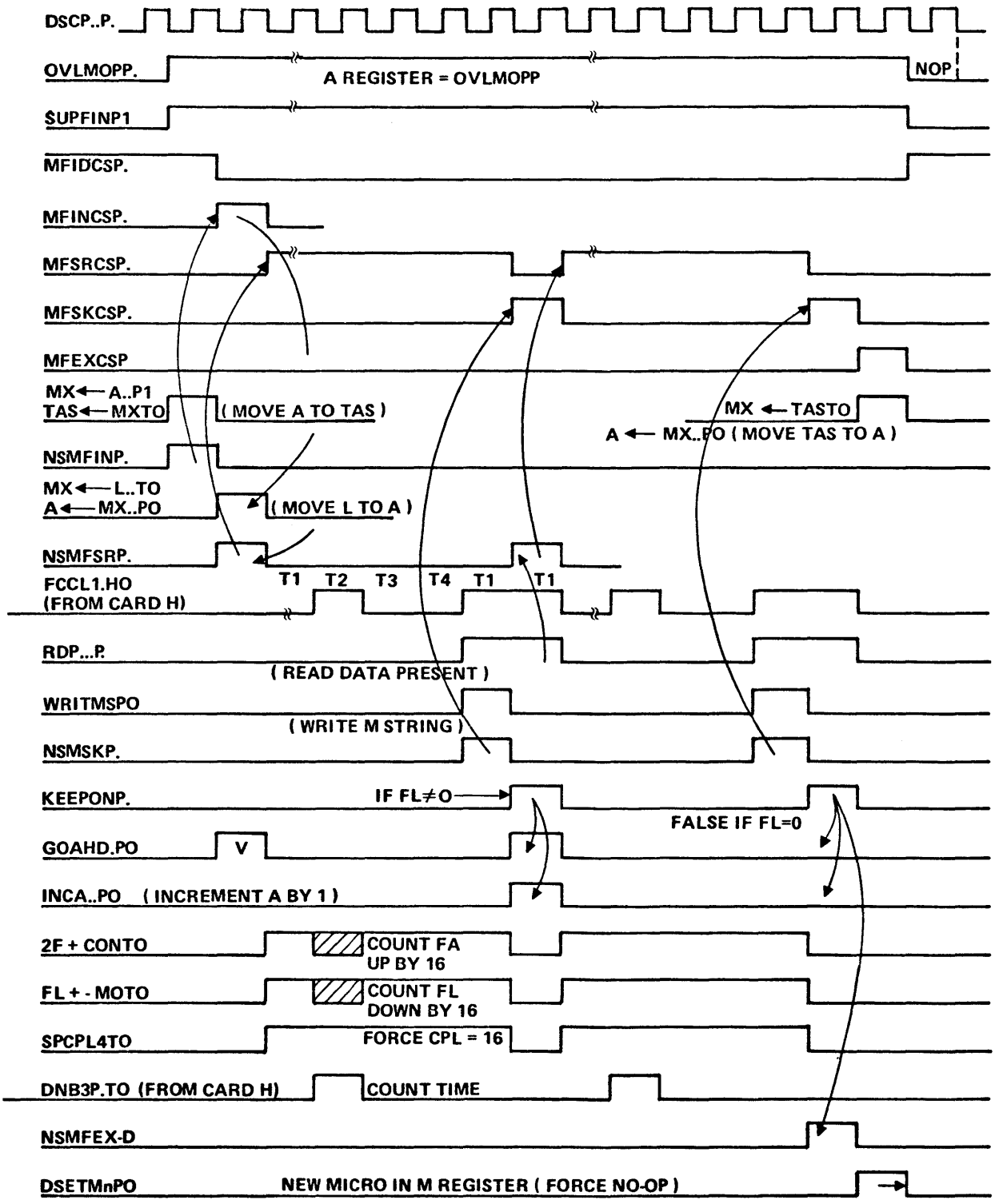


Figure 2-116. The Overlay Micro Timing Chart (Processor)

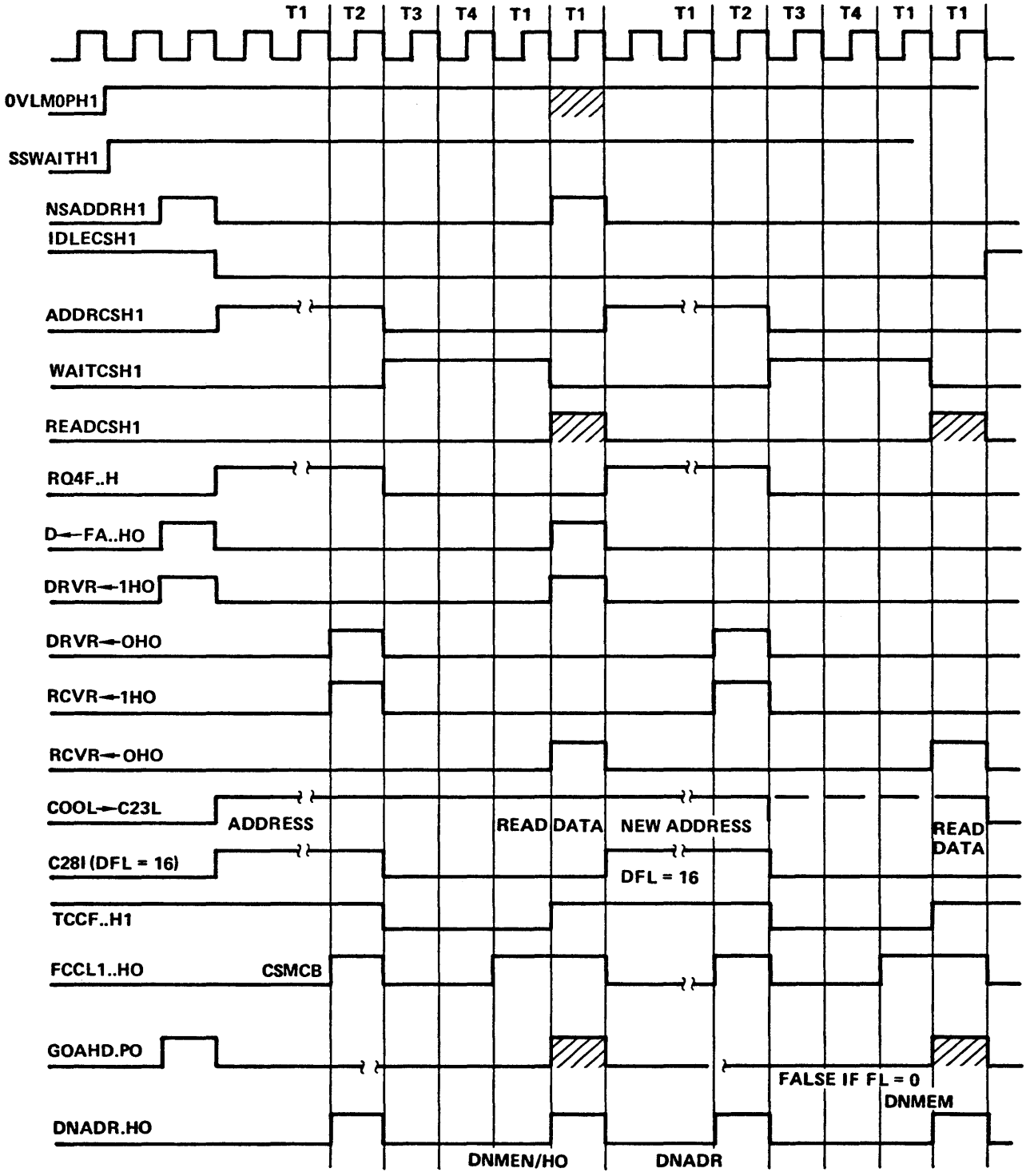


Figure 2-117. The Overlay Micro Timing Chart (PA/PD Interface)

After the address has been accepted by S-memory, the PA/PD interface control waits for the second FCCL pulse, which indicates DATA present on the main exchange. The two-clock-long FCCL pulse enables the level RDP...P. (read data present) to be true on card P. RDP...P. goes also to the multiple function sequencer, where it generates WRITMSPO and NSMFSKP. (buffers numbered 3, figure 2-114). WRITMSPO enables writing in M-string memory, and NSMFSKP. allows the sequencer to advance to the sink state (MFSKCSP.).

During the sink state, if FL≠0 is true, the gate and buffers numbered 4 (figure 2-114) are enabled. At this time the A-register is incremented by 1 (INCAS.P.) and NSMFSRP. allows the sequencer to return to the source state. GOAHD.PO is sent to the PA/PD interface control to begin a new memory cycle.

If in the sink state and FL=0, then gate 5 is enabled, producing NSMFEXP., which causes the sequencer to go to the exit state (MFEXCSP.) at the next clock, finishing the micro. In this case GOAHD.PO stays false, and another memory cycle is not initiated.

When in the exit state the gate and buffers numbered 6 (figure 2-115) are enabled. This causes the M-register to be placed in the D-set mode (DSETM.PO). Because SUPFINP1 is still true, M←MSM.PO is disabled, preventing the addressed location in MSM from being gated to M for execution. Therefore, zeroes are loaded in M to force a one-clock no-op. At this time, TAS is transferred to A to address the next in-line micro (following the 2F currently executing). NSMFIDP. coming true causes the sequencer to return to the idle state.

Timing for the 7E exercise MSM micro and 4F bind micro is shown in figure 2-118.

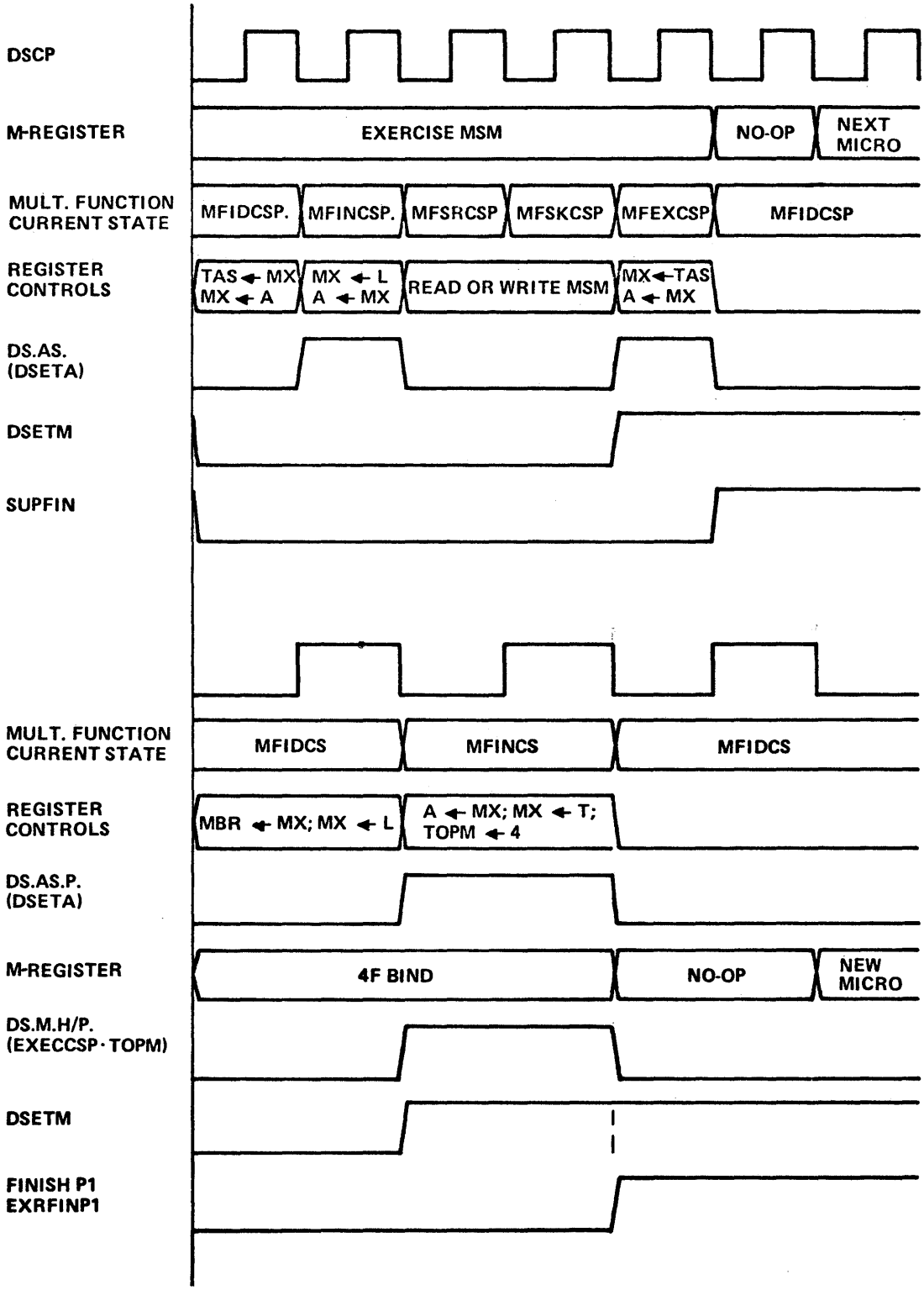


Figure 2-118. Multiple Function Current States Timing Chart (Exercise and Bind)

CONCURRENCY

In order to achieve maximum utilization of the processor at all times, certain selected micro instructions can be executed while a basic 3 (7C read, 7C write, or 2D swap) memory cycle is in process. The micro instructions are divided into two sets: a non-concurrent set and a concurrent set.

The non-concurrent set must wait until the memory cycle is completed before proceeding with its execution. The concurrent set can overlap its execution with the memory cycle. The overlap starts with the clock period following address accept and can extend up to or beyond the time that memory data is accepted.

If the concurrent micro is another memory micro, the overlap only encompasses base-limit checking for it. The actual memory request for such a concurrent micro is made during the clock period following acceptance of memory data for the original basic 3 micro. Except for the 8D micro, all one-clock micros and two-clock branch micros are followed by a no-op inserted by the hardware in order to allow time to examine the next in-line micro from the M-string memory. The micro operators comprising the concurrent set are: 123C, 8D, 4E, 6D, 9D, 5E, 7D, 3E, 7C, and 2D.

Concurrency States

The concurrent operation is governed by a sequencer which generates eight defined states during which the various functions required are performed. Refer to figure 2-119. Following is a brief description of the function of each state:

CONICSP. Concurrent idle current state. The logic remains in this state until a basic 3 micro comes up for execution. The logic also returns to this state when FINISHP1 comes true at the end of the basic 3 micro execution time. GPCLRBHO forces the idle state.

CONDCSP. Concurrency defer current state. This state is entered if the exposed micro is not a concurrent micro or if the concurrency is not allowed (MTR + AOB + HALT). The XPOS flip-flops are reset and a NOP is forced whenever this state is true.

CONACSP. Concurrency address current state. This is the first state entered from the idle state. The logic waits in this state for DNADR.HO which is indicated by the first FCCL signal from the port interchange (address accepted). At this point the basic 3 micro is accepted by the port interchange which proceeds with its execution independent of the processor. At this time, a concurrency micro can be entered into the M-register for execution.

CONICSP. Concurrency first current state. The logic is in this state during the first clock period of the execution of an exposed concurrency micro. The micro presently in M-register is also examined for its type in order for the logic to proceed to the next proper concurrency state. (Basic 3, one-clock micro, two-clock micro.)

CONJCSP. Concurrency jump current state. This state generates a no-op to create a time delay whenever a jump function is performed, in order to allow the address register (A-register) to change its contents (increment or D-set).

CONNCSP. Concurrency no-op current state. This state forces a no-op after every one-clock or two-clock branch micro to allow time for the evaluation of MSM read data (next in-line micro examination).

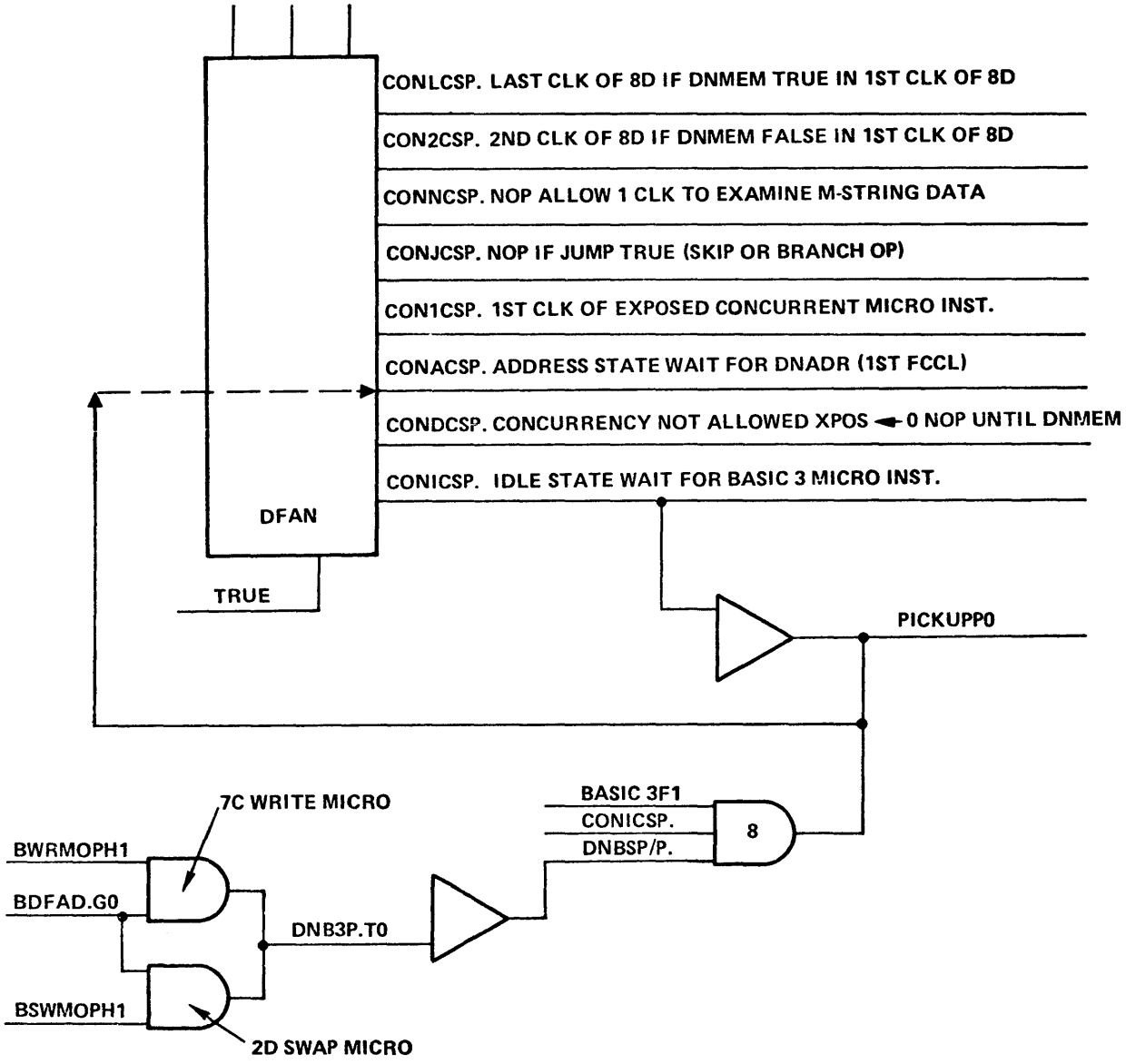


Figure 2-119. The Concurrency State Sequencer

Concurrency Flow

Under normal conditions the sequencer is set to the idle state. Refer to figure 2-120. Flip-flops A, B, and C (figure 2-121) are reset. Upon decoding a basic 3 micro, an address check is performed immediately by comparing the LR and the BR registers with the FA register (figure 2-122). The address compare is performed on a continuous basis; however, it is monitored only during the memory write operation if the CD-bit 2 is false. If the address check is bad in a write operation, FINISHP1 is not suppressed, and the 7C write is discontinued. If the address check is good, or if the basic 3 micro is a read operation, the gate 8 (figure 2-121) is enabled and PICKUPP0 comes true. On the next clock the concurrency address current state is set. With the level PICKUPP0 true, SUPFINP. is set, which, in turn resets FINISHP1. PICKUPP0 also sets the signal OR ← MSM., which allows the M-string read data to be gated to the main exchange for concurrency micro examination. Refer to figure 2-123.

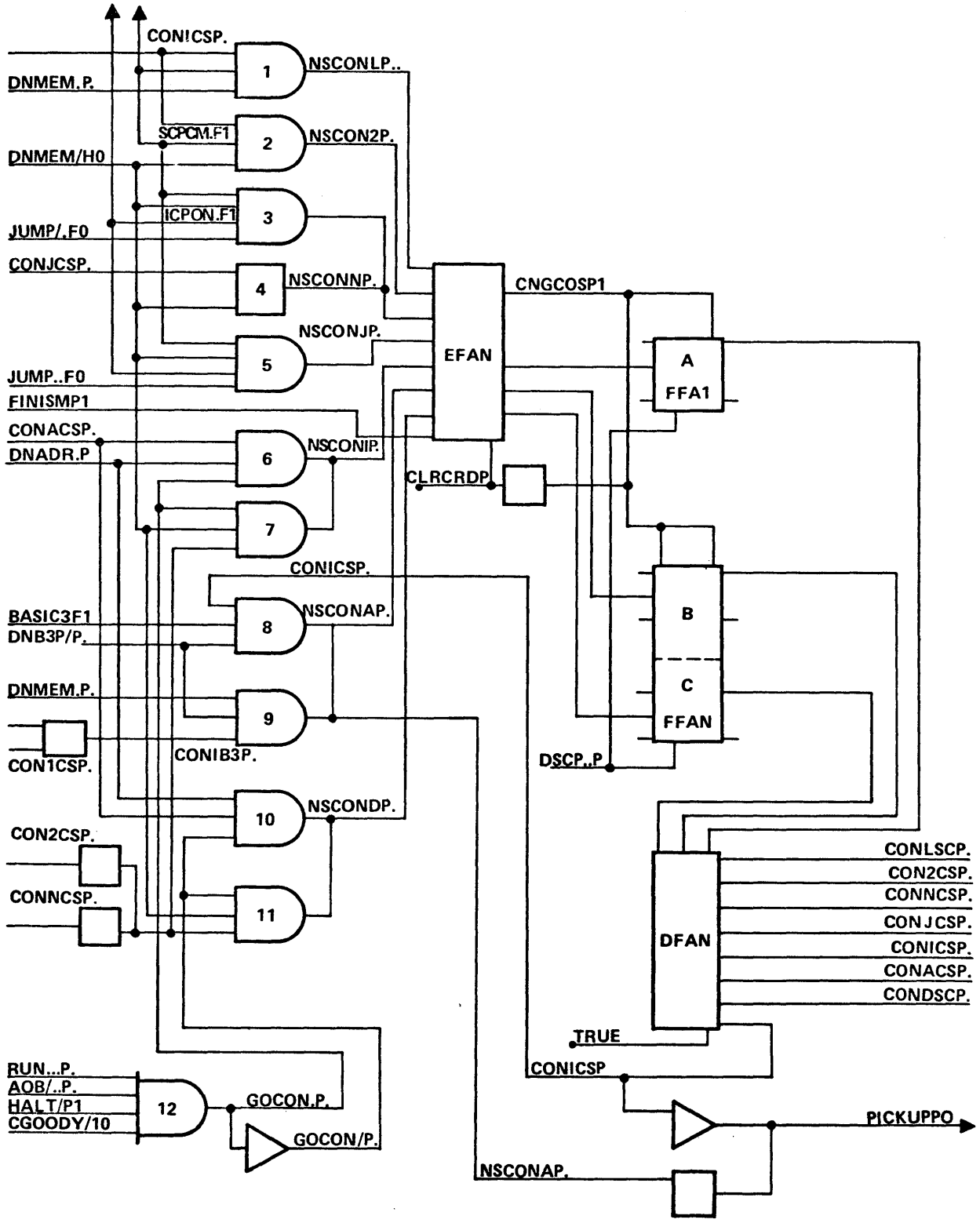


Figure 2-121. The Concurrency Sequencer Control

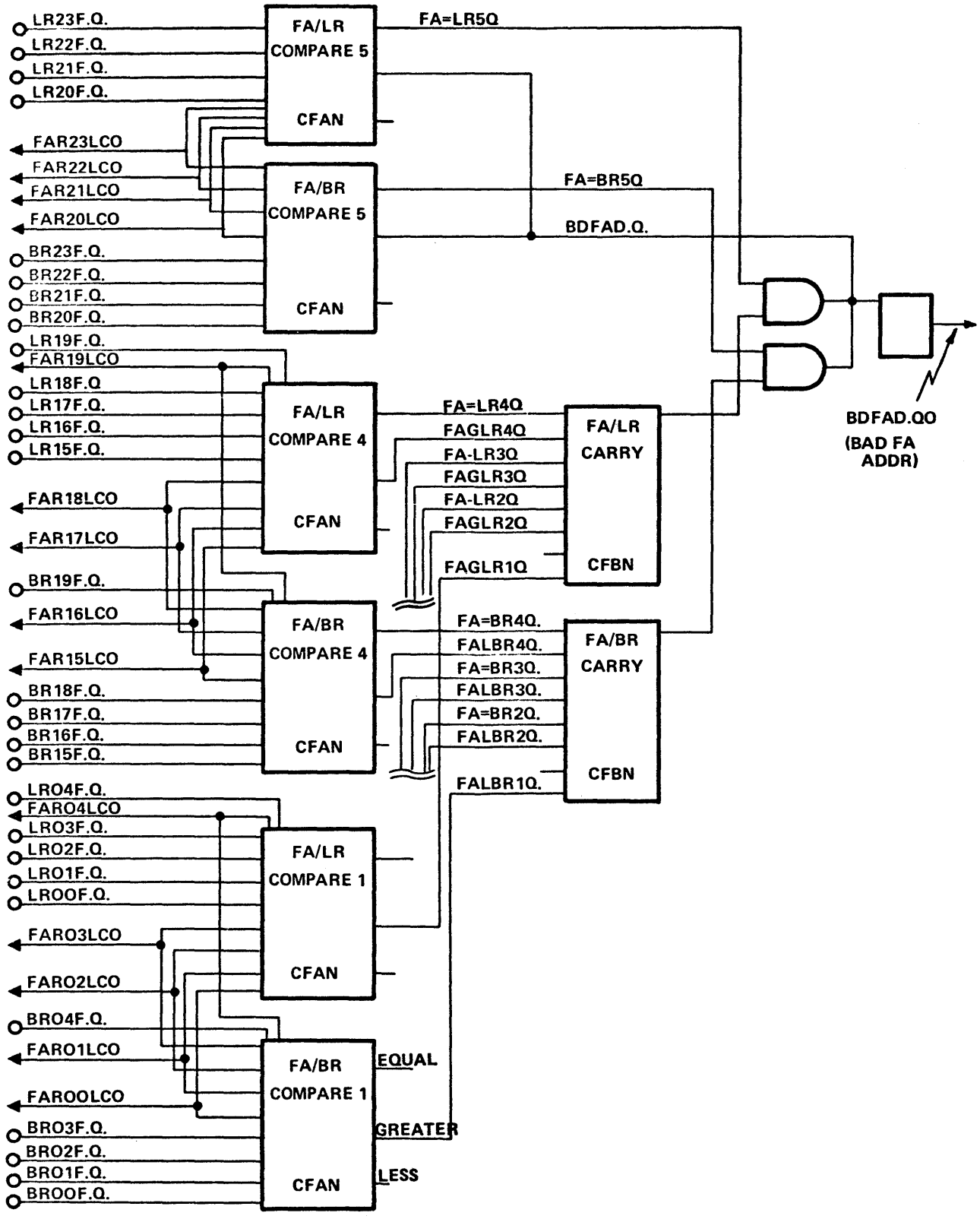


Figure 2-122. The Base and Limit Check Logic

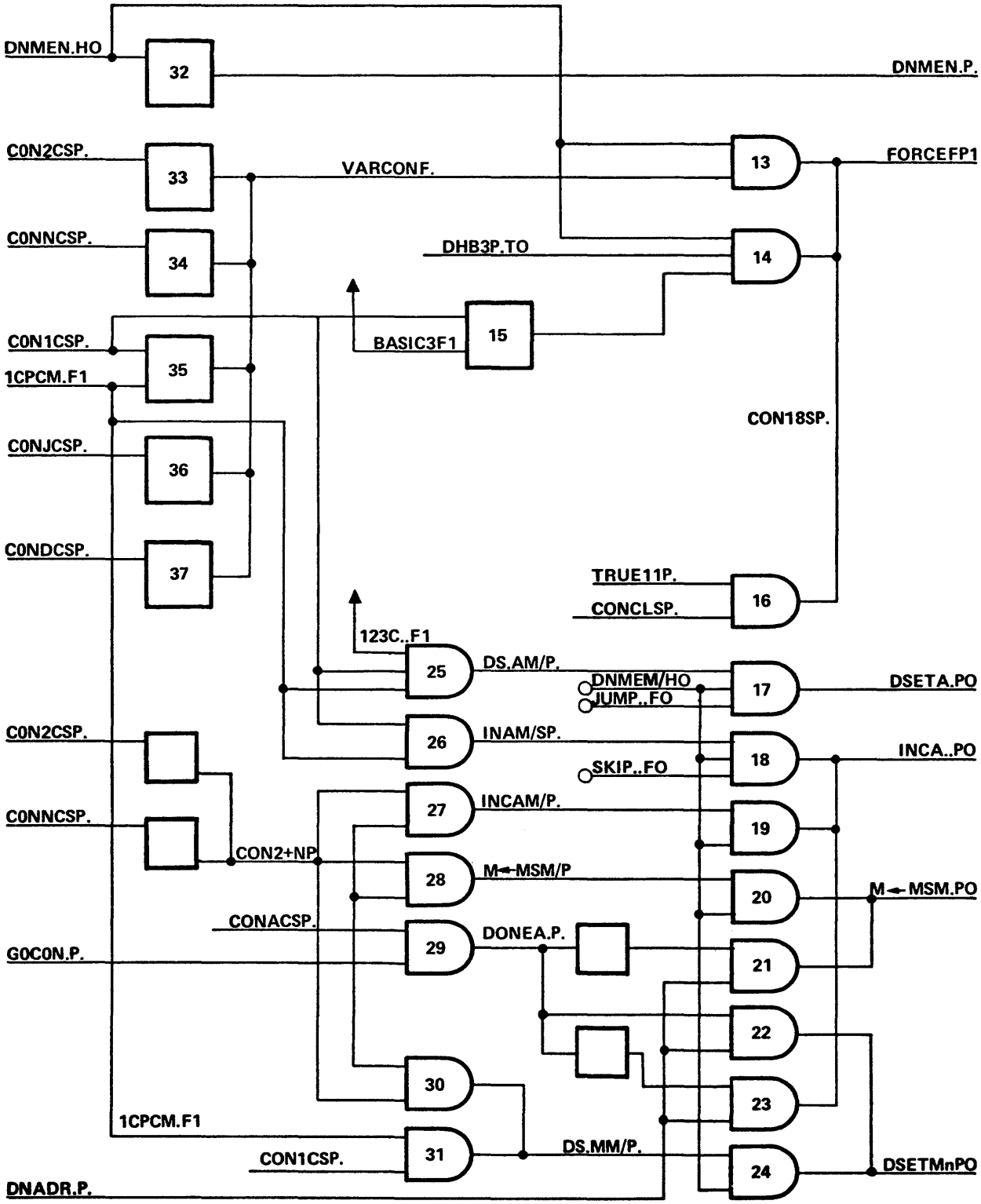


Figure 2-123. MSM, M, and A-Register Controls

If the signal CGOODYNO (figure 2-124) comes true, provided the logic is in run mode with the A-register not out of bounds, and no halt condition is present, gate 12 is enabled and GOCON.P. (go concurrency) comes true. The logic is now waiting for DNADR.HO (done address). If for any reason gate 12 is disabled, GOCON/.P comes true.

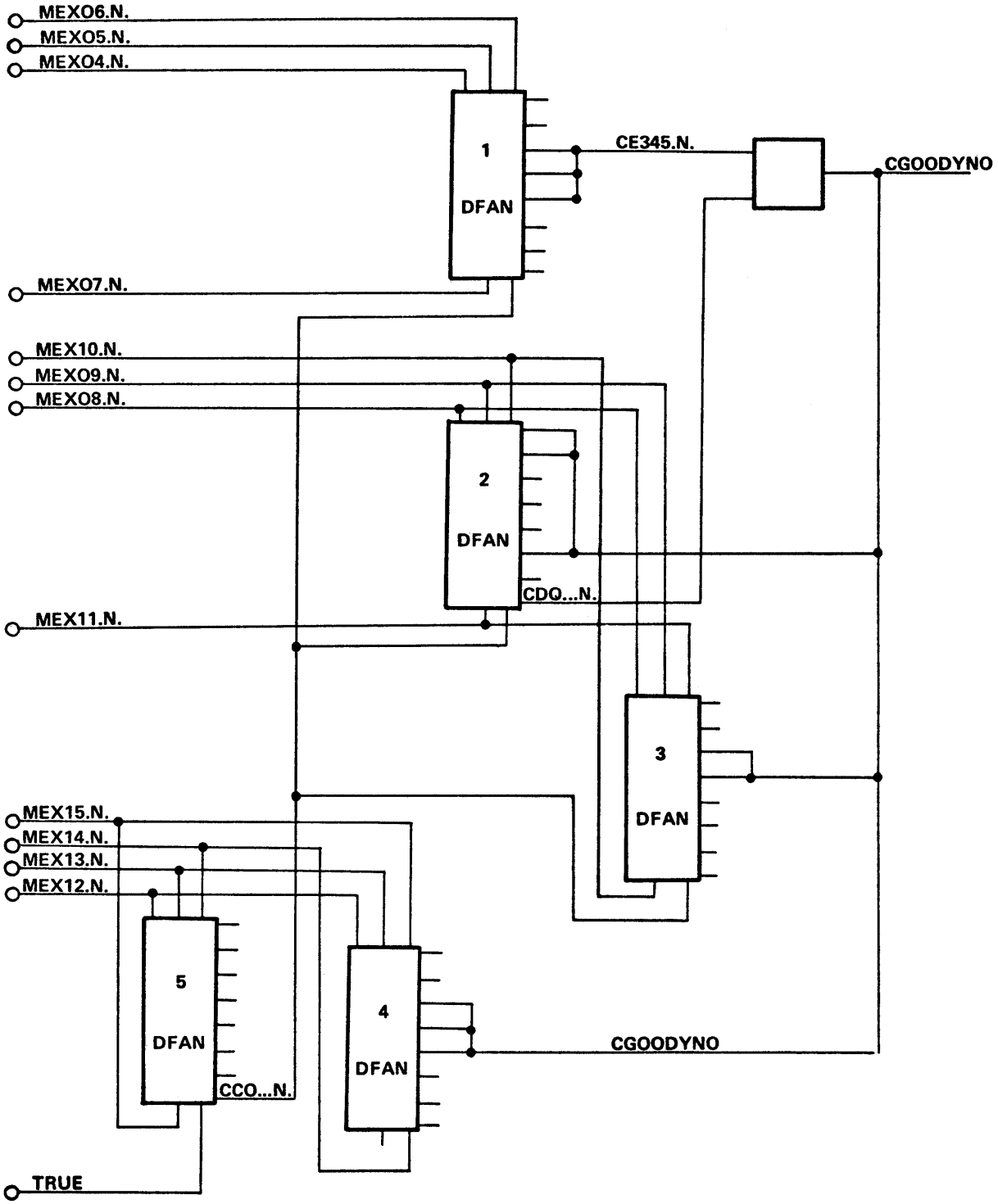


Figure 2-124. Concurrency Micro Validity Checker (Goody)

With DNADR.HO and GOCON/.P true (concurrency not allowed), gate 10 is enabled and the concurrency defer current state is set and the XPOS flip-flops are reset. With the XPOS flip-flops reset, zeros are forced to the MOP lines and the processor executes no-ops until DNMEM.HO is true, which, in turn, forces FINISHP1 (FORCEFP1) through gate 13. The moment FINISHP1 is true the sequencer resets, the concurrency idle current state is set and PICKUPPO is reset. FINISHP1 also sets the XPOS flip-flop again.

With DNADR.P. and GOCON.P. true (concurrency allowed) the M-register is put in D-set mode. M MSM.PO becomes true via gate 29. At the same time gate 6 sets the concurrency first current state. During this state the first clock period of the concurrent micro is executed and its type is examined in order that execution may proceed through the proper logic path. There are three possible type of micros which can come up as the concurrent micro. (Refer to the concurrency flow chart, figure 2-120, for the following discussion.)

Basic 3

In this case the logic proceeds to the address check. If the address check is good, the logic waits for DNMEM.HO from the initial basic 3 micro being executed. When DNMEM.HO goes true, gate 9 is enabled which sets the concurrency address current state and the logic returns to point 1 (in the concurrency flow).

If the address check is bad, the logic waits for DNMEM.HO from the initial basic 3 micro being executed. When DNMEM.HO goes true, gate 14 is enabled, FORCEFP1 comes true, the concurrency idle current state is set and PICKUPPO is reset.

1-Clock Pulse Micro

If DNMEM.HO is true while a one-clock micro is being executed, gate 13 has forced FINISHP1 at the start of this micro, which means the next in-line micro is set in M-register and execution proceeds in a normal mode, unless a skip or branch was required. In this case, the signal JUMP/.FO is false for one-clock no-op, preventing M ← MSM.PO from coming true, which results in a no-op. At the end of the one-clock micro the state returns to the concurrency idle current state and PICKUPPO is reset.

If DNMEM.HO is false and the one-clock micro is not a jump, gate 3 is enabled and the concurrency no-op current state is set, performing the required no-op in order to examine the next concurrent micro (gate 31/24 enabled and M ← MSM.PO false). If DNMEM.HO is still false at this point, the logic proceeds to allow concurrency checking. However, if DNMEM.HO is true, FINISHP1 is forced true and normal micro timing resumes control.

If DNMEM.HO is false and the one-clock micro is a skip or branch, gate 5 is enabled and the concurrency jump current state is set. M←MSM/P. is false in this state, which forces a no-op. The skip function increments the A-register via gate 18 (INCA..PO) and the branch function puts the A-register in the D-set mode via gates 25 and 17 in order to receive the new address.

With the concurrency jump current state set and DNMEM true, FINISHP1 is forced, the concurrency idle current state is set and PICKUPPO is reset. With the concurrency jump current state set and DNMEM false, buffer 4 is enabled and the concurrency no-op current state is set to allow time for concurrency micro examination.

With the concurrency no-op current state set and DNMEM.HO true, the logic proceeds to point 4. With the concurrency no-op current state set and DNMEM.HO false, the logic proceeds to allow concurrency checking.

2-Clock Pulse Micro (8D Only)

If the micro examined during the concurrency first current state is an 8D micro and DNMEM.HO is false, gate 2 is enabled and the concurrency second current state is set during which the second portion of the 8D micro is being executed. If DNMEM.HO is true in this state the logic proceeds to point 4; however, if it is false the logic returns to allow concurrency checking.

If the micro examined during the concurrency first current state is an 8D micro and DNMEM.HO is true, gate 1 is enabled which sets the concurrency last current state in order to execute the second portion of the 8D micro. With concurrency last current state true, FINISHP1 is forced, and the logic proceeds to point 4. Concurrency timing for the 7D, 6D, 9D, and 1C micro string is shown in figure 2-125.

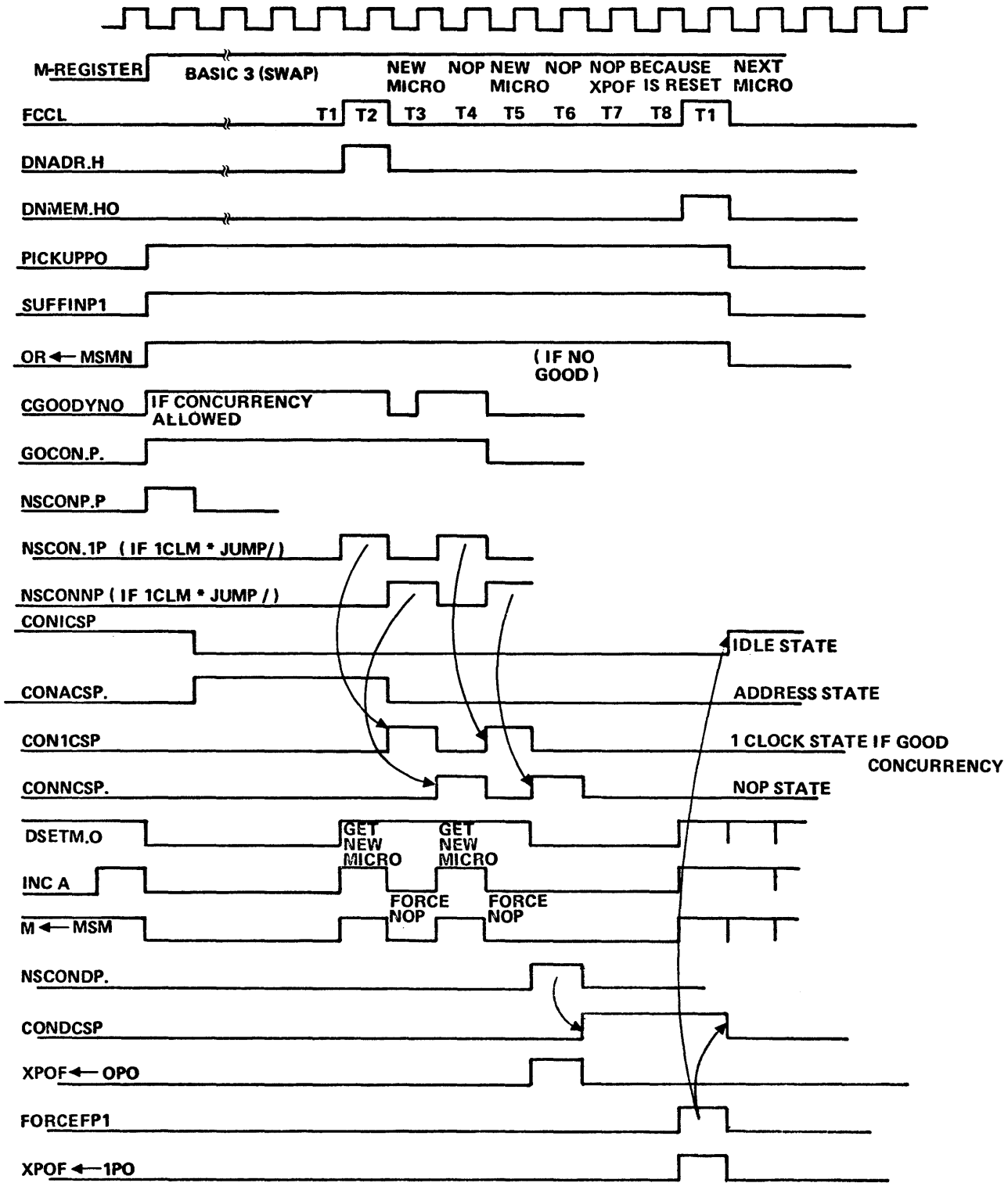


Figure 2-125. Concurrency Timing Chart
(Executing the following Micro String - 7D/6D/9D/1C)

MICRO INSTRUCTION TIMING

The following times (actually the number of clock pulses required) apply where the next M-instruction is contained in M-memory. In general, five clocks should be added to the time indicated if the M instruction is fetched from S-memory.

1C Register Move

The number of clocks required for 1C micros varies considerably. Refer to Table 2-11.

Table 2-11. 1C Micro Timing Variants

Source \ Sink	TOP M A	MSM MSM	Data CMND	MBR and A Out-of-Bounds	M and A Out-of-Bounds	Others
MSM, SUM, DIFF	3	3	4	3	6	2
DATA	4	4	-	4	-	3
U	U + 1	U + 2	U + 2	U + 1	-	U
OTHERS	2	2	3	2	6	1

2C Scratchpad Move	1 *
MSM, SUM, DIFF as source	2
DATA as source	3
U as source	U**
MSM, TOPM, A as destination	2 *
DATA, CMND as destination	3 *
MBR as destination and A out-of-bounds	2 *
M as destination and A out-of-bounds	6

*Add one clock if previous micro was write into scratchpad.

**Number of clocks depends on time required to fill U-register from the cassette.

3C 4-Bit Manipulate	1
Skip taken	2
BICN, FLCN XYCN, XYST	2
BICN, FLCN, XYCN XYST and skip taken	3
45C Bit Test Branch	1
Branch taken	2

6C Skip When	1
Skip taken	2
BICN, FLCN, XYST, XYCN	2
BICN, FLCN, XYST, XYCN and skip taken	3
7C Read Memory	
Followed by non-concurrent M-OP	6
Followed by memory cycle	5
Followed by 3 concurrent M-OP's	3
7C Write Memory	
Followed by non-concurrent M-OP	4
Followed by memory cycle	5
Followed by 1 concurrent M-OP	3
If outside base-limit and inhibited	1
8C Move 8-Bit Literal	1
A, MSM as destination	2
M as destination and A out-of-bounds	6
9C Move 24-Bit Literal	3
MSM as destination	4
MSM, TAS as destination and A out-of-bounds	8
Others as destination and A out-of-bounds	7
Tape mode	U + 1 *

*Number of clocks + 1 depends on time to fill U-register from the cassette.

10C Shift/Rotate T-Register Left	1
A, TOPM, MSM as destinations	2
DATA, CMND as destinations	3
MBR and A out-of-bounds	2
M and A out-of-bounds	6
11C Extract From T-Register	1
123C Branch	2

145C Call	2
2D Swap Memory	
Followed by non-concurrent M-OP	10
Followed by memory cycle	9
Followed by 7 concurrent M-OP's	3
3D Clear Registers	1
4D Shift/Rotate X or Y Left/Right	1+S/R count
5D Shift/Rotate X and Y Left/Right	1+S/R count
6D Count FA/FL	1
7D Swap F with DPW	1
8D Scratchpad Relate FA	2
9D Monitor	1
1E Dispatch	
Dispatch lockout	Same as 7C read plus 1
Skip taken	Same as 7C read plus 2
Dispatch write	Same as 7C write
Dispatch read	Same as 7C read plus 1
Dispatch read and clear	Same as 7C read plus 1
Dispatch port absent	1
2E Cassette Control	1
3E Bias	1
Skip taken	2
4E Store F Into DPW	1
5E Load F from DPW	1
6E Carry F.F. Manipulate	1
7E Exercise MSM	6
1F Halt	1
2F Overlay M-String	4 + 6 x number of words moved
FL = P initially	1

3F Normalize X	1 + number of bits shifted
4F Bind	3
No-op	1

PROCESSOR STATES FLOW CHART

The following chart indicates the prerequisite control signals for specific events to occur in the various Current States of processor operation.

State	Event or Series of Events	Prerequisite Control Level(s)*
Execute Current State (EXECCSP.)	INCA	SKIP
	DSETM to zero	RUN · SKIP
	DSETA	BRANCH
	DSETM to zero	RUN · BRANCH
	INCA DSETM (to M-string only) M ← MSM	<RUN> JUMP/ · AOB/ · MSINK/ · HALT/
	INCA DSETM (to M-string only) M ← MSM XPOSF ← 0 Gate-stopping state	<RUN> JUMP/ · AOB/ · MSINK/ · HALT
	INCA DSETM M ← MSM, M ← MX } Bit OR	<RUN> JUMP/ · AOB/ · MSINK · HALT/
	INCA DSETM M ← MSM, M ← MX } Bit OR XPOSF ← 0 Gate-stopping state	<RUN> JUMP/ · AOB/ · MSINK · HALT

*This symbol < > indicates operating mode (RUN, STEP or TAPE) selected on the console.

State	Event or Series of Events	Prerequisite Control Level(s)*
Execute Current State (EXECCSP.) (Cont)	FETCH=1 XPOSF ← 0 Gate FETCH state	<RUN> JUMP/ · AOB · MSINK/ · HALT/ <RUN> JUMP/ · AOB · MSINK/ · HALT
	INCA DSETM (to MEX only) M ← MX	<RUN> JUMP/ · AOB · MSINK · HALT/
	INCA DSETM (to MEX only) M ← MX XPOSF ← 0 Gate-stopping state	<RUN> JUMP/ · AOB · MSINK · HALT
	XPOSF ← 0 Gate-stopping state	<TAPE> · HALT
	XPOSF ← 0 Gate FETCH state	<TAPE> · HALT/
Start Current State (STRTCCSP.)	XPOSF ← 1 HALT ← 0 Gate execute state	<RUN>
	Cassette start DSETM (to zero) XPOSF ← 1 HALT ← 0 Gate execute state	<TAPE>
	Parity error lamp ← 0	

State	Event or Series of Events	Prerequisite Control Level(s)*
Fetch Current State (FTCHCCSP.)	INCA DSETM (to MEX only) M ← MX	DNMEM
	XPOSF ← 1 Gate execute state	DNMEM · HALT/
	Gate-stopping state	DNMEM · HALT
	DSETM (to MEX only) M ← MX UDATA accept = 1 XPOSF ← 1 Gate execute state	UDP · HALT/
	DSETM (to MEX only) M ← MX UDATA accept = 1 Gate-stopping state	UDP · HALT
	HALT = 1 HALT ← 1 Parity error lamp ← 1	Cassette parity error MFETCH parity error
Stop Current State (STOPCCSP.)	Cassette stop Gate console state	
Console Current State (CONSCSP.)	TAS push/pop ignore = 1 Enable load/display function	
	Gate-starting state	*START button
	INCA by 1	[Move MSM to ϕ] * INCR button

State	Event or Series of Events	Prerequisite Control Level(s)*
Console Current State (CONSCSP.) (Cont)	MEX = CSW 00 → 23 MOP ← [Move ϕ to ——] for 2CCP	Load
	MOP ← [Move —— to ϕ]	Load/

PORT ADAPTER/PORT DEVICE INTERFACE CONTROL (PAPDIC)

The port adapter/port device interface control serves as the link between the processor and S-memory, and is used for all operations which involve memory access. These include the 7C read, 7C write, 2D swap, 1E dispatch, and 2F overlay micros, plus S-memory micro fetching. PAPDIC functions as a semi-independent unit, maintaining control over memory cycles initiated by the processor. Because constant attention to memory cycles is not required of it, the processor is free to perform other non-conflicting operations while such cycles are in progress. This concurrent execution capability enhances the system's operating speed through conservation of time.

PAPDIC MICRO DECODING

The micros which exercise PAPDIC are decoded by the logic shown in figure 2-125a. Note that a controlling signal is developed for each of the five micro types, plus the variants of the dispatch micro. In addition to controlling the operation of PAPDIC, these signals are used to develop the memory cycle mode signals which are sent to the port interchange.

ADDRESS TRANSFER CONTROL

The address transfer control logic (figure 2-126) is used to enable memory addressing. Its basic functions are to control enabling of the line drivers for data going to memory (during address time only) and to generate the signal NSADDRH1 which increments the sequencer from idle to address state. Driver control is accomplished by a decoding circuit which is active only during the execution of basic 3 micros when PAPDIC is in the idle or write current states, or in read current state with FCCL true. This circuit may either enable or disable the line drivers depending on the conditions which exist at its inputs. In addition, the drivers may be enabled by additional logic when a dispatch micro is executed.

For basic 3 micros, the signals BRDMOPH1 (basic read micro), BDFAD.QO (bad FA address) and CD2F..AO (override bit) are monitored. Enabling of the drivers is allowed under all conditions except when BDFAD.QO is true and the other two signals are false. This indicates that a bad FA address has been detected (outside the limits set by the contents or BR and LR), and writing is not to be allowed. The exception condition generates STOPB3H1, which resets the driver enable flip-flops. If the memory cycle is to be allowed, D←FA..H is generated, enabling the gating of FA to the drivers, and, in addition, enabling the drivers themselves (DRVR←1HO) plus generating NSADDRH1 (next state address). The two latter signals may also be generated when a dispatch micro (DISP6.H1) is executing and the port absent variant (DPWMOPH1) is not present.

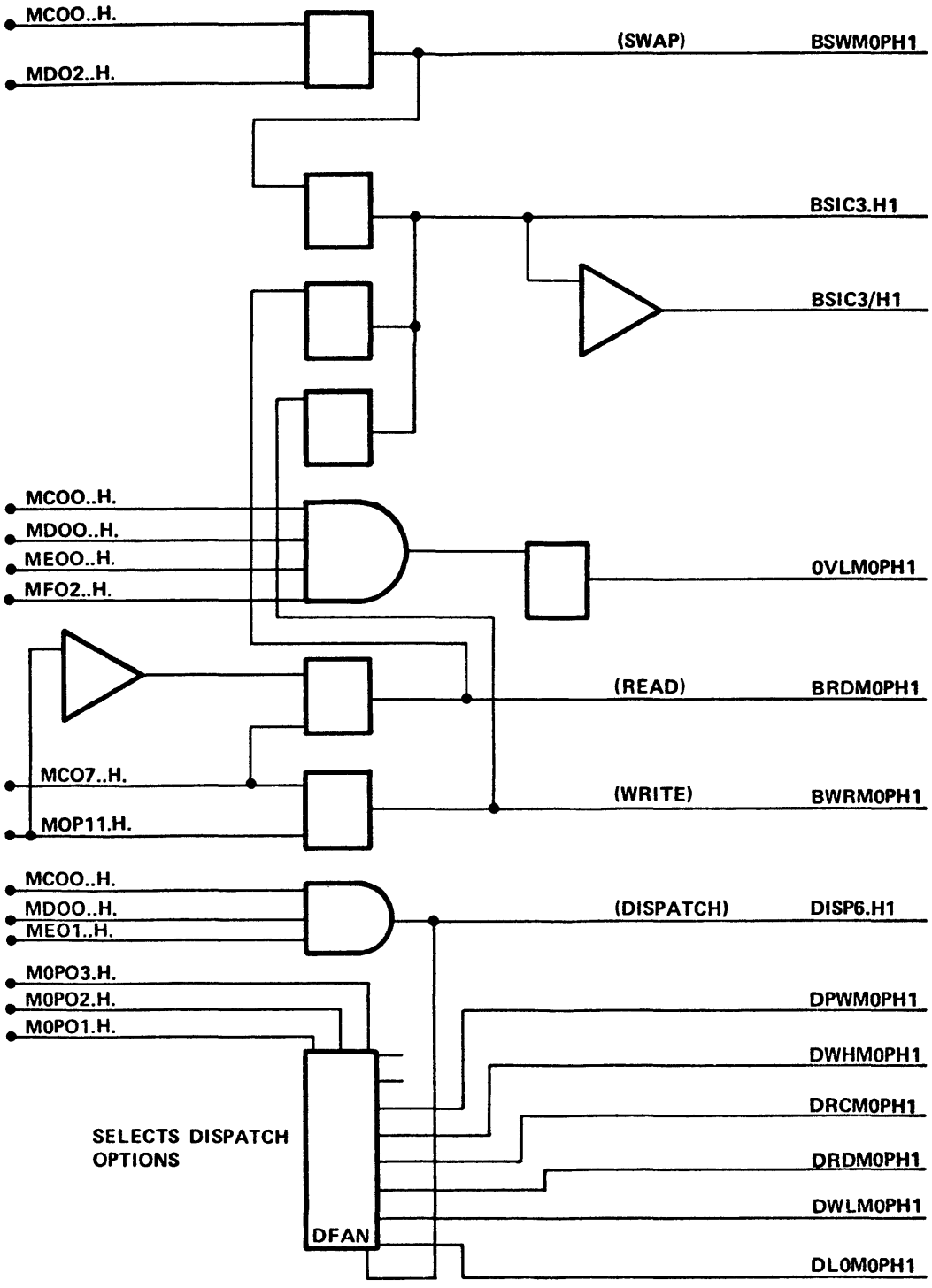


Figure 2-125a. Basic 3 and Dispatch Micro Decode Logic

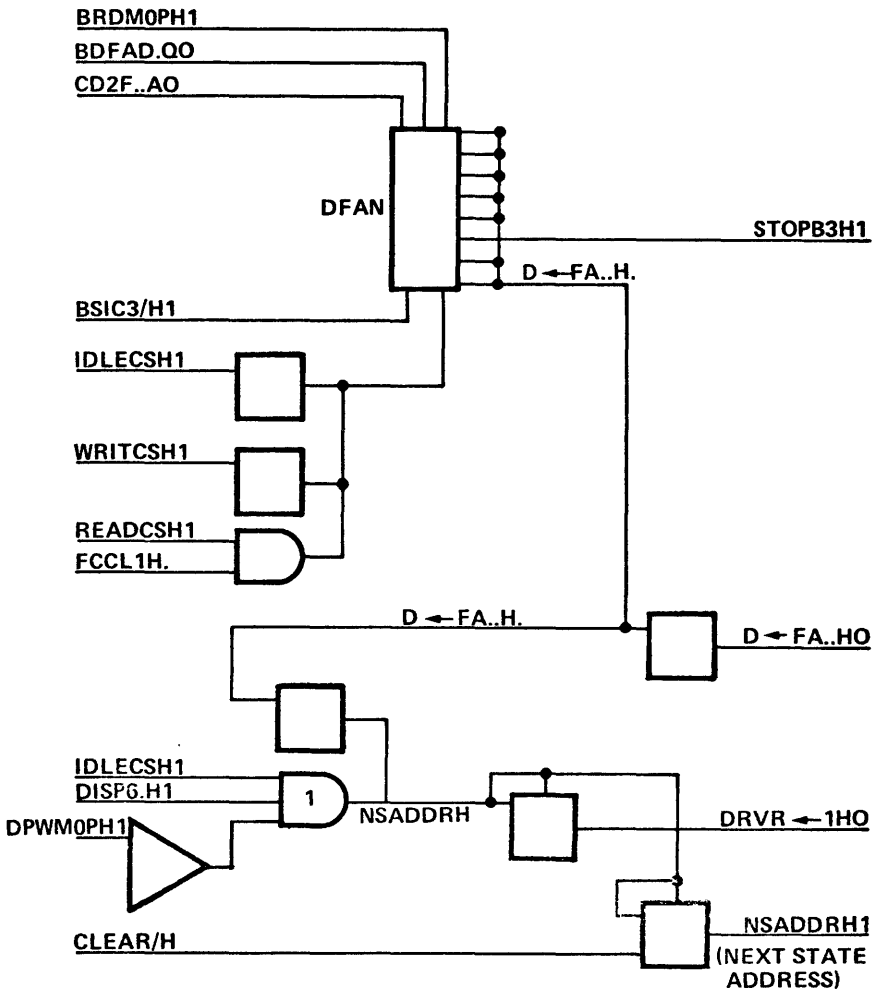


Figure 2-126. Address Transfer Control

PAPDIC SEQUENCER

The overall operation of PAPDIC is controlled by the sequencer (figure 2-127). This circuit generates enabling signals which "condition" the PAPDIC logic to perform the various functions required of it, much like the sequencer which maintains overall control of the processor's operating states. The outputs of the sequencer are eight control levels, known as current states. Each of these force the PAPDIC logic to perform a certain function, with a typical memory operation being made up of three or more of the current states occurring in sequence.

The sequencer consists of two EFAN encoder chips (with their outputs parallel connected), three flip-flops, and a DFAN decoder chip. In practice, the current state is determined by the binarily-encoded value at the outputs of the encoders, which is set into the flip-flops, and then decoded by the DFAN. The encoders themselves are arranged to operate in a "branch from address state" manner. In other words, each memory cycle begins with the idle current state and proceeds to the address state. From there it may branch to either read, write, swap, or wait. (Refer to section 1 for a description of the current states and sequencer flow.) After the initial branch from address state, which is determined by micro decoding, the sequencer may proceed through any of the remaining states, with the actual sequence depending on which micro is executing and/or the state preceding each change.

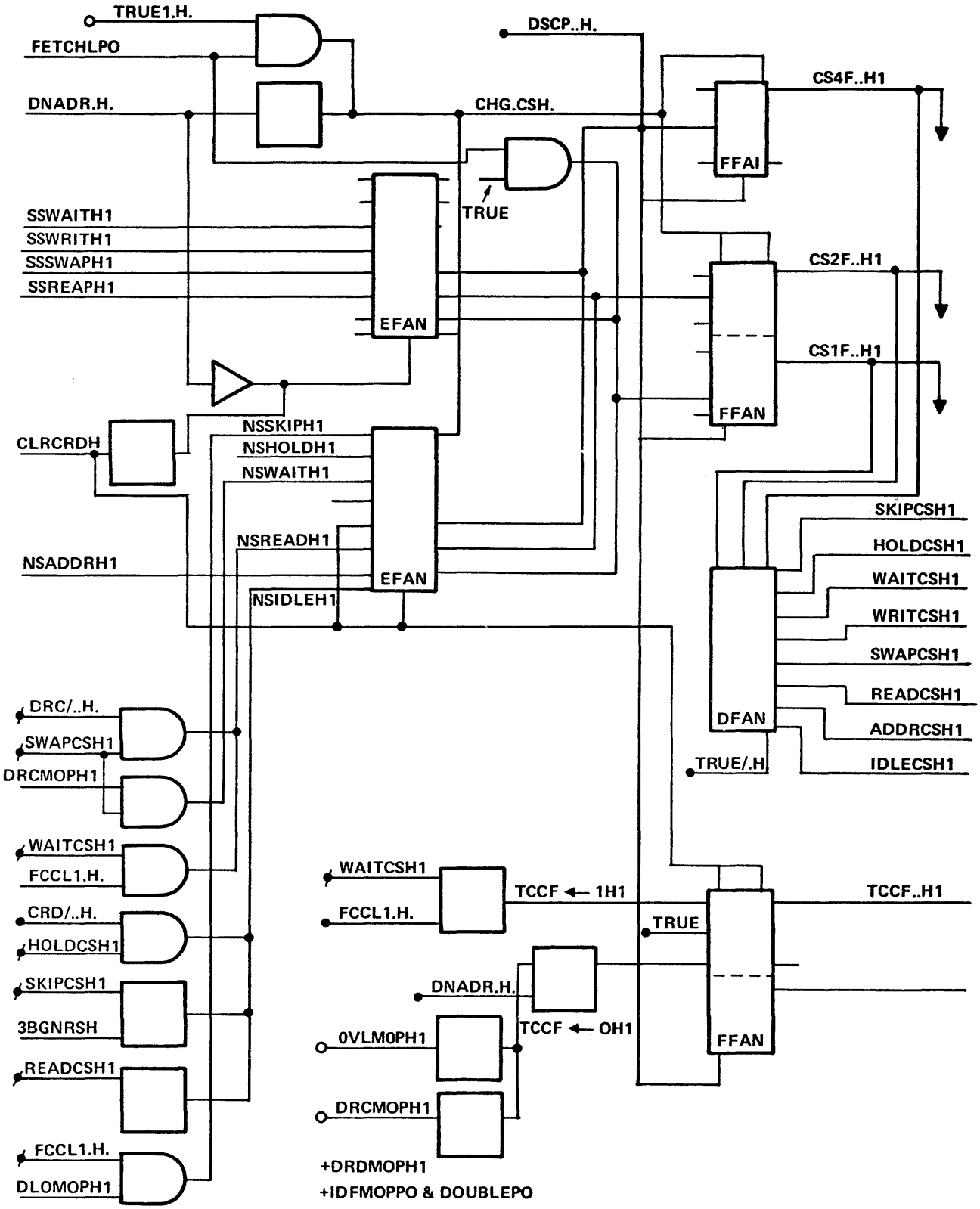


Figure 2-127. PAPDIC Control Sequencer

To accomplish the desired current state manipulations, two means of deriving the 3-bit current state code are provided. For normal changes of state (exclusive of the branch after address), the first EFAN chip (G7) is active. Driving this chip are "next state" signals produced by the gates labeled 1 in figure 2-127. Branching after address is done by the second EFAN (G7), which is active only when the signal DNADR.H. is true (and therefore only once during the execution of each micro). The second EFAN is driven by the "successor state" signals produced by the logic shown in figure 2-128. All successor state signals are the direct result of micro decoding. Note that none of the next state signals are ever true when DNADR.H. is true, so there is no possibility of conflict at the outputs of the EFANs. Operation of the sequencer is summarized in table 2-12.

Table 2-12. PAPDIC Sequencer Control Signals

Next State Signals	Logic Equations
NSIDLEH1	3BGNRSH + SKIPCSH1 + (HOLDCSH1 · CRD/..H.) NOTE 3BGNRSH = IDLECSH1 + WRITCSH1 + (READCSH1*FCCL1.1+) CRD/..H. is console read not.
NSADDRH1	CLEAR/ · (IDFMOPPO · IDLECSH1) + (IDLECSH1 · DISP6.H1 · DPWMP/H.) + (D←FA..H.) NOTE IDFMOPPO = internal data fetch (pseudo) micro. DISP6.H1 = dispatch micro (variants unspecified). DPWMP/H. = dispatch port withdrawal variant not.
NSREADH1	(DRC/..H. · SWAPCSH1) + (WAITCSH1 · FCCL1.H.) NOTE DRC/..H. = dispatch read and clear variant not.
NSHOLDH1	CRDMOPH1 * RCVR←OHO NOTE CRDMOPH1 = console read micro. RCVR←OHO = disable line receivers.
NSSKIPH1	READCSH1 · FCCL1.H. . DLOMOP1+1 NOTE DLOMOPH1 = dispatch micro with lockout variant.
Successor State Signals	**Logic Equations**
SSREADH1	(DUBLE/H. . IDFMOPPO) + BRDMOPH1 + CRDMOPH1 + FMDMOPH1 + DLOMOPH1 NOTE DUBLE/H. = Do not double clock read data. IDFMOPPO = Internal data fetch (pseudo) micro. BRDMOPH1 = Basic read micro. CRDMOPH1 = Console read micro. FMPMOPH1 = Fetch (pseudo) micro. DLOMOPH1 = Dispatch micro with lockout variant.

Table 2-12. PAPDIC Sequencer Control Signals (Continued)

Successor State Signals (Cont)	Logic Equations
SSSWAPH1	BSWMOPH1+DRCMOPH1 NOTE BSWMDPH1 = Basic swap micro. DRCMOPH1 = Dispatch micro with read and clear variant.
SSWRITH1	BWRMOPH1 + CWRMOPH1 + DWLMOPH1 + DWHMOPH1 NOTE BWRMOPH1 = Basic write micro. CWRMOPH1 = Console write micro. DWLMOPH1 = Dispatch micro with write low variant. DWHMOPH1 = Dispatch micro with write high variant.
SSWAITH1	OVLMOPH1 + DRDMOPH1 + (DOUBLEPO * 1DFMOPPO) NOTE OVLMOPH1 = Overlay micro. DRDMOPH1 = Dispatch micro with read variant. DOUBLEPO = Extend read data presentation to two clocks. IDFMOPPO = Internal data fetch micro.

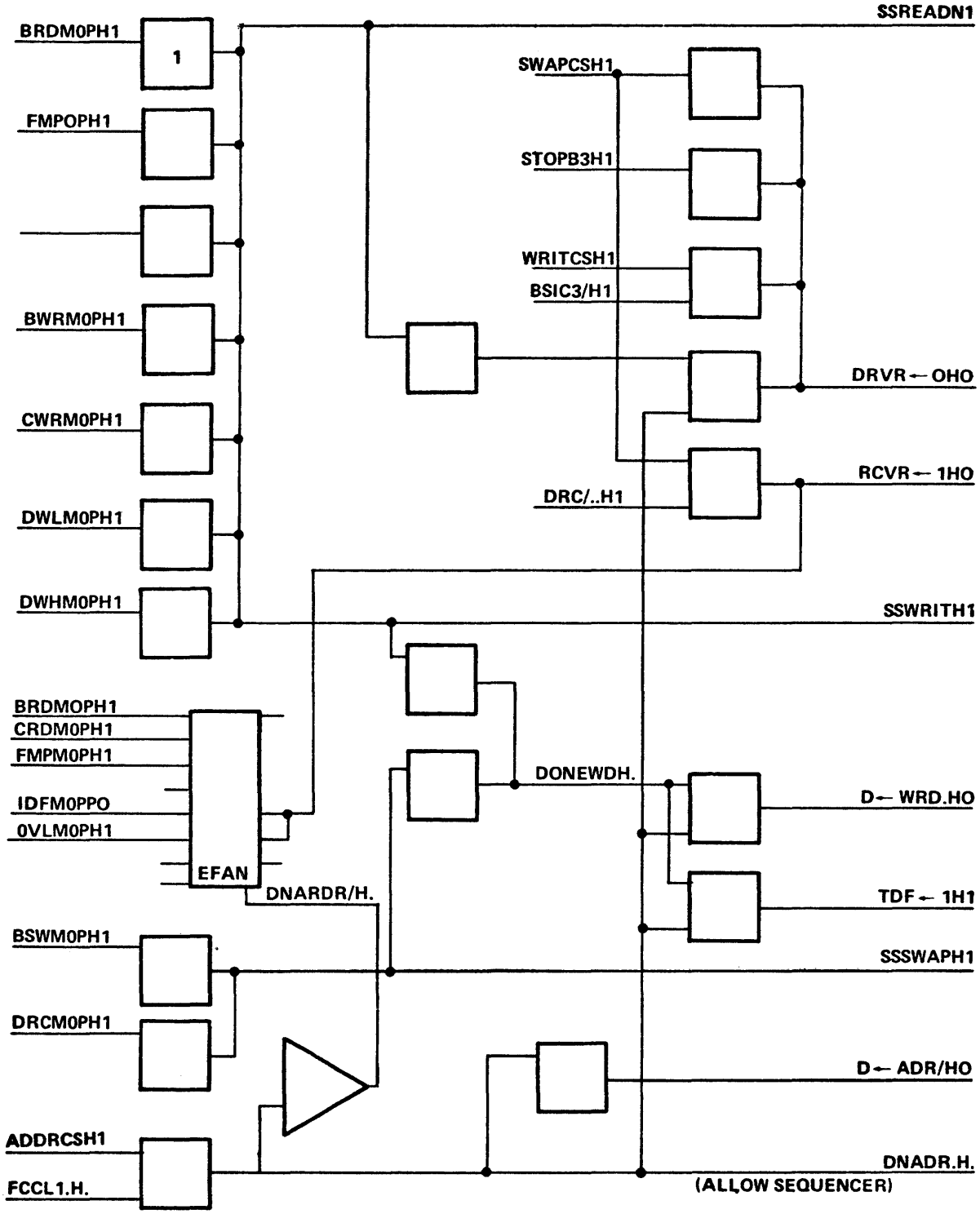


Figure 2-128. Successor State Signal Generation

PAPDIC BASIC 3 READ (7C) MICRO EXECUTION

The basic read micro is decoded when MC07..H and MOP11/H. are true, generating the levels BRDMOPH1 and BSIC3.H1 (figure 2-125). BSIC3/H1 false and IDLECSH1 (initially true) enables DFAN D7 (figure 2-126), generating the level D←FA..H (move the contents of FA to the line drivers). At the same time, DRVR←1HO (enable drivers) and NSADDRH1 also go true, which, at the next clock pulse, causes the drivers to contain the memory address (COOL through C23L), data field length and field direction sign (C24L through C29L), plus the mode control code (RQ1 through RQ5). For a basic read operation, RQ4 is only true when NSADDRH1 is true.

At the sequencer (figure 2-127), NSADDRH1 sets CS1F..H1, causing the ADDRCSH1 output of the DFAN to go true. With the address and other memory request information having been gated to the port interchange, the logic then waits for the signal FCCL (forward communication control level), which indicates that the address has been accepted. When FCCL appears, the level DNADR.H. (figure 2-127) goes true, disabling the line drivers (DRVR←0HO) and enabling the line receivers (RCVR←1HO). DNADR.H. also enables EFAN G6 in the sequencer, and because SSREADH1 was set when the micro was decoded, CS2F..H1 is set and CS1F..H1 is reset. (NSADDRH1 is false at the input of G7). This causes the level READCSH1 (read current state) to come true at the output of the DFAN. When the second FCCL pulse appears, the read data from memory is present, and is routed through the line receivers to the MEX. Further gating within the processor is determined by the logic shown in figure 2-129. Assuming that the Y-register has been selected as sink, flip-flop 1 would have been set (by MOP06.H. and BSIC3.H1) at address current state time. When READCSH1 comes true, the READ DFAN is enabled and the level Y←MX..HO appears, enabling transfer of the read data to Y.

With READCSH1 true, the second FCCL pulse generates 3BGNRSH., which, in turn, produces NSIDLEH1. This resets the sequencer (IDLECSH1) unless another basic 3 micro is waiting for Done memory in the concurrency logic. In this case, NSADDRH1 is also true at this time, causing the sequencer to go directly to the address current state. (Note that the EFAN chips are priority encoders, with the highest binarily-weighted input which is true taking precedence.) Basic 3 read timing is shown in figure 2-130.

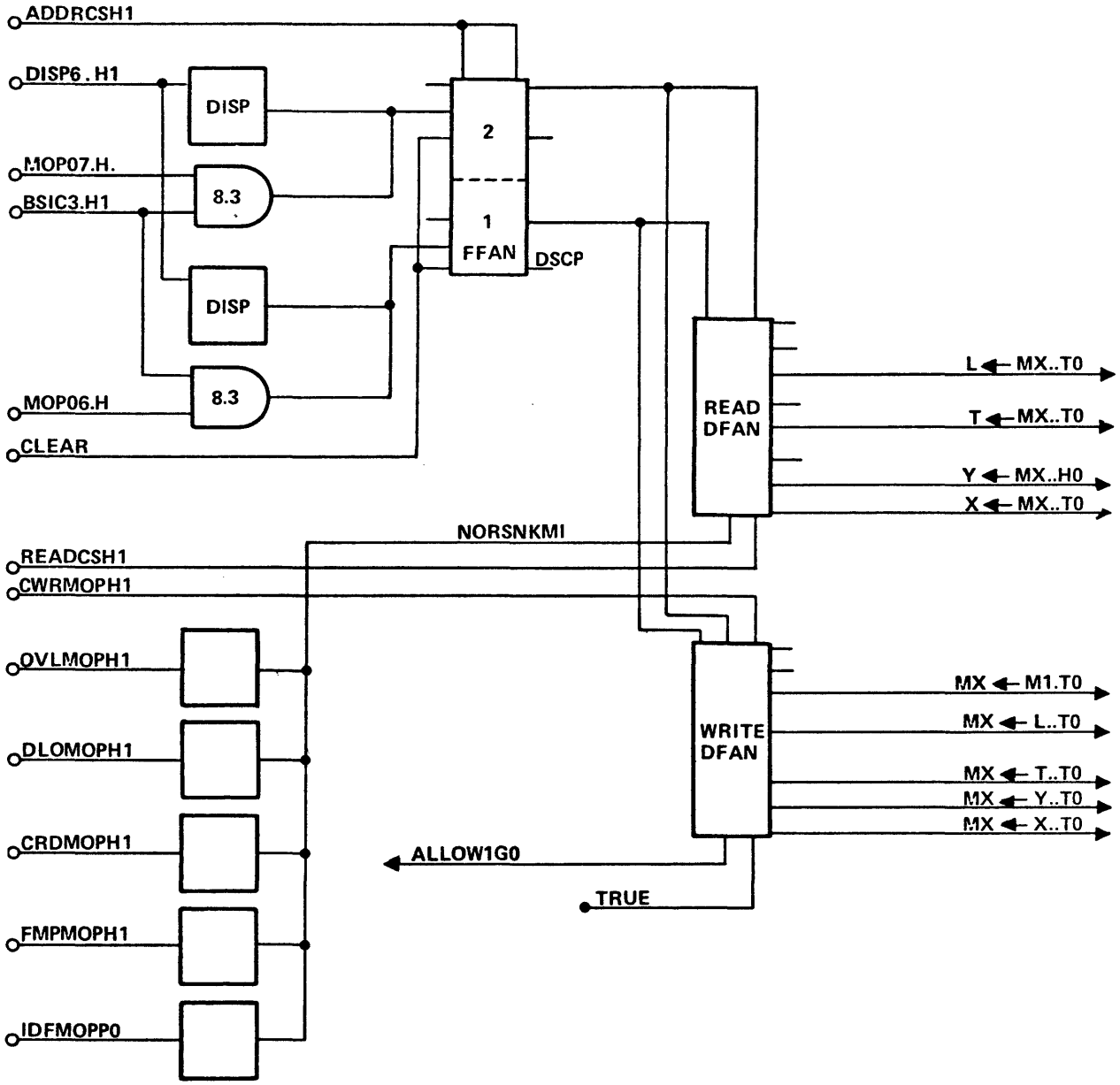


Figure 2-129. Main Exchange to-from Register Controls

PAPDIC DISPATCH READ AND CLEAR MICRO EXECUTION (1E WITH READ AND CLEAR VARIANT)

This micro accomplishes a number of functions as follows:

a. Clear the lock bit and interrupt bits in the dispatch register.

b. Read the seven least-significant bits of the dispatch register (source port number and channel number) and transfer them to the seven least-significant bits of the T-register.

c. Read the 24-bit message contained in S-memory address 0 and transfer it to the L-register.

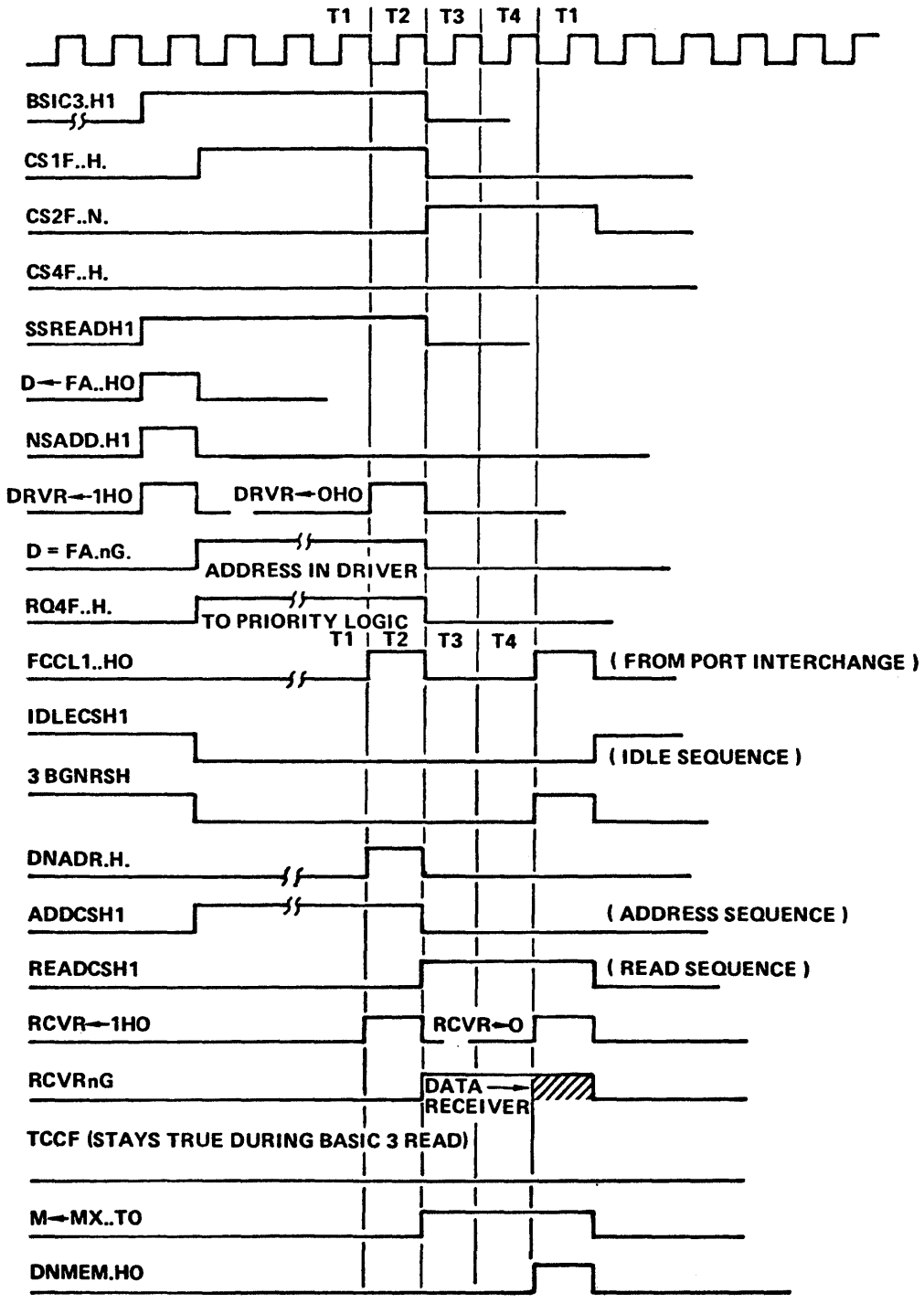


Figure 2-130. Basic 3 Read Timing Chart

The dispatch micro is decoded when M-register bits MC001.H., MD00..H., and ME01..H. are all true. These three levels indicate that MC=0 (M-register bits 12 through 15 are all false), MD=0 (same for M bits 08 through 11), and ME=01 (of M bits 04 through 07, only 04 is true). DISP6.H1 enables the DFAN in figure 2-125 which, in turn, decodes the contents of MOP01.H., MOP02.H., and MOP03.H. to determine which variant has been selected. In this case, DRCMOPH1 (dispatch read & clear) comes true. DISP6.H1 also generates NSADDRH1 and DRVR←1.H1 (enable line drivers). D←FA..H0 stays false, meaning that the drivers (C00L through C23L) are filled with zeroes. This causes address zero in memory to be accessed.

Simultaneously, with BSIC3/H1 (basic 3 not) true, flip-flop 2 in figure 2-131 is set, which causes C27L..H. and C28L..H. to be forced true (data field length = 24). In figure 2-132, RQ4F..H., RQ5F..H. and RQ2F..H. are set.

Note that SSSWAPH1 is true when DRCMOPH1 is true. At the next clock pulse, the sequencer advances to ADDRCSH1, and the logic waits for the first FCCL pulse from the port interchange.

When FCCL appears, the address and mode control signals have been accepted, and the lockout bit and interrupt bits in the dispatch register have been cleared. FCCL sets the sequencer to swap current state and TCCF is reset. During the swap state, the drivers are reset (DRVR←0H0) and one clock later the sequencer goes to the wait current state (WAITCSH1). With WAITCSH1 true, buffers are enabled (figure 2-133) which allow the contents of bi-directional lines C24L through C29L, RQ1L, and RQ2L to be read. These contain the source port number and channel number (XDSP0.T0 through XDSP6.T0) at this time. When the next FCCL pulse comes true (WAITCSH1 still true) the level T←XDISP. is enabled and this data is transferred to the seven least-significant bits of the T-register. Because the level TCCL is false, the second FCCL pulse stays true for two clocks.

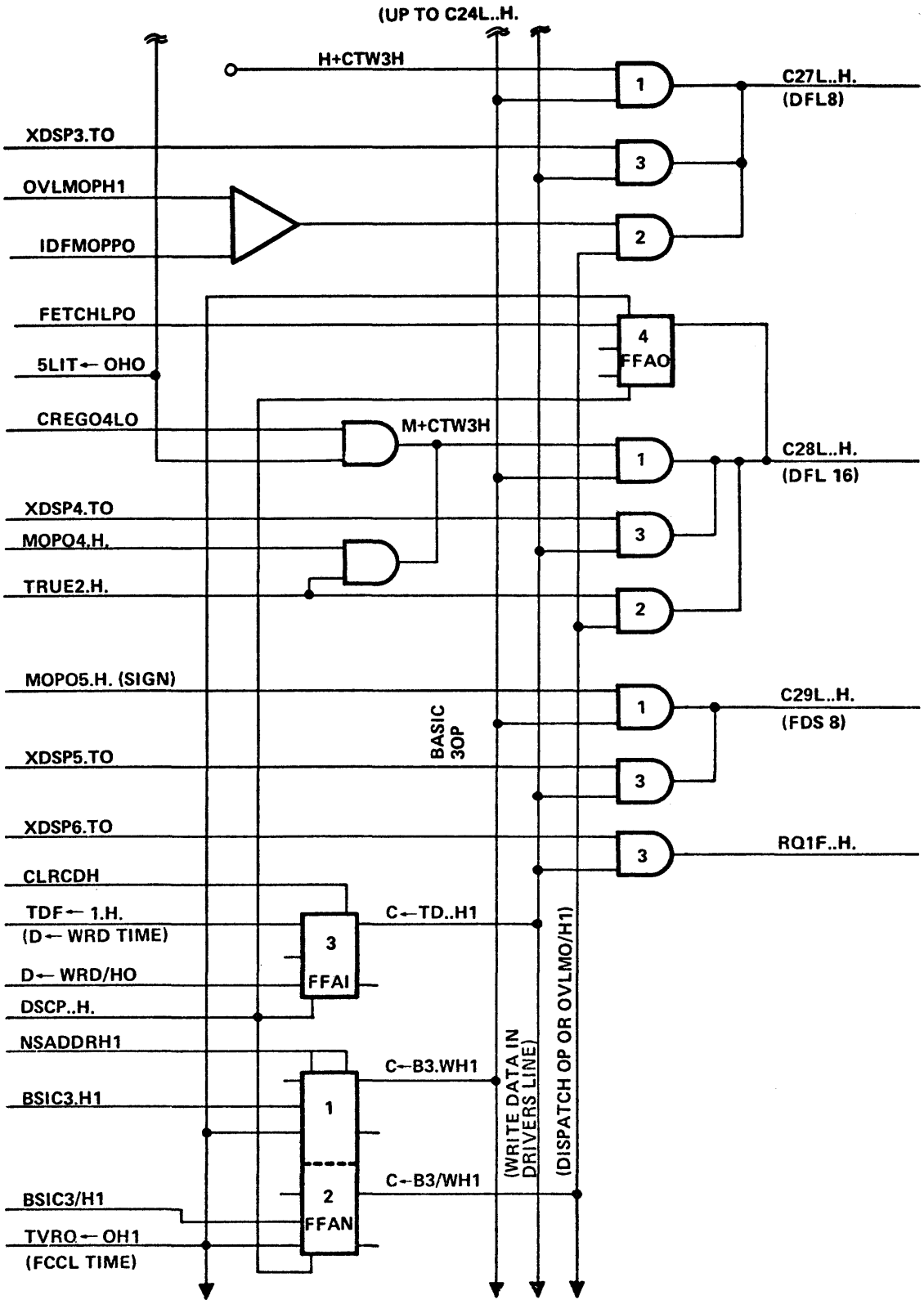


Figure 2-131. Field Length and Field Sign Gating Logic

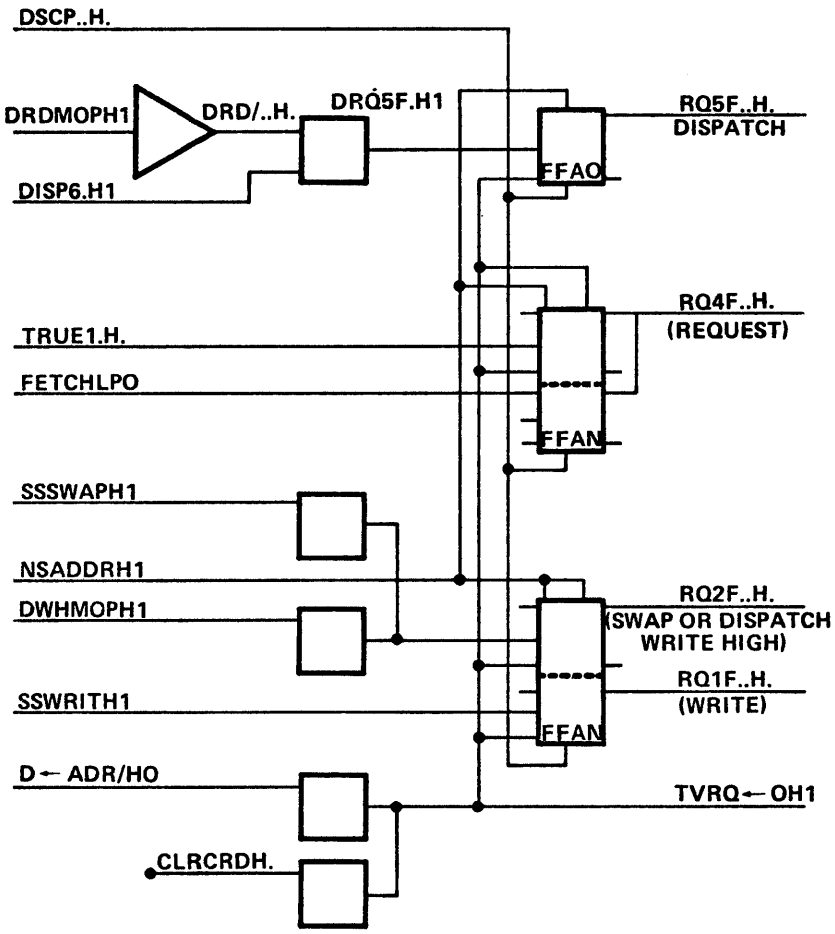


Figure 2-132. Request Line Flip-Flops

At the end of the first clock period of FCCL, the sequencer is set to the read state (READCSH1), the line receivers are enabled, and TCCF is set again. During the second clock period of FCCL the read data is available. Because DISP6.H1 was true, flip-flops 1 and 2 in figure 2-131 were set during the address state. Therefore, L←MX...T0 is true, which permits gating the read data into the L-register. Dispatch read and clear timing is shown in figure 2-134. Also shown are timing diagrams for Basic 3 write, Basic 3 swap, dispatch write lock, and fetch S-memory. These comprise figures 2-135, 2-136, 2-137 and, 2-138, respectively.

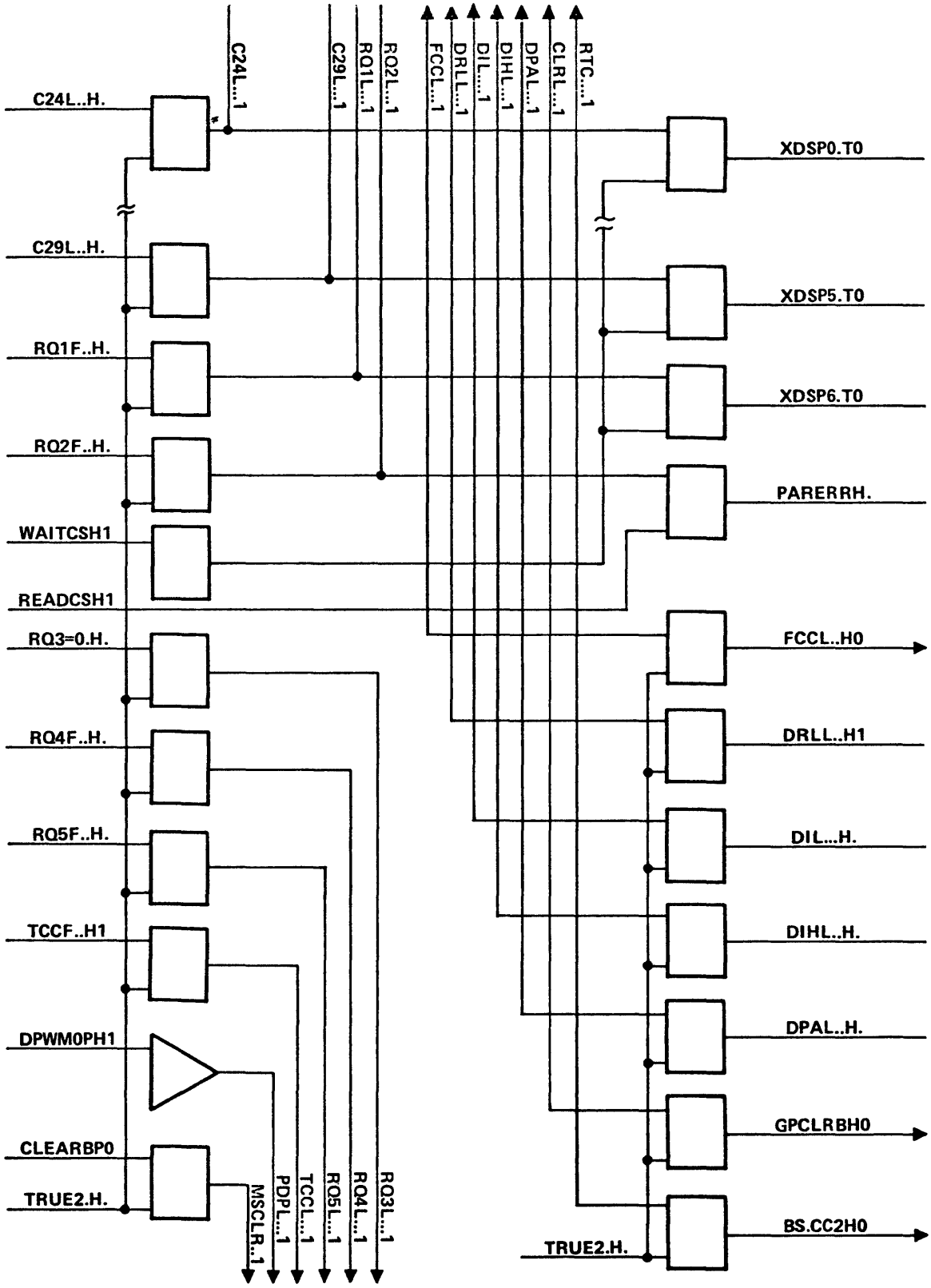


Figure 2-133. Drivers and Receivers for Control Lines

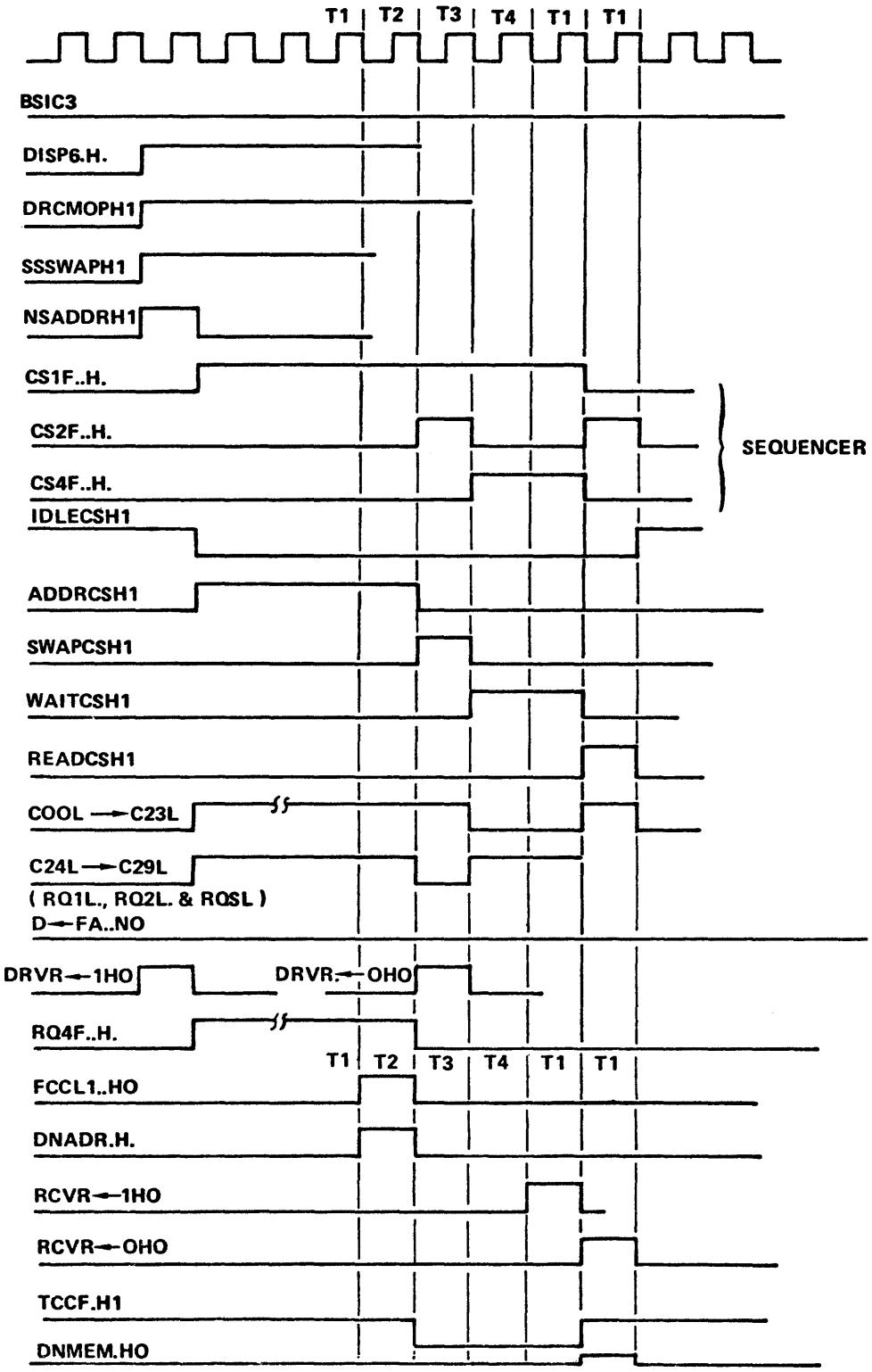


Figure 2-134. Dispatch Read and Clear Timing

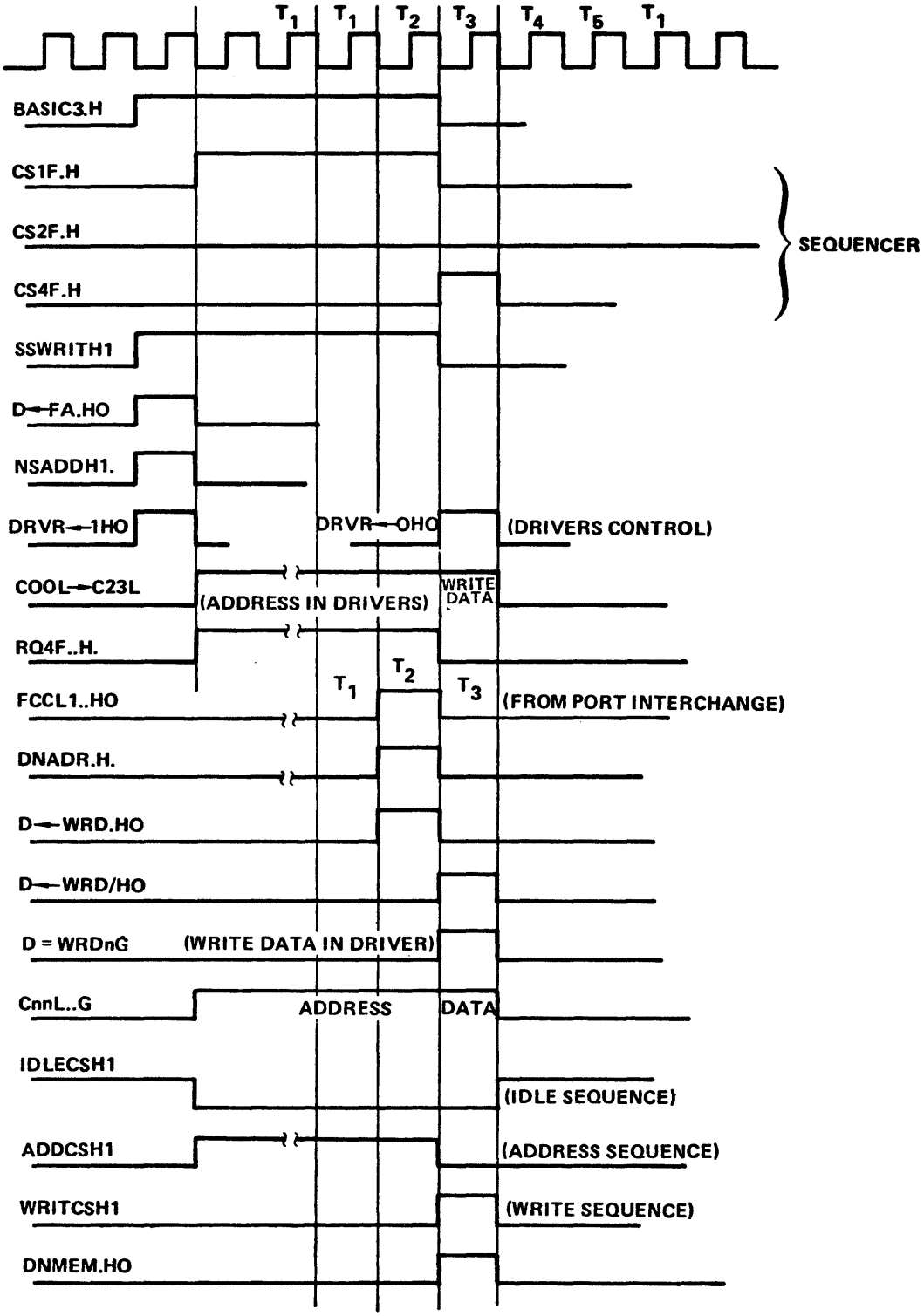


Figure 2-135. Basic 3 Write Timing Chart

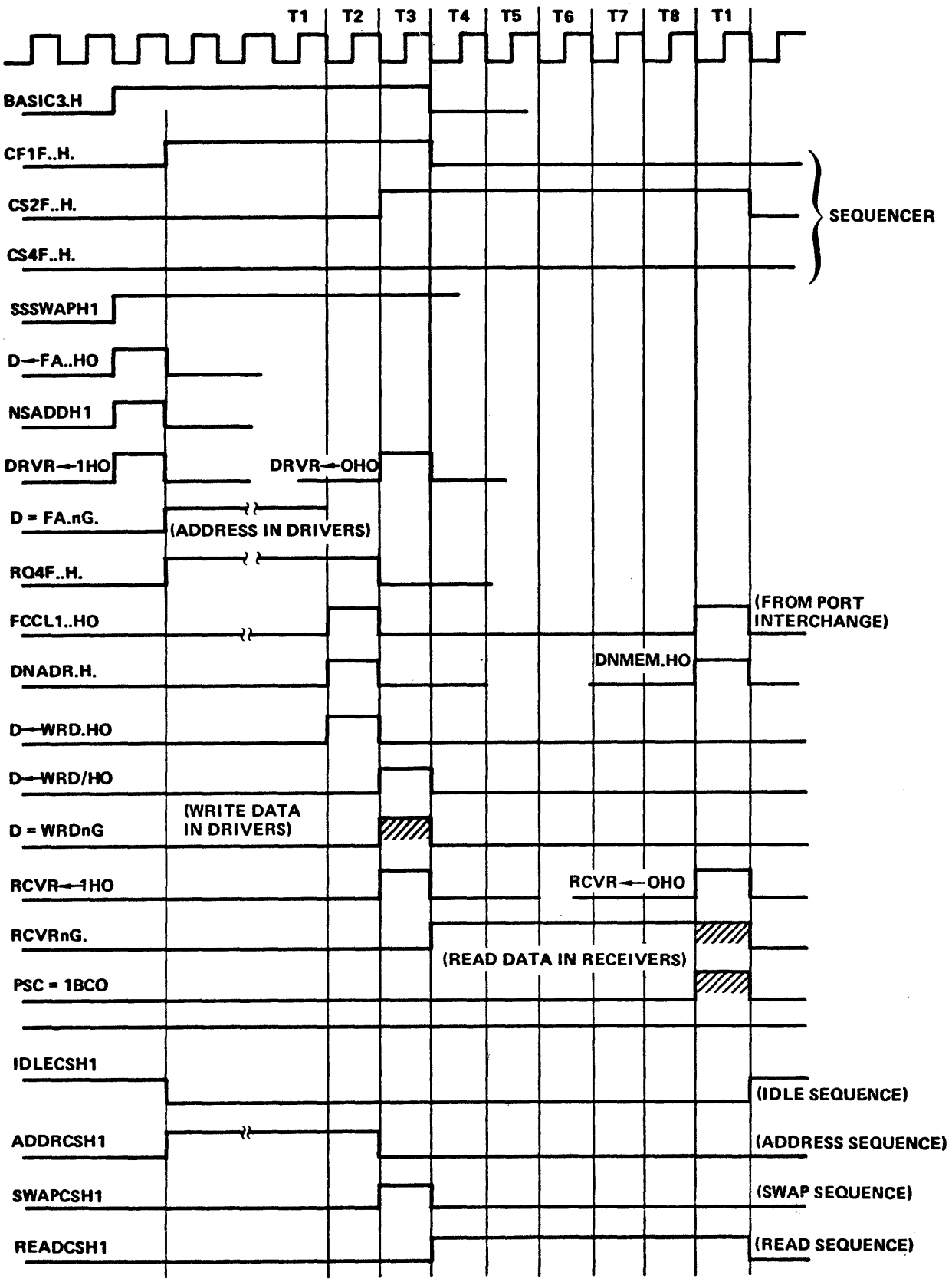


Figure 2-136. Basic 3 Swap Timing Chart

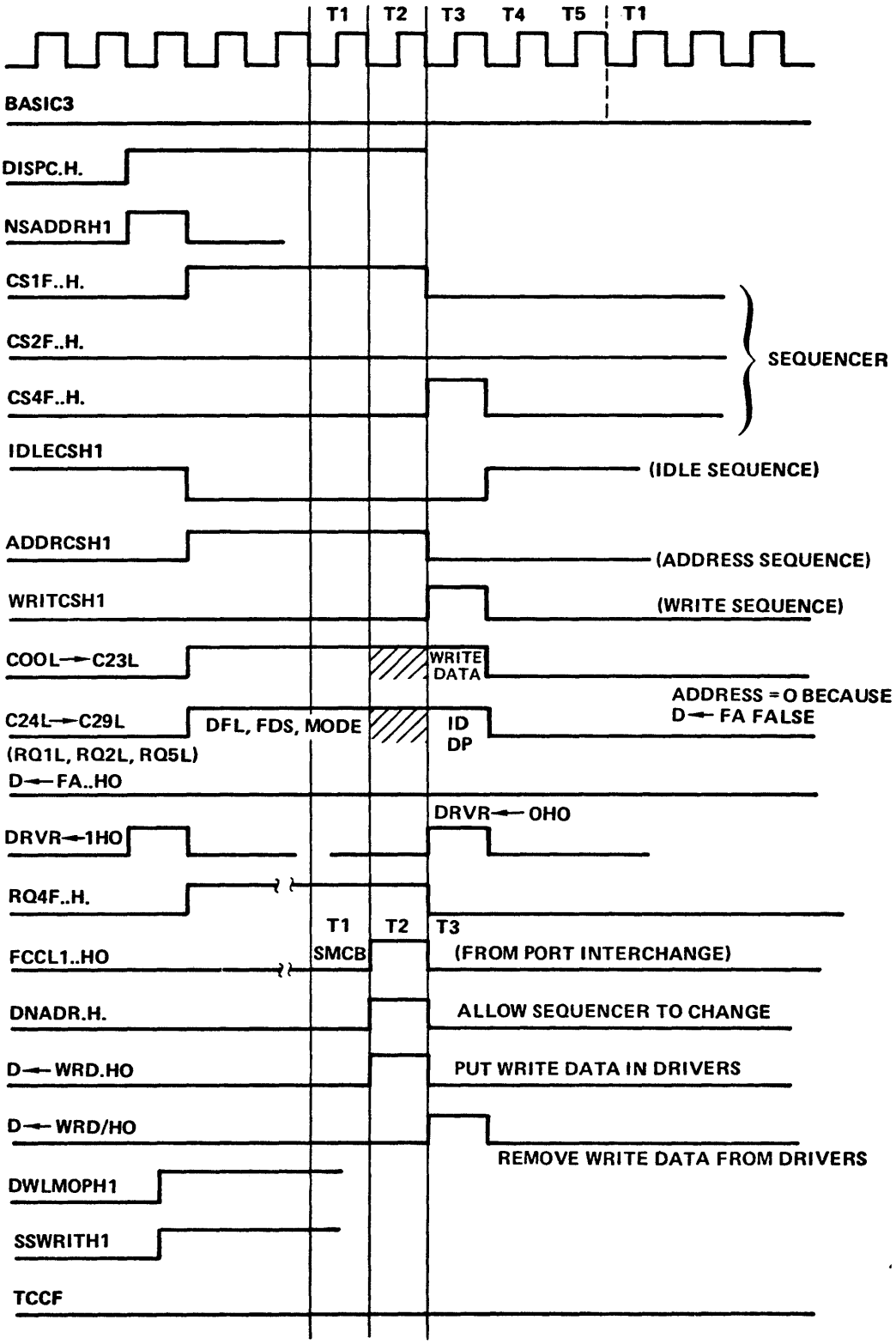


Figure 2-137. Dispatch Write Lock Timing Chart

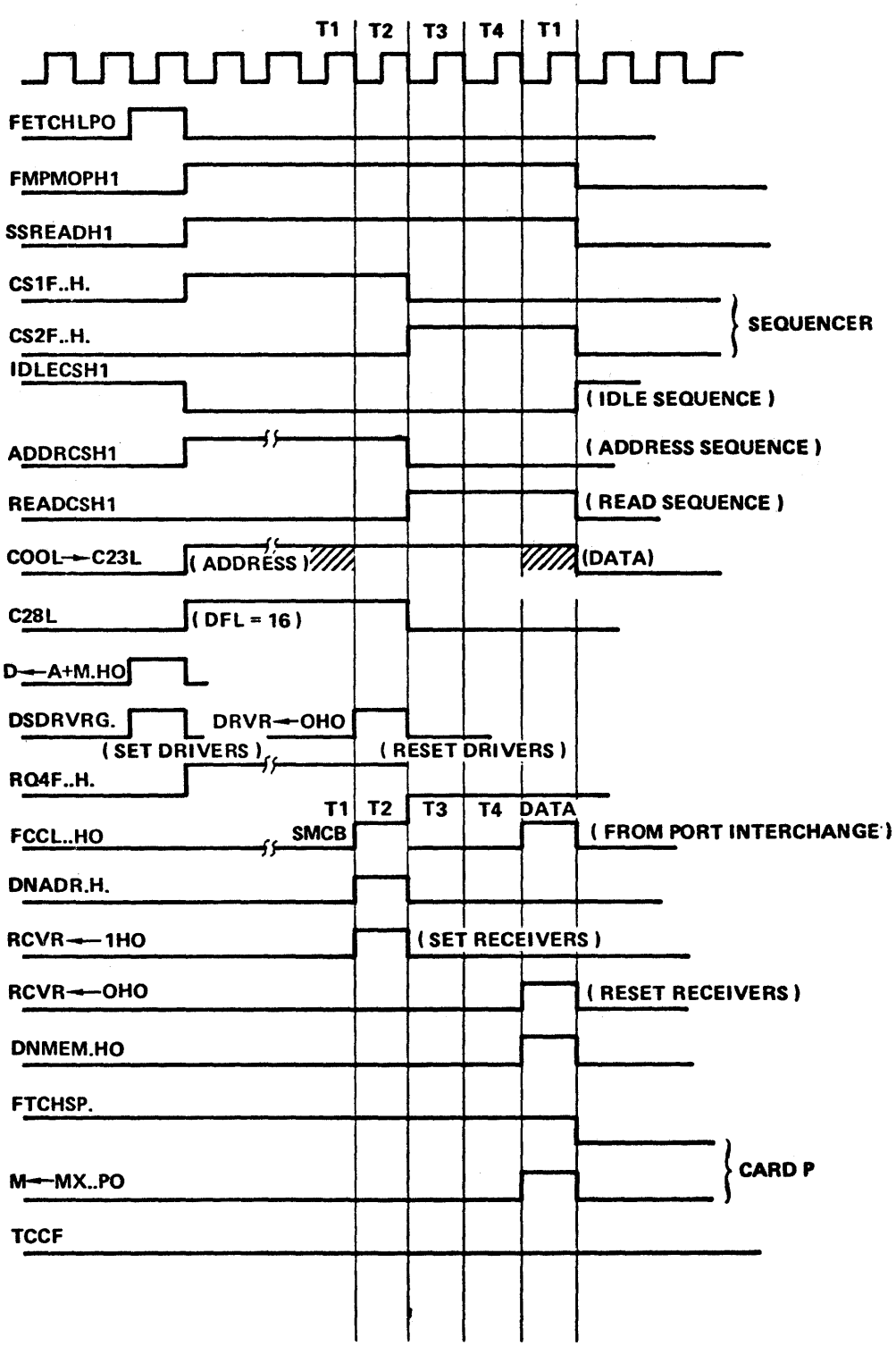


Figure 2-138. Fetch S-Memory Timing Chart

PORT INTERCHANGE

The port interchange is that portion of the system which maintains control over S-memory. A separate circuit to perform this function is required because the system may be configured to employ multiple S-memory user devices. To satisfy the memory access needs of several independent devices (of which one is the processor), the port interchange has been provided with a number of diverse capabilities. Included in these are access priority resolution, distribution of synchronizing signals, memory addressing, field selection and isolation, intermediate data storage, parity generation and manipulation, control signal interfacing, and facilities for communication between user devices. The actual hardware sections for accomplishing these functions include a memory address register, a memory information register, priority resolution logic, a mask generator, a field isolation unit (rotator), a dispatch register, parity logic, and the real time clock circuit. The port interchange occupies six logic cards, and is located in the system between the M-memory processor and S-memory base.

Accessing memory involves presentation of a memory request by a port device. Included in the request are a control code which designates the type of operation desired, a 24-bit binary memory address, a five-bit binary field length, and a one-bit field direction sign. In the port interchange, such requests are subjected to priority resolution, and, if more than one is received simultaneously, that bearing the highest priority is granted.

Granting a memory request consists of accepting the address and control codes, and signaling the port device that the operation is to proceed.

PORT INTERCHANGE CLOCK

The port interchange derives synchronization from the 6.0 megahertz system clock pulses which are 167 nanoseconds wide. This clock signal is actually generated in the clock module, and is passed directly to the port interchange. The port interchange clock distribution circuit is shown in figure 2-139.

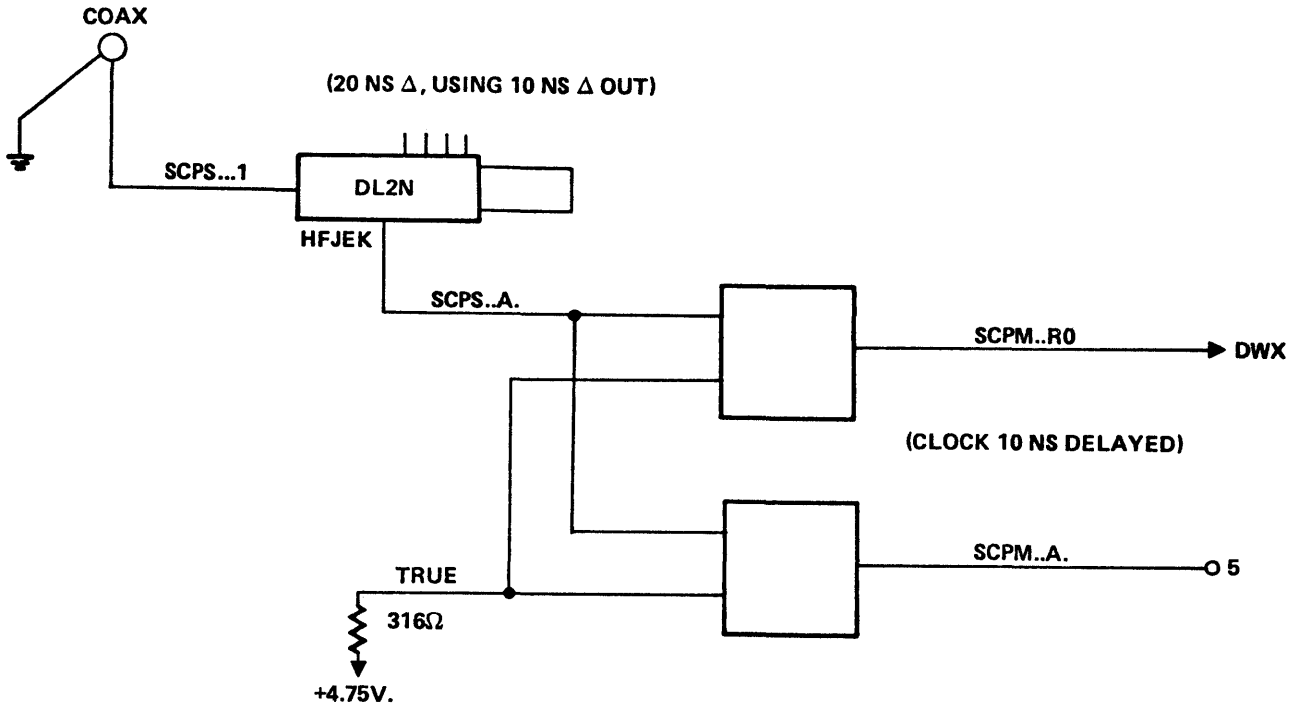


Figure 2-139. The Port Interchange Clock Distribution

REAL TIME CLOCK

The real time clock logic generates a 100-millisecond signal (10 clocks per second) which is used primarily for setting interrupts. This signal is derived from the 50/60 hertz power line frequency, and is supplied to each port device by way of the port adapter interface.

Generation of the real time clock signal is accomplished by the circuit shown in figure 2-140. The input circuit is a clipper which is used to change the sine wave power line signal to a square wave. CR1 and CR2 are negative clamp diodes to protect BHAZ and IHAO, respectively. CR5, 6, and 7 are positive clamp diodes to protect and clamp RTC04 at +2.0 volts when positive. CR6 and 7, in conjunction with CR4, perform the same function for RTC02.

FFA1 operates in the JK mode, and is set during the positive half cycle of the power line wave. The negative half cycle resets it. FFA0 operates in the D-set mode, and follows FFA1 one 6 megahertz clock later. The ANDed output of the two flip-flops (RTC10.C.) consists of a 167-nanosecond-wide pulse occurring once every 20 milliseconds (figure 2-141).

RTC10.C. drives the RFAN, which operates in the add mode. With its carry input true, the RFAN counts from 0 to 5 (or 0 to 4 for 50 hertz line frequency), with the count being incremented by one for each RTC10.C. pulse. When the RFAN output is equal to binary 5 (60 hertz), RTC14.C. comes true, enabling the output buffer where it is ANDed with RTC10.C.. The net output (RTCLKBCA) is a 167-nanosecond-wide pulse occurring once every 100 milliseconds. Note that RT14..C. is also fed back to the clear input of the RFAN, resetting it. For 50 hertz operation a count of only 4 is required, with RT13..C. replacing RT14..C.. The conversion is effected by jumper chip connections as shown in figure 2-140, and must be set by the field engineer.

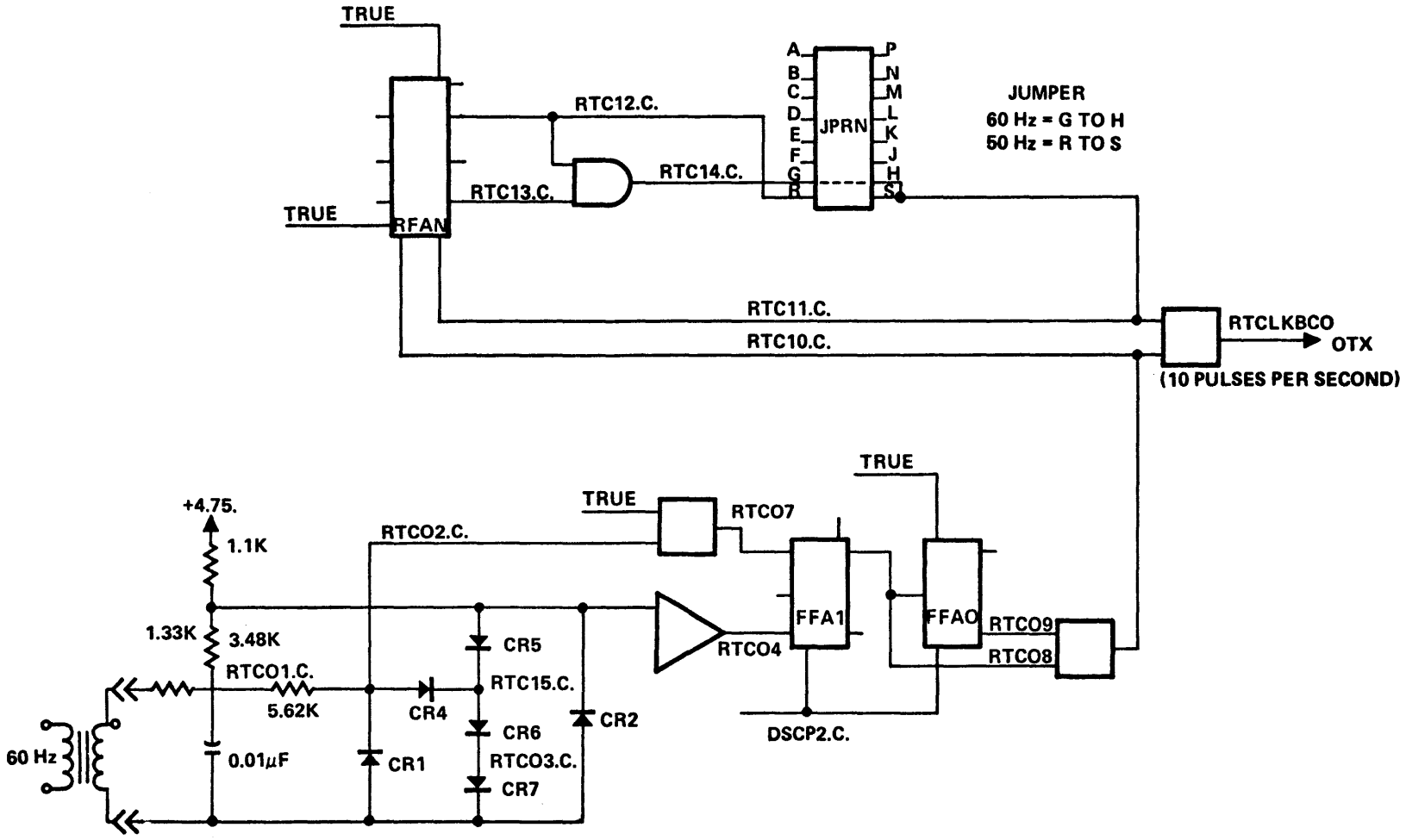


Figure 2-140. The Real-Time Clock Generation

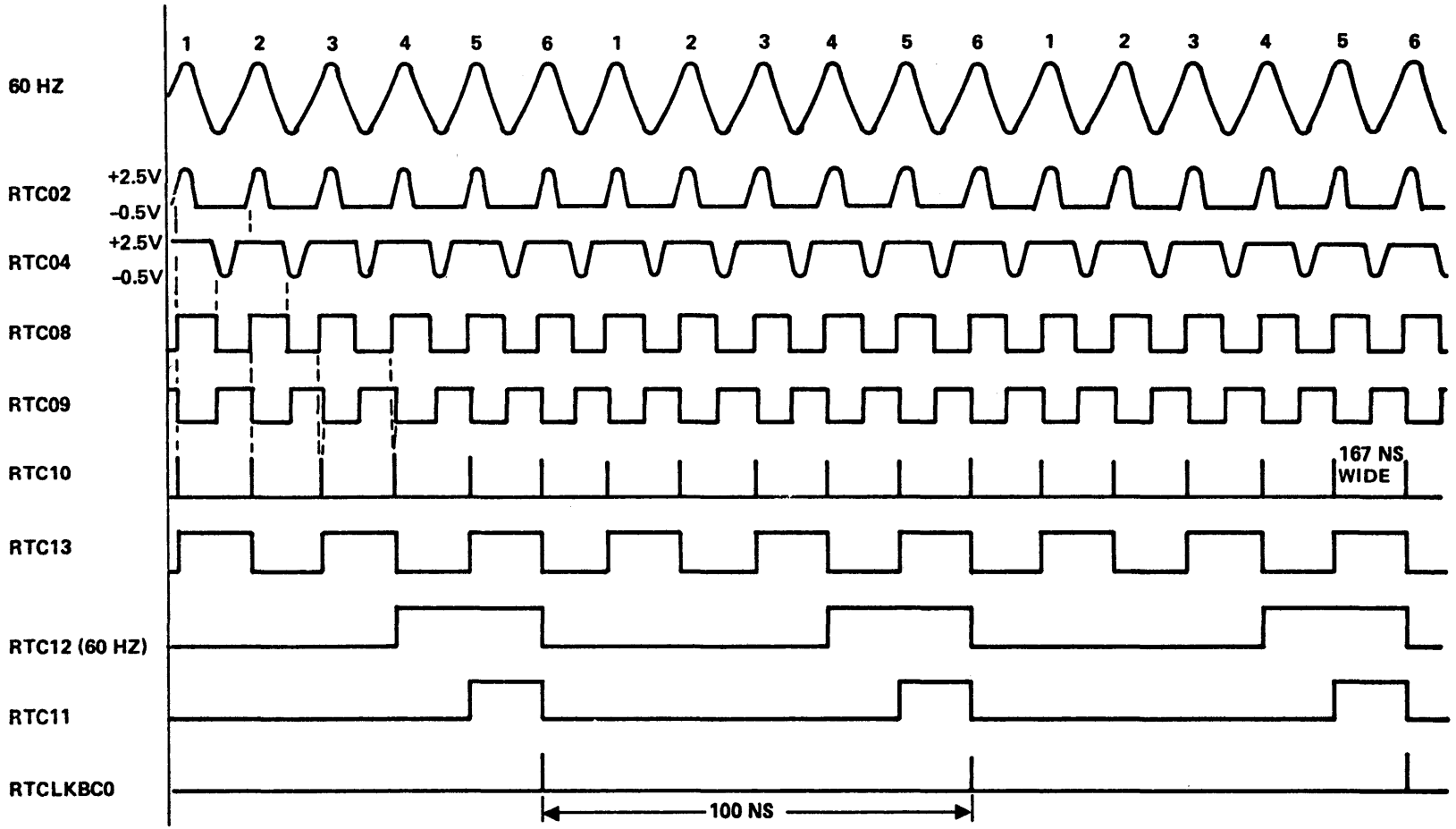


Figure 2-141. The Real Time Clock Pulse Timing Chart

POWER-UP LOGIC

The power-up level (PWR.UP.0) is generated in the power supply. Refer to figure 2-142. This signal is false during the power-up sequence and goes true when the sequence is complete. PWR.UP.0 remains true until the system is powered down.

The power up level is fed into the port interchange, which distributes this level via the port adapter interface to each port device.

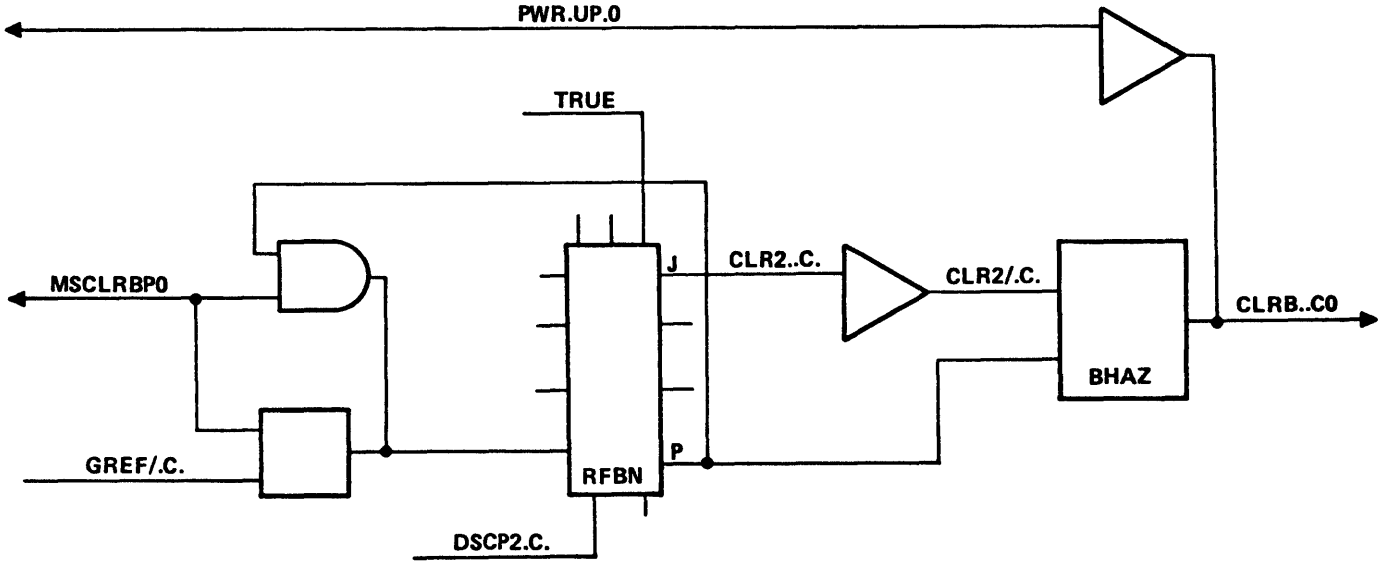


Figure 2-142. Power-up and Clear Level Generation

CLEAR SIGNAL LOGIC

The port interchange distributes a clear signal via the port adapter interface to each port device. This occurs when it receives a master clear signal (MSCLRBP0) from the control panel, or when it receives a power-up clear level (PWR.UP.0) from the system. The clear (CLRB..C0) level is three clocks in duration. Operationally, MSCLRBP0 is gated with refresh/ (GREF/.C.) in BHA2, and the output goes to the RFBN which operates in the shift mode (see figure 2-143).

The first clock enables RFBN(P), which is fed into BHA2(D). CLR2/.C. is true, enabling the output. CLR2..C. remains true up to the fourth clock when it disables the clear output. This output is ORed with PWR UP/, causing a clear during power-up.

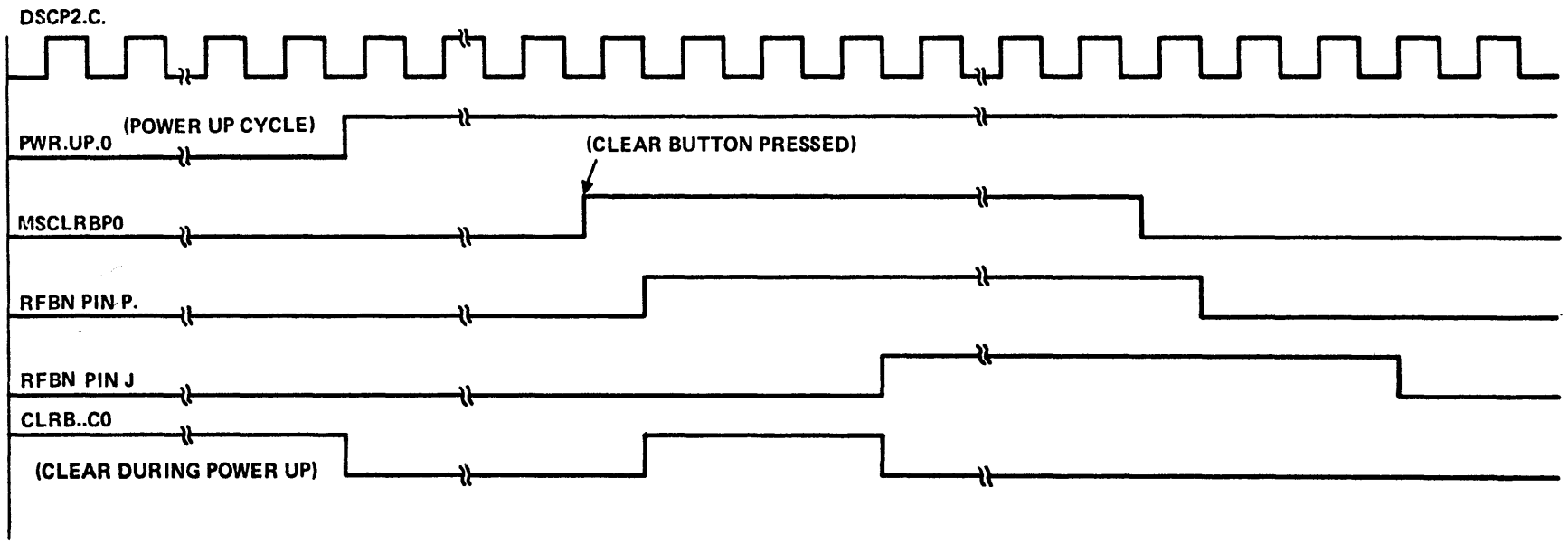


Figure 2-143. Power-up and Clear Signal Timing Chart

HOLD REGISTER

The hold register is 24 bits wide, and is used for temporary storage of incoming write data. See figure 2-144. The register consists of six RFBN chips which operate in the D-set mode only. Input to the hold register is from the address and write bus (AW00B through AW23B), with the output going to the rotator.

During the third clock of a write or swap operation, the write data is available at the register input, with HRCLK.C. enabling the RFBNs. The outputs of the hold register (WT00HRC1 through WT23HRC1) go to the rotator during the fourth clock of the micro.

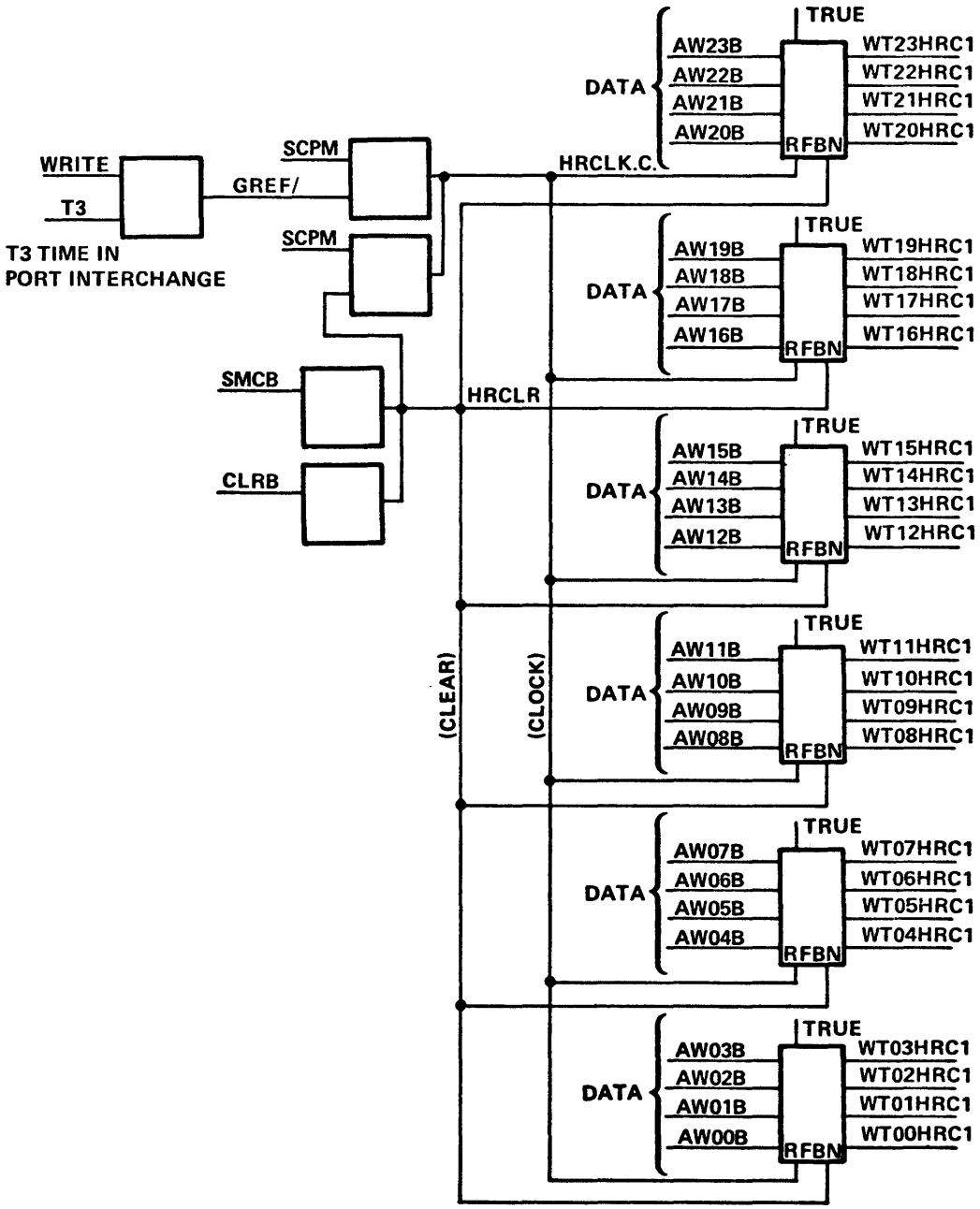


Figure 2-144. The Hold Register

MASK GENERATOR

The mask generator is a device for producing from 1 to 24 one bits, with the number actually produced being equal to the specified data field length of the memory cycle being performed. The true outputs are consecutively applied in accordance with the field length, from the least significant to most significant bit positions. For example, if the field length is 18, the mask generator produces 18 true levels in bit positions 00 through 17 (WR00HRC1 through WR17HRC1). Because the enabling signals must coincide with the placement of data in memory, the output of the mask generator is passed through the rotator before being stored in the mask register.

Operation of the mask generator (figure 2-145) is like that of the similar circuit in the 24-bit function box, employing DFAN chips driving CFBN chips, the latter of which are operated in the additive mode. In this case, inputs to the circuit are the signals DFL01.B1, DFL02.B1, DFL04.B1, DFL08.B1, and DFL16.B1, which comprise the binarily-encoded field length. These inputs are present during the first and second clocks of all memory operations. MSKSTBC0 is the level which enables the generator at the second clock for a write operation, during the second and third clocks for a read operation, and during the second and seventh clocks for a swap operation. MSKCMPC0, when true, forces a true output on all mask bit lines. This signal can occur on the third clock pulse of a read operation or the seventh clock of a swap, but never occurs during a write cycle.

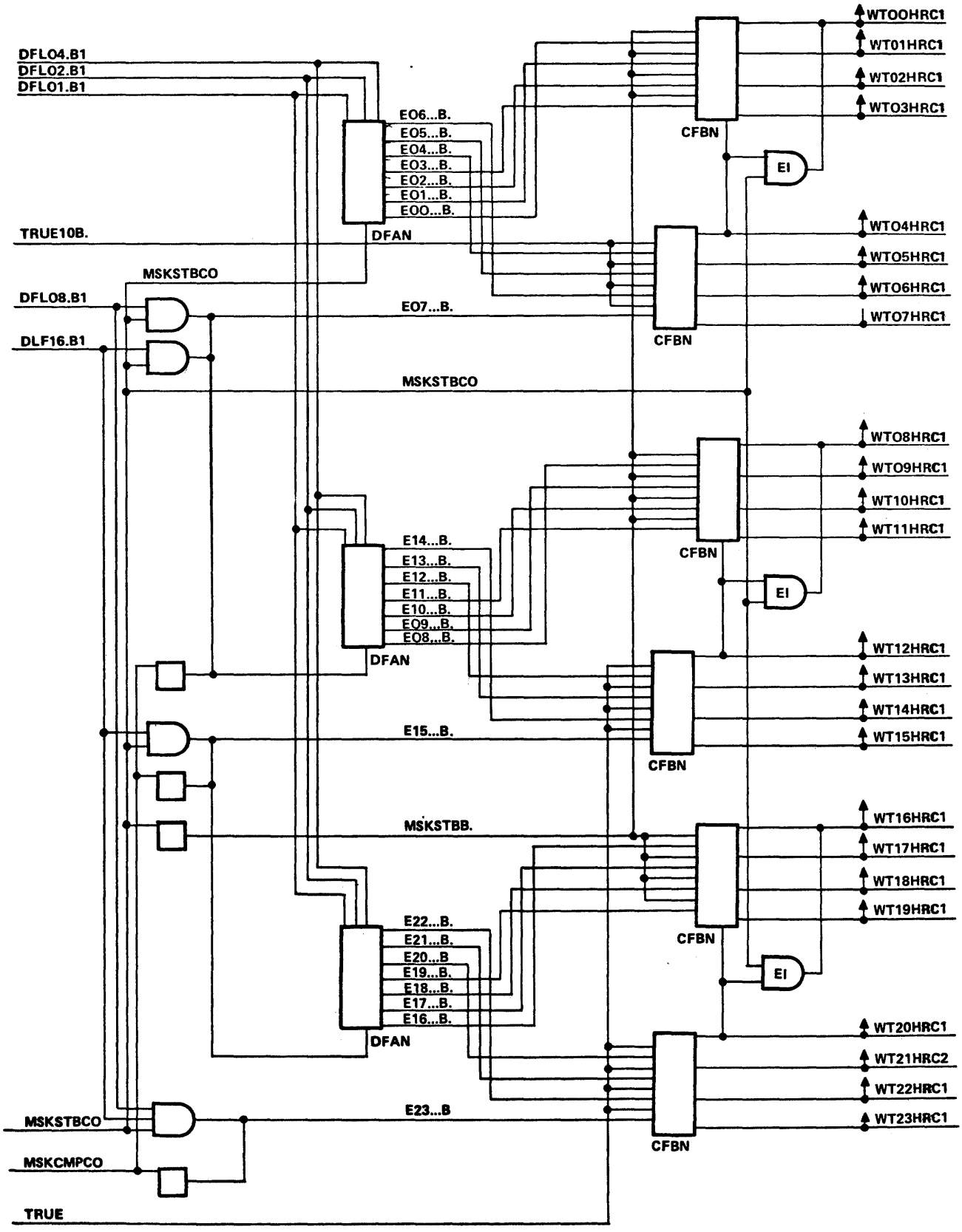


Figure 2-145. The Mask Generator

MASK REGISTER

The mask register is 32 bits wide, corresponding to the normal field length of a memory cycle (32 bit locations in memory are automatically accessed, regardless of the field length specified in the operation). Of these 32 bits, a maximum of 24 may be read or written in a given memory cycle, with the portion actually selected depending on the bit address as isolated by the rotator. The mask bits themselves are used to select gating of read or write data to the output of the merger, on a bit-for-bit basis. True and false mask bits allow write and read data to the output, respectively.

The mask register (figure 2-146) is composed of RFBN chips which operate in the D-set mode for write operations, and the complement mode for reads. This is done to place true mask bits in the desired bit positions for a write (where new data is overlaying old), and false mask bits in the desired positions for a read (where the data read from memory is desired as the output). In the latter case, zero fill occurs in those bit positions where a true mask bit is present. This happens because no write data is present at the input to the merger at this time.

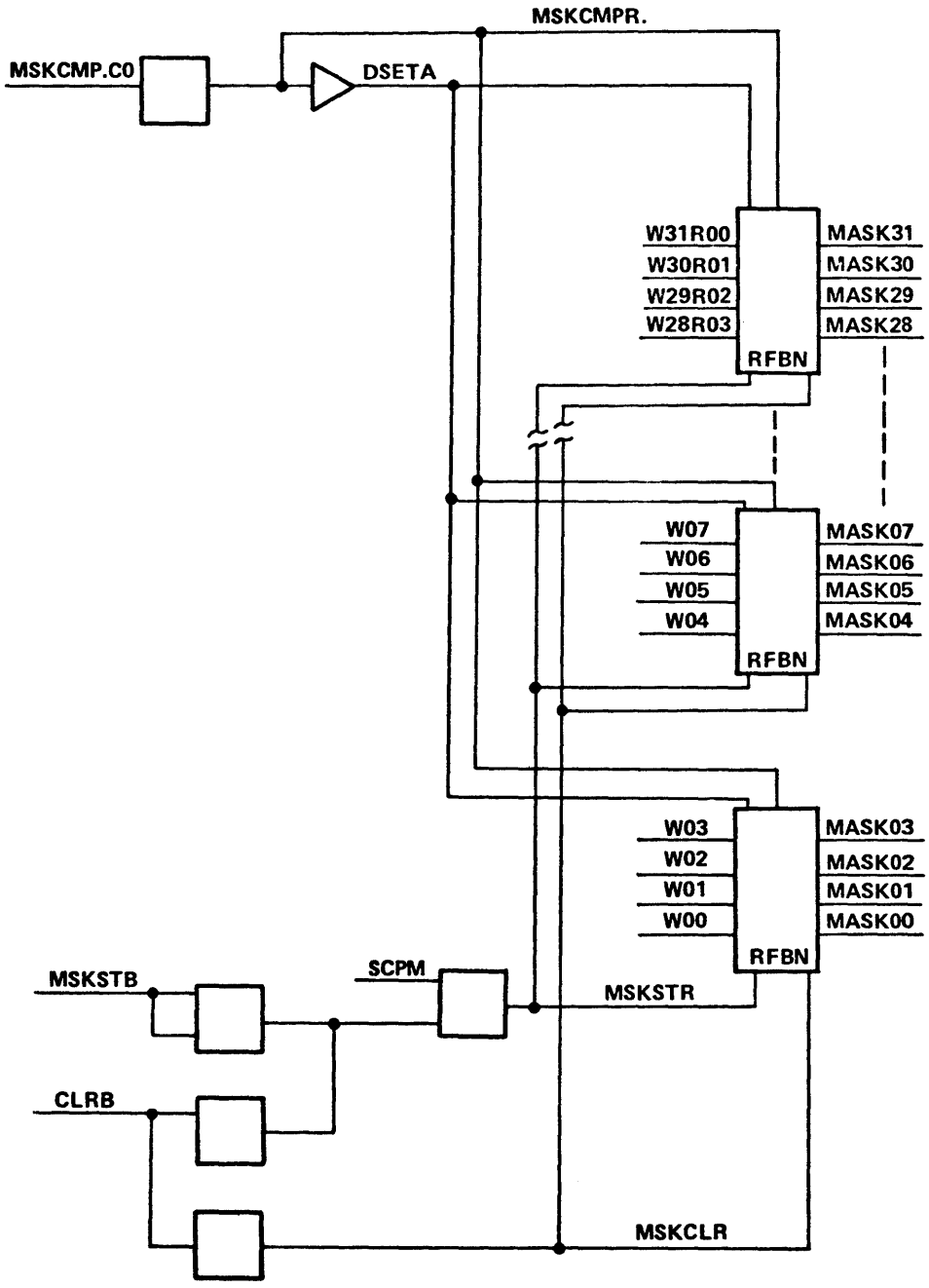


Figure 2-146. Mask Register

MEMORY ADDRESS REGISTER (MAR)

The memory address register (figure 2-147) is 21 bits wide, and is used to store the starting address in memory which is to be read/written/swapped. When used with the 7C or 2D micros to address data, the contents of the FA register are transferred to MAR via the Cnnl and AWnnB lines. MAR stores the address lines AW03 through AW23, which serve to identify the key byte address. The address lines AW00, AW01, and AW02 have no corresponding bits in the MAR register, since they serve to identify the bit address. These, along with AW04 and AW04/ go directly to the control logic for the rotator, which serves to perform the bit addressing function.

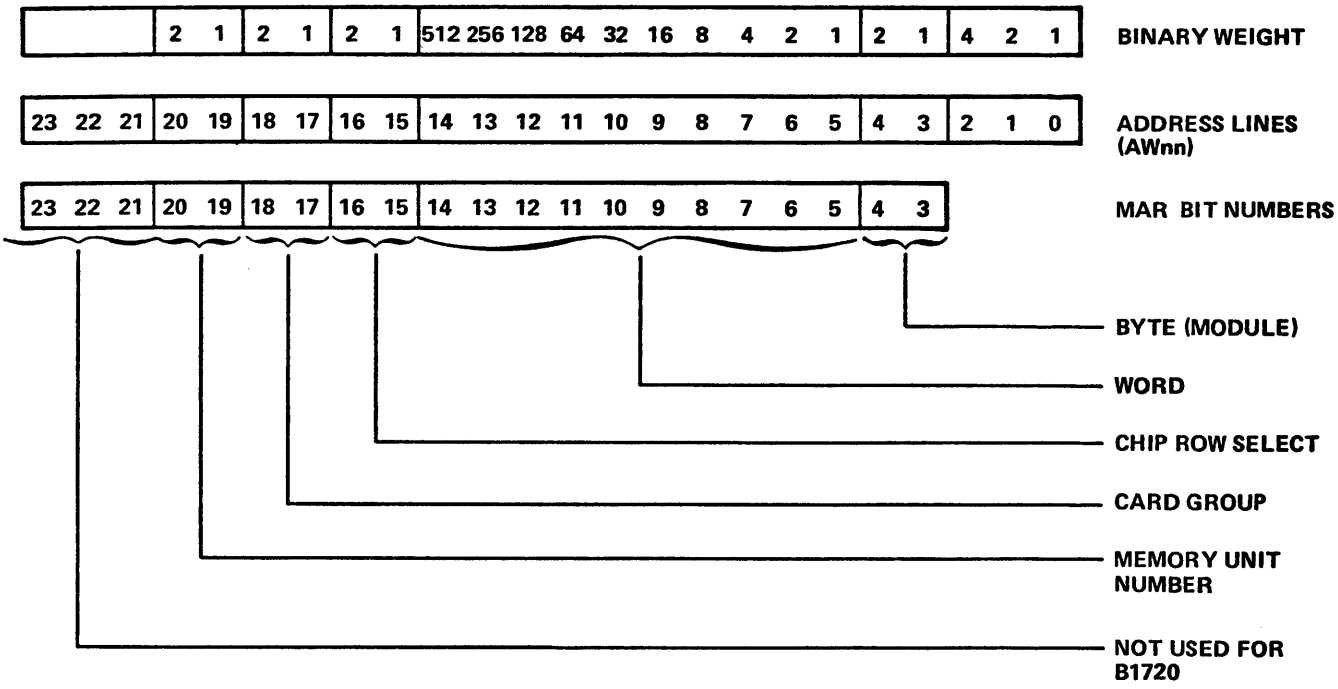


Figure 2-147. The Memory Address Register Structure

MAR consists of 11 FFAN chips (figure 2-148) which operate in the D-set mode only. Note that one FFAN is shared with the field direction sign (FDS). The incoming address data is stored in MAR during the first clock of a memory access micro, being enabled by SMGB, which places the flip-flops in the D-set mode. The data is clocked into the flip-flops by the MAR clock, (MARCLKC.) which may follow either the system clock or an adjustable early clock. This selection is made on a JPRN (at chip location LO) on port interchange card C. Generation of MARCLKC. is shown in figure 2-149.

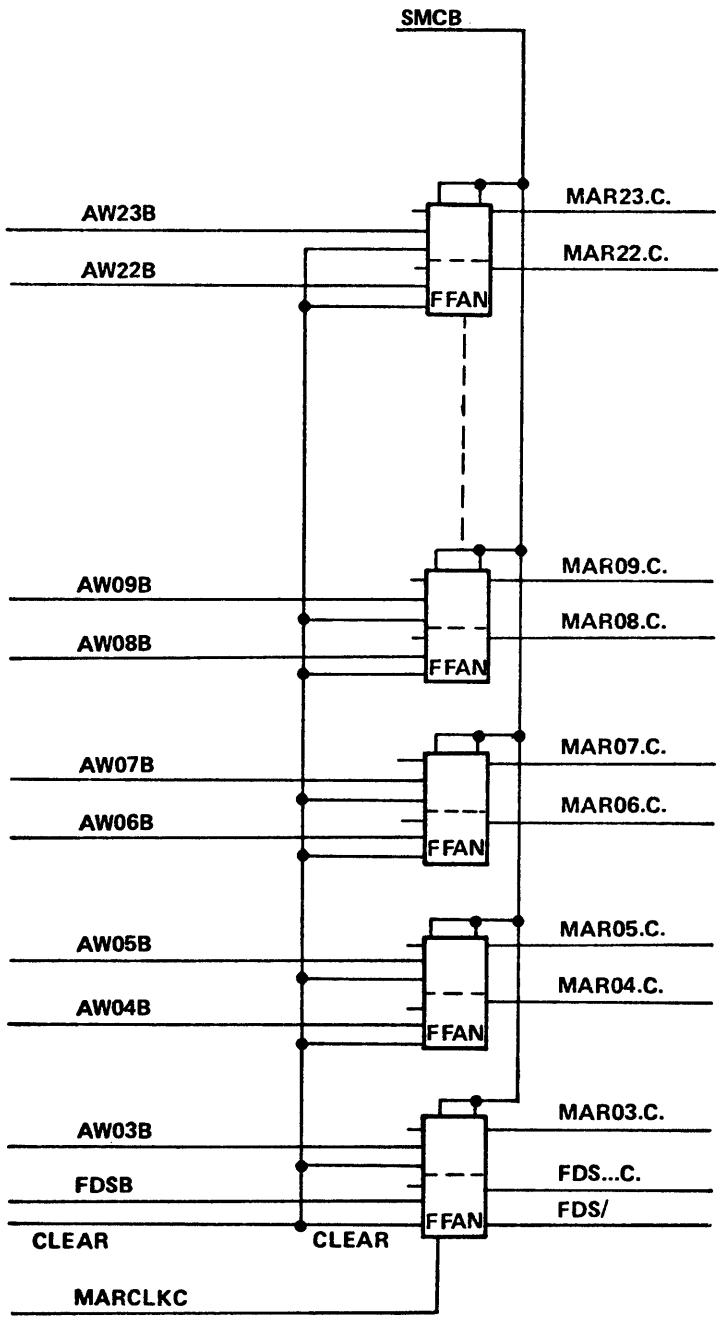


Figure 2-148. Memory Address Register (MAR)

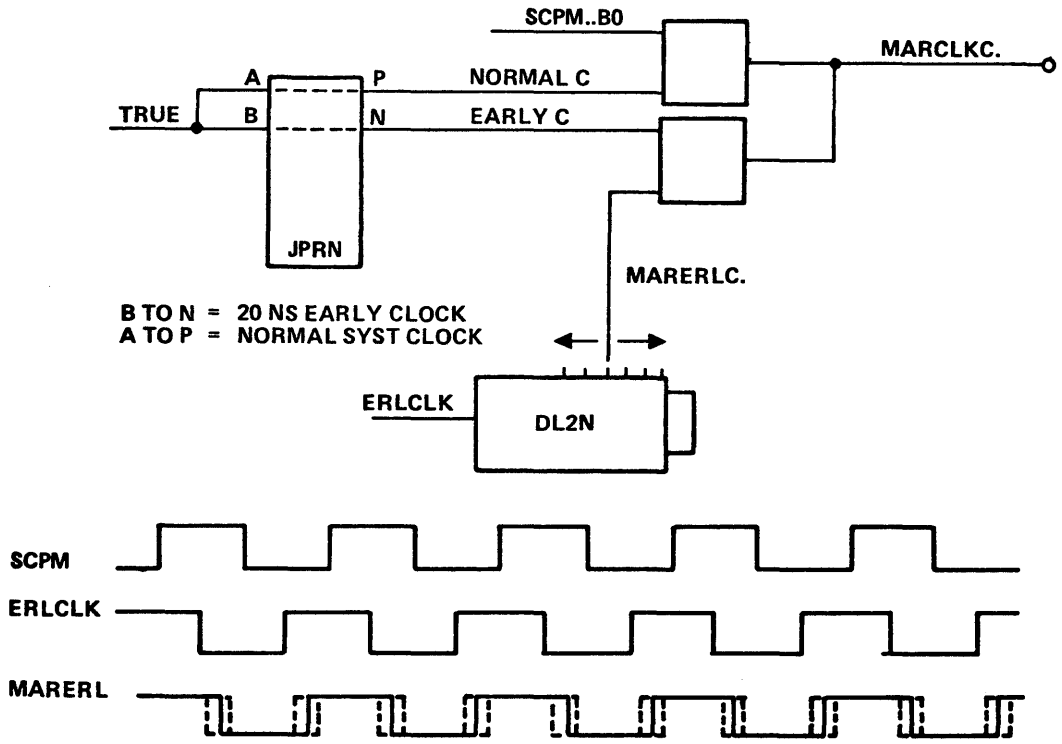


Figure 2-149. The MAR Variable Clock

READ MEMORY INFORMATION REGISTER (RMIR)

The read memory information register is 36 bits wide, and serves to receive and temporarily store the data from memory during each memory access operation. Thirty-six bits of storage are required to accommodate 32 data bits with their accompanying 4 parity bits, the latter being stored in memory in the same manner as data.

RMIR consists of 12 RFBN chips (figure 2-150), 8 of which are used for storing data, and 4 for parity. The chips are wired so as to always operate in the D-set mode. The read information arrives from memory on the MRnn lines and is strobed into the register on the fourth clock pulse of the memory operation by RSTR.0., RSTR.1., RSTR.2., and RSTR.3. for bytes 0, 1, 2, and 3, respectively. Separate strobe signals are provided for the individual bytes because a difference in data propagation delay may occur when the read information comes from two different memory units (the data field selected straddles a physical division of memory). In such cases the strobe signals affecting the numerically higher unit of memory are delayed to allow for the longer propagation time. This is illustrated in figure 2-151. Normally, all four strobes occur at the same time. Delayed strobe generation logic is shown in figure 2-152. Note that it has not been necessary to utilize the delayed strobe provision in memories of four units (256K bytes) or less. Such use may be necessary if larger memories are assembled, especially if there is a significant physical separation (electrically) between adjacent units.

The output of RMIR goes to the merger via the RMIRnnA lines during the fifth clock period of a write or swap operation, or following the fourth (and last) clock of a read operation. The four parity bits go to a parity check circuit, where if bad parity is found, the parity bus line is forced true.

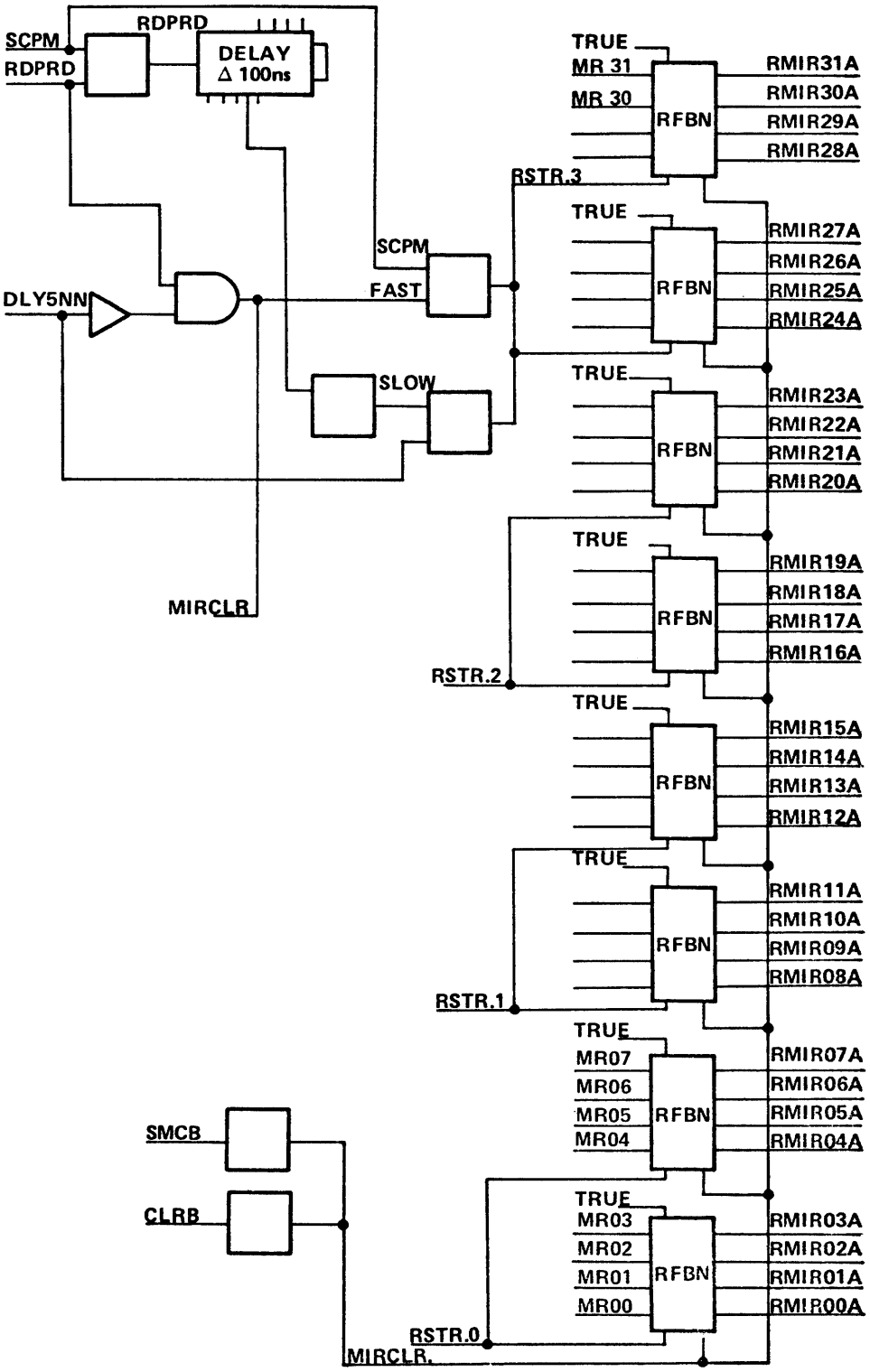
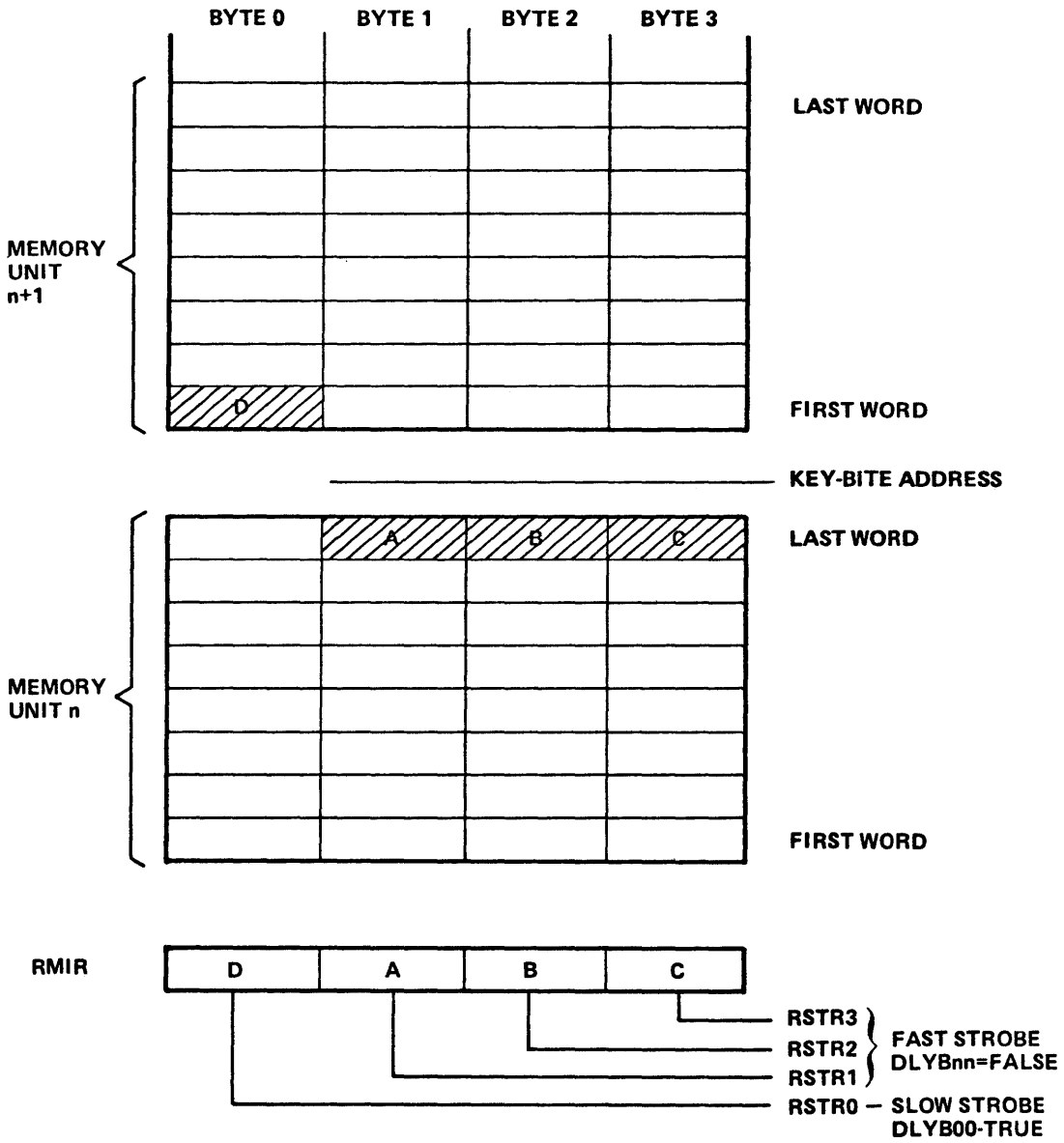


Figure 2-150. The Read Memory Information Register (RMIR)



NOTE: THE DATA BYTES A, B, AND C USE THE FAST STROBE. THE DATA BYTE D USES THE SLOW STROBE BECAUSE IT COMES FROM MEMORY UNIT n+1 WHICH HAS A PROPAGATION TIME GREATER THAN MEMORY UNIT n. THIS IS DONE BECAUSE THE READ DATA FROM THE MORE DISTANT UNIT TAKES LONGER TO REACH RMIR, AND THEREFORE IS NOT PRESENT AT THE TIME OF THE FAST STROBE.

Figure 2-151. RMIR Strobe Selection

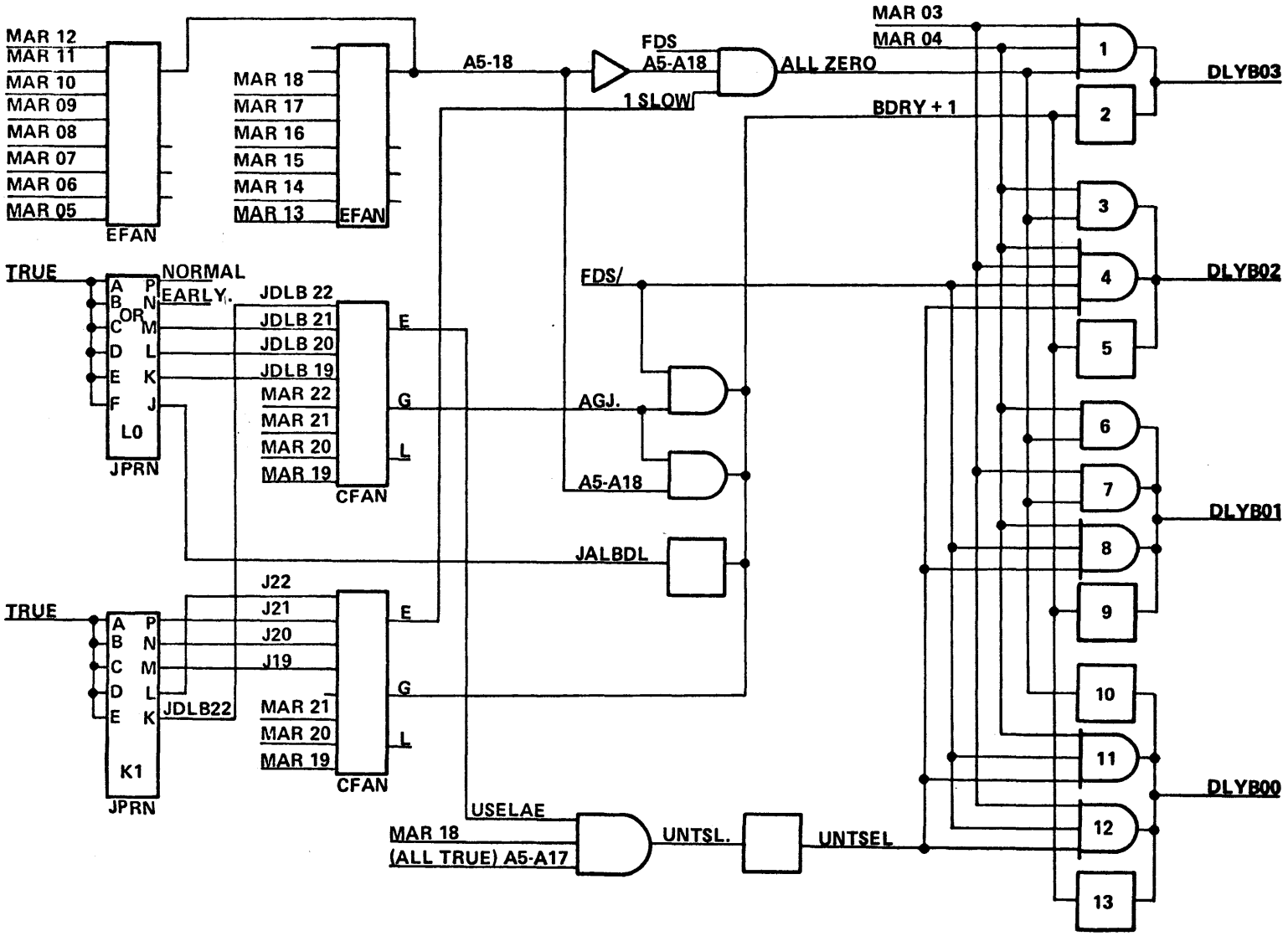


Figure 2-152. The Delayed RMIR Strobe Generation

MERGER

The merger is a network of gating elements which serves to merge two data sources. This process involves producing an output data field composed of selected portions of the two input data fields. Merging is required as part of the bit addressability scheme, which makes use of a rotator stage. Inputs to the merger are as follows:

a. Write data from the hold register (WnnRnnBO).

b. Mask bits from the mask register (MASKnnA.).

c. Read data from RMIR (RMIRnnA.).

Note that the write data passes through the rotator before entering the merger.

For a write operation, the incoming write data (maximum 24 bits) is inserted within a field of 32 bits read from memory, replacing the contents thereof in the bit positions which it occupies. The combined output is written into memory as gated from the merger.

For a read operation the merger is used to either allow or block the passage of read data to the rotator, and then to the output of the port interchange.

The merger consists of 64 AND gates and 32 inverters, with 2 AND gates and 1 inverter being used at each bit position (see figure 2-153). During write operations, true mask bits allow the rotator output bits (WnnRDO) to the output of the merger, and false mask bits allow RMIR bits to the merger output. Merger operation is the same for read operations, with RMIR bits being allowed to the output if the respective mask bit is false. Although the rotator output bits are enabled by true mask bits just as during write operations, this is not significant because no data output from this source is present during read operations. The output of the merger is routed to the rotator, write buffers, and parity generation logic.

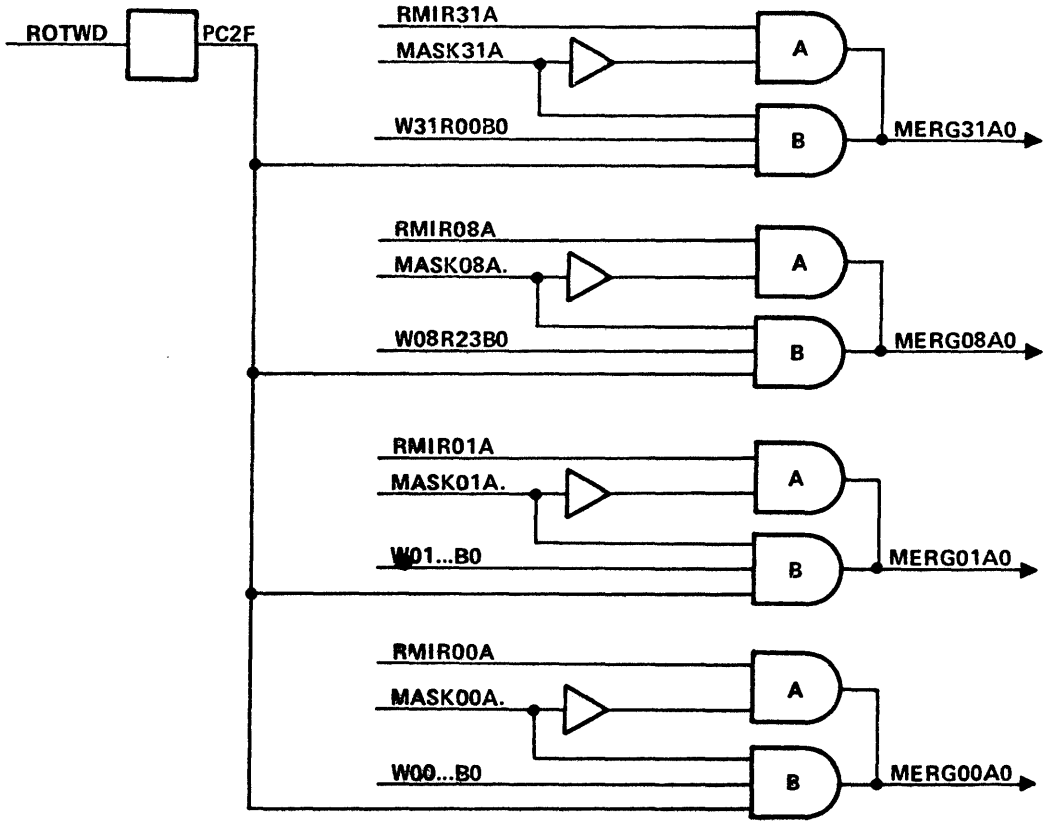


Figure 2-153. The Merger
(Four Bits Shown)

FIELD ISOLATION UNIT (ROTATOR)

The rotator is a two-stage multiplexing device, the purpose of which is to allow transfer of any given input data bit to any desired output position. This is, of course, done in parallel for all bits at once, with wrap around occurring automatically. The rotator can also allow data to pass through it without rotation. Inputs and outputs to the rotator are as follows:

a. Inputs: Hold register (write data).

Mask generator (mask bits).

Merger output (read data).

b. Outputs: Rotated write data to merger.

Rotated read data to the port interchange output.

The rotator consists of 64 MFAN multiplexor chips arranged in two rows of 32 each (see figure 2-154). The first stage provides a maximum shift of four bit positions, with eight provided by the second. This gives a maximum possible total shift of 32 positions (4 x 8) for the rotator. In addition to providing a shift of up to 4 bits, the first stage of the rotator is used to select either the hold register/mask generator input or the Merger input. Input selection is accomplished by feeding the MERGnnA0 lines to inputs 0 through 3 and the WTnnHRC1 lines to inputs 4 through 7 of each MFAN chip in the first stage. Having done this, the binary weight 4 control input (ROTWD.) is used to choose between them, with WTnnHRC1s enabled when true and MERGnnA0s when false.

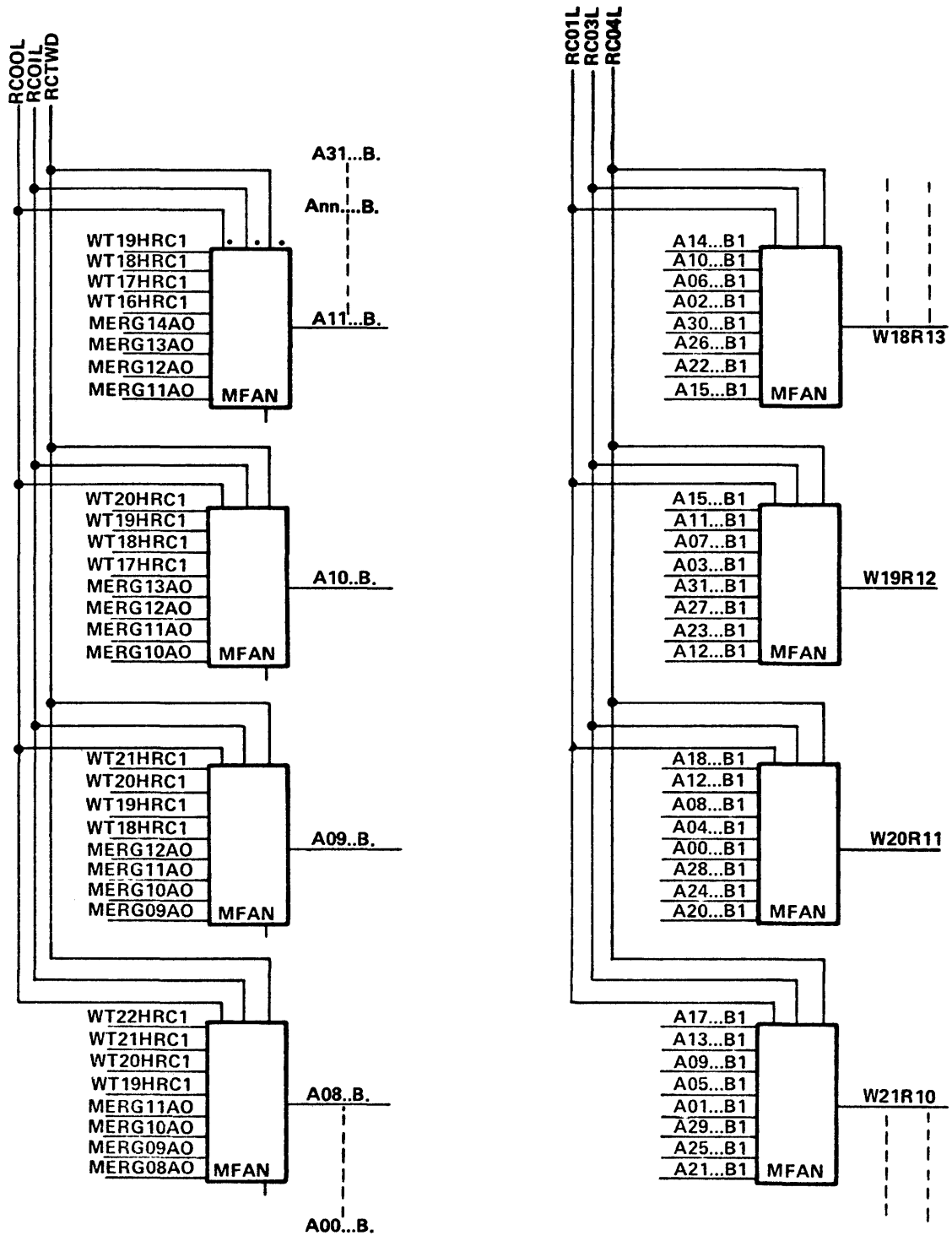


Figure 2-154. The Rotator

It should be noted that the rotator is a right rotator. For example, if a left rotation of eight positions is required, the rotator actually rotates the data 24 bits of the right, which in effect is equivalent to a left rotate of eight.

Rotator Control (Card B, Pages 1 and 2)

The Rotator control lines have the following binary values for rotation. Refer to figure 2-155.

Control Line	Binary Value	
RC00L	1	To 1st Stage
RC01L	2	
ROTWD	WR (selects input)	
RC02L	4	To 2nd Stage
RC03L	8	
RC04L	16	

The rotator control lines are derived from the field length (FL01BP0 through FL16BP0), the five least-significant bits of the address (AW00B.P0 through AW04B.P0), and the field direction sign (FDSB..P0).

The rotator control information is loaded into a register (vector register) with start memory cycle (SMCB..B.) and is then available to the input of the rotator control adder. The adder consists of 3 AFANs and one CFBN operating as a 5-bit adder. The adder output is the sum of the DFL 1, 2, 4, 8, 16, and the A00, A01, A02, A03, and A04 addresses. This sum is called RC00L, RC01L, RC02L, RC03L, and RC04L. Finally, the lines used for rotation are RC02L through RC04L.

For a read operation (ROTWD false):

RC02 → RC02L

RC03 → RC03L

RC04 → RC04L

For a write operation (ROTWD true) the complement is used:

RC02/ → RC02L

RC03/ → RC03L

RC04/ → RC04L

The lines RC00L and RC01L are not modified. See table 2-13.

Table 2-13. Examples of Rotations

Input		First Stage Out		Second Stage Out	First Stage RC001 1	First Stage RC01L 2	First Stage ROTWD W	Second Stage RC02L 4	Second Stage RC03L 8	Second Stage RC04L 16	Number of Shift
		For Read									
MERG00	=	A00	=	W00 ...	0	0	0	0	0	0	0
MERG08	=	A08	=	W08 R23							
MERG23	=	A23	=	W23 R08							
MERG31	=	A31	=	W31 R00							
MERG23	=	A22	=	W22 R09	1	0	0	0	0	0	1
MERG23	=	A23	=	W19 R12	0	0	0	1	0	0	4
MERG23	=	A23	=	W15 R16	0	0	0	0	1	0	8
MERG23	=	A23	=	W07 ...	0	0	0	0	0	1	16
		For Write									
WT23HR	=	A04	=	W08 R23	0	0	1	1	1	1	28
WT00HR	=	A27	=	W31 R00							
		For Write									
WT19HR	=	A08	=	W20 R11	0	0	1	1	0	1	20
WT18HR	=	A09	=	W21 R10							
		For Read									
MERG11	=	A08	=	W20 R11	1	1	0	1	0	1	23
MERG12	=	A09	=	W21 R10							

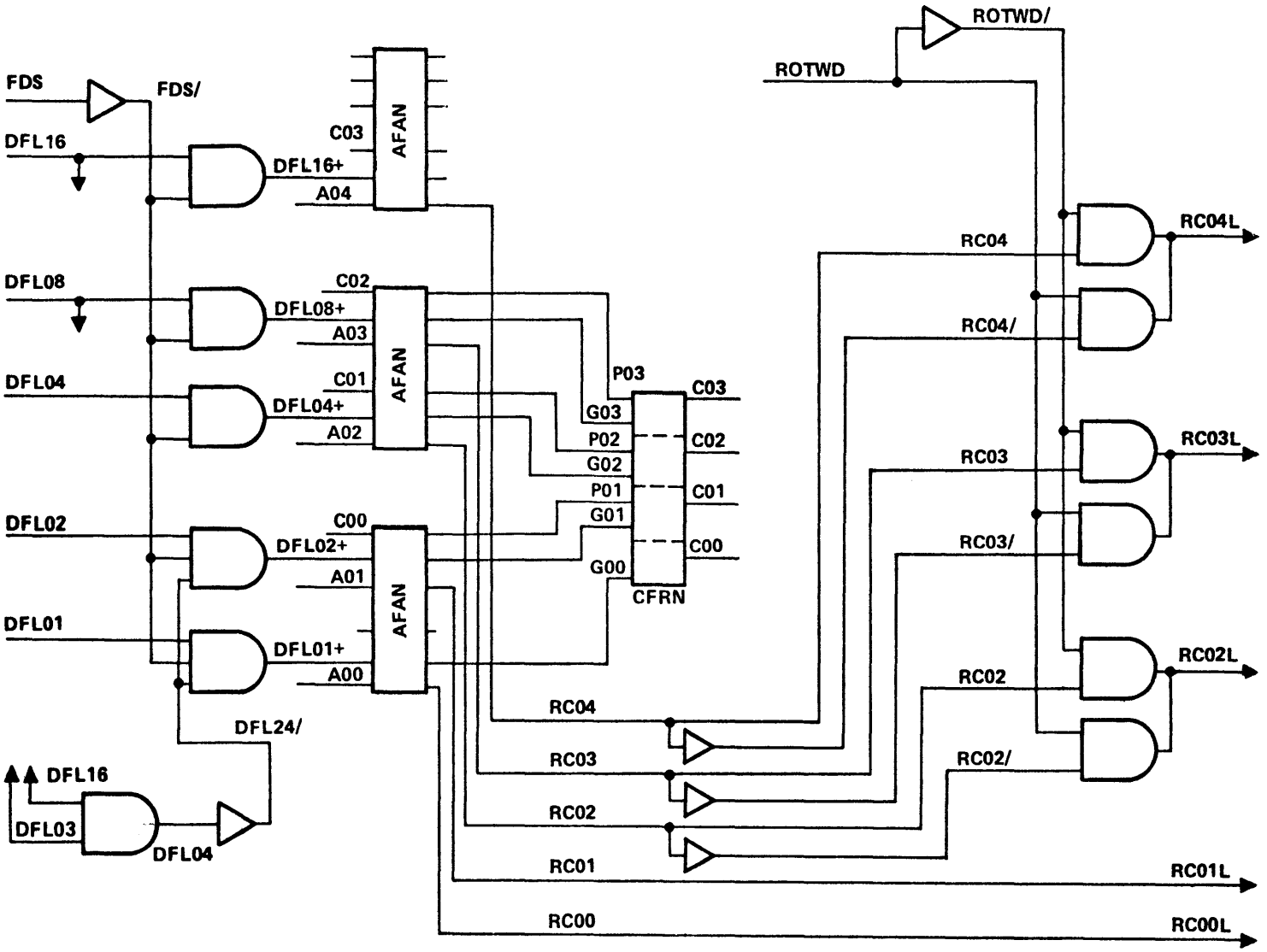


Figure 2-155. The Rotator Control Signal Generation

MEMORY CYCLE TIMING COUNTER

The memory cycle timing counter (figure 2-156) supplies clock signals for use by the port interchange. Its outputs are eight separate, single (system) clock length pulses which occur in sequential order. These are identified as T1....C1 through T8....C.. The memory cycle timing pulses are used for synchronization throughout the port interchange.

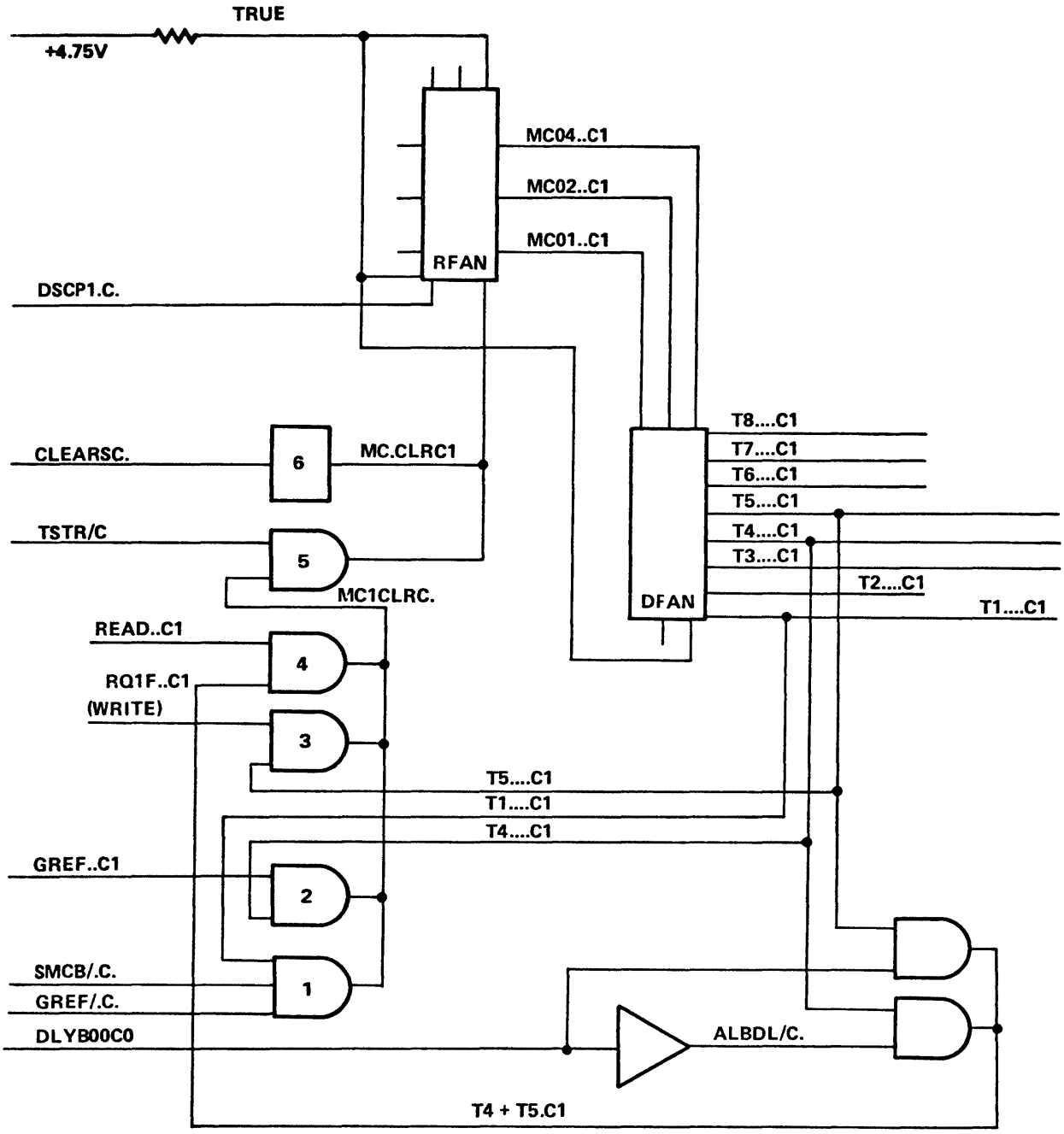


Figure 2-156. Memory Cycle Timing Counter

The counter consists of an RFAN operating in the add mode, followed by a DFAN decoder which produces a separate output for each binary increment. Although the counter follows the system clock, it is started and stopped by external action. This is accomplished by the gates driving the RFAN clear input. Gate 1 in figure 2-156 is used to start the timer. Under normal conditions the timer is reset, with T1....C1, SMCB/.C., and GREF/.C. all true, thus holding the clear input true. A cycle is initiated by either SMCB1.C. (start memory not) or GREF/.C. (granted refresh not) going false. This removes the clear level, allowing the timer to increment at each system clock pulse. Gates 1, 2, 3, and 4 are used to halt and reset the timer by forcing the clear line true again:

Gate 1 stops the timer when it wraps around to zero and T1....C1 comes true again.

Gate 2 stops the timer at T4....C1 during a refresh cycle.

Gate 3 stops the timer at T5....C1 during a write operation.

Gate 4 stops the timer at T4....C1 during a read operation.

Gate 5 is used by the card tester.

Gate 6 is used for the general processor clear operation.

Note that gate 4 can also stop the timer at T5....C1 if DLYBOOCD is true. This level is not used in present versions of the system (level is permanently false), but may be employed in the future when larger memories are installed. Counter timing is illustrated in figure 2-157.

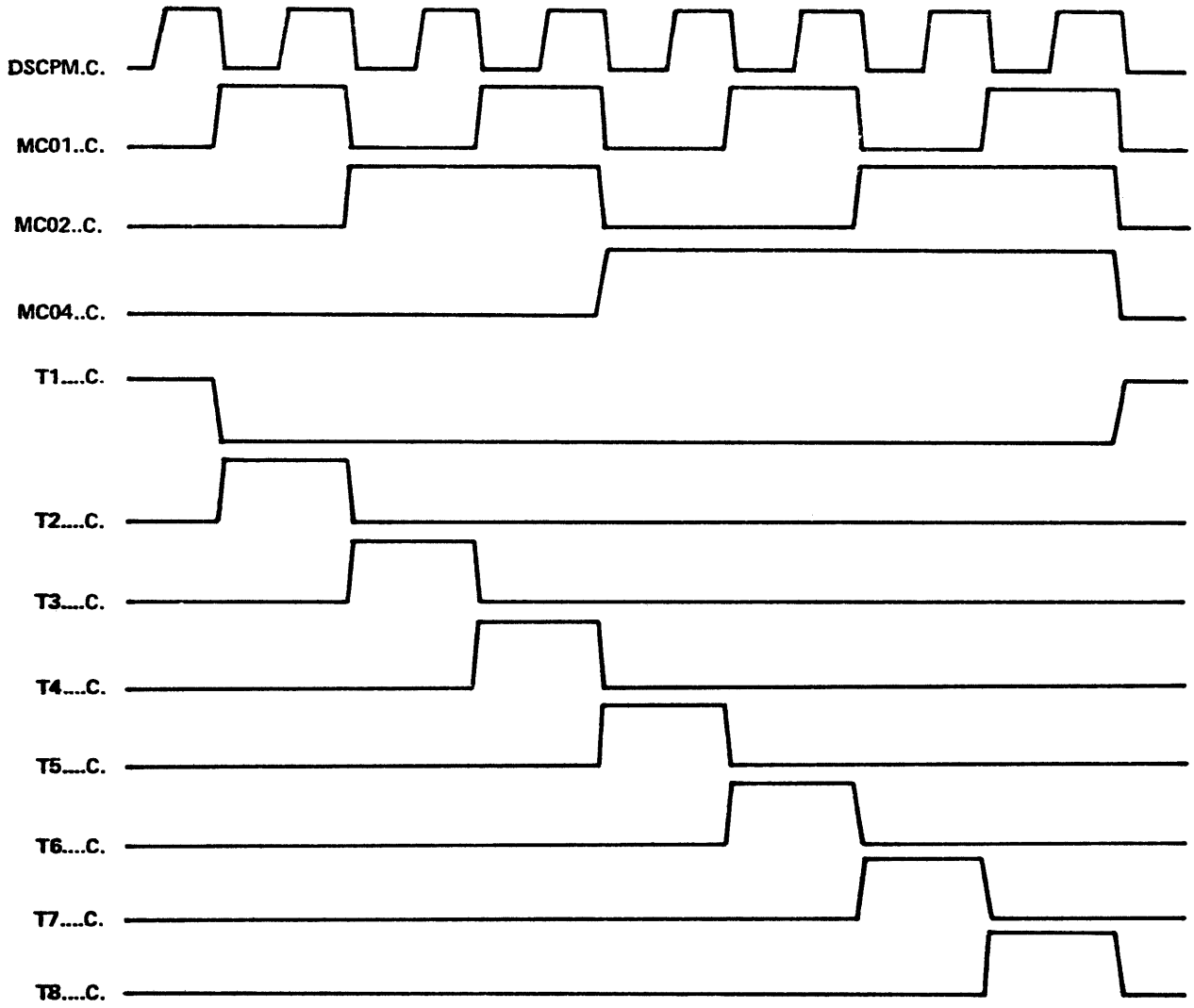


Figure 2-157. Memory Cycle Counter Timing Chart

PRIORITY RESOLUTION LOGIC

Since each port device can request memory access at any time, some means of determining access priorities is required. This is accomplished by priority resolution logic, which receives the memory request signals (RQ4) from each of the port devices. The priorities are assigned by request line number, with the highest request line which is true taking precedence. Present convention assigns line numbers as shown in figure 2-158.

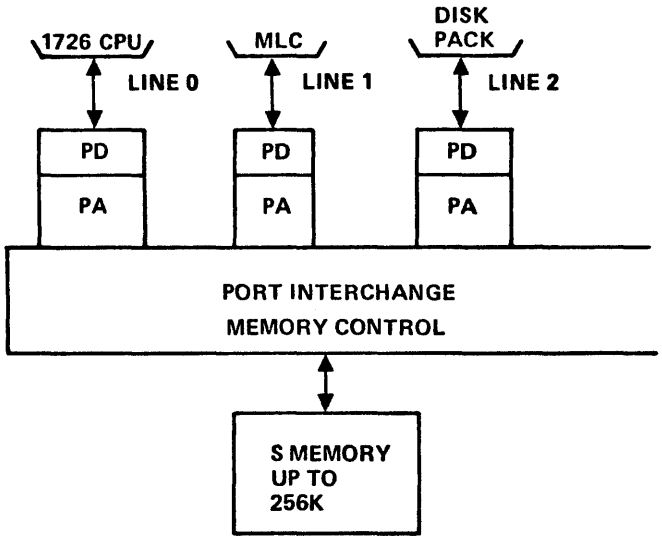


Figure 2-158. Priority Line Assignment

The priority resolution logic consists of an EFAN priority encoder driving a DFAN decoder. Inputs to the encoder are the RQ4 lines from each of the port devices (RQ4L0.P0 through RQ4L4.P0). Note that more than one port device need not be present for the circuit to function. Assuming that port devices are connected to lines 0 and 1, and the RQ4 signal is received from both simultaneously, priority automatically is given to line 1 by EFAN K8. This causes the outputs PG1...C. and ANYPRQC. to be true. The former signal is an input to the decoder, J8. For the DFAN to produce an output, GREF..C1 (generate refresh) must be false, and the priority enable flip-flop G7 (figure 2-159) must be set (PEF...C. true). This flip-flop is set each time a memory cycle is completed, and is reset by T1....C. after starting a new memory cycle (SMCB..C0). With PEF..C1 true, decoder J8 is enabled, and because only PG1...C. is true, the level PG1...C0 (priority granted line 1) is true. This priority line is routed through the backplane to port adapter 1.

The signal ANYPRQC (any priority request) generates SMCB..C0 at T1....C. time, via gate 1 and buffer H7. In addition to resetting the priority enable flip-flop, SMCB..C0 is routed to all port adapters. This signal is used in conjunction with the priority granted signal to enable the port adapter. (Generates PAE...P. in that particular port adapter.)

Note that gate 3 in figure 2-159 is used to set PEF..C1 if GREF appears at the end of a dispatch read and clear or at the end of an overlay micro. This can occur when a high priority refresh is required. Timing for gates 3 and 4 is shown in figure 2-160.

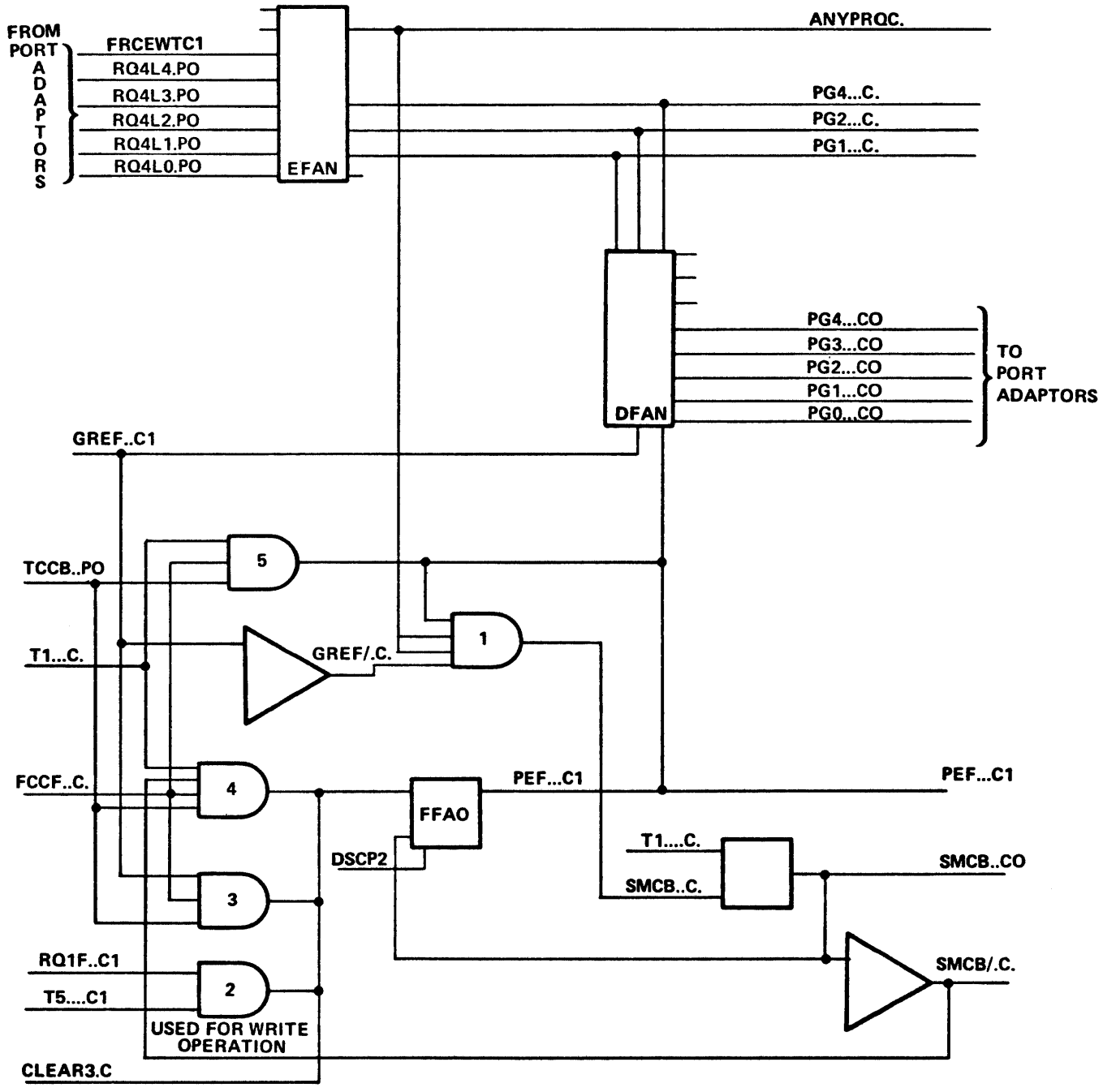


Figure 2-159. The Priority Enable Flip-Flop

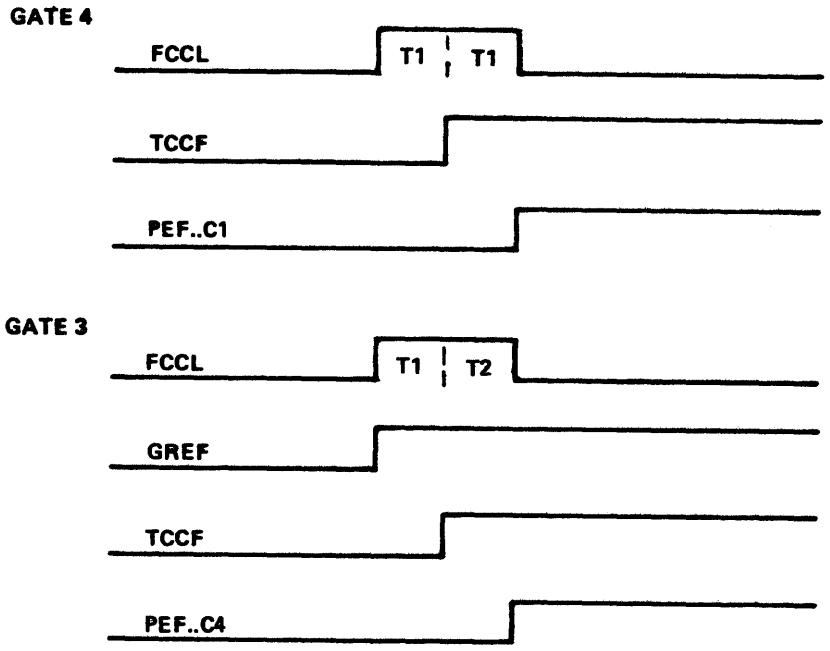


Figure 2-160. Priority Enable Flip-Flop Timing

PORT INTERCHANGE WRITE EXAMPLE

The signal SMCB (start memory cycle) causes the following levels to be true at the trailing edge of the first clock: FCCL, ROTWD, and MSKSTB. Refer to figures 2-161, 162, and 163. Also at this time the RMIR is cleared, the MAR receives the address and the counter starts.

During the second clock, enabled by MSKSTB, the mask generator develops the mask according to the DFL. The level ROTWD enables the mask generator to the inputs of the rotator. The mask, via the rotator, reaches the input of the mask register. At the end of the second clock the mask register is set.

During the third clock time the write data is present at the port interchange input, with the level PC2FB gating the data to the hold register inputs. The HRCLK is generated and at its trailing edge the data is placed in the hold register.

At the beginning of the fourth clock the merger receives the mask from the mask register, and the data from the hold register via the rotator. However, the read data from RMIR (to be rewritten) is not present yet. This data is present at the fifth clock.

The signal RDPRD, which occurs during the fourth clock, enables the generation of the RMIR STROBE (RSTR). At the end of RMIR STROBE the read data is present in the merger input. Now the merger output contains the four bytes of write data ready to be written into memory.

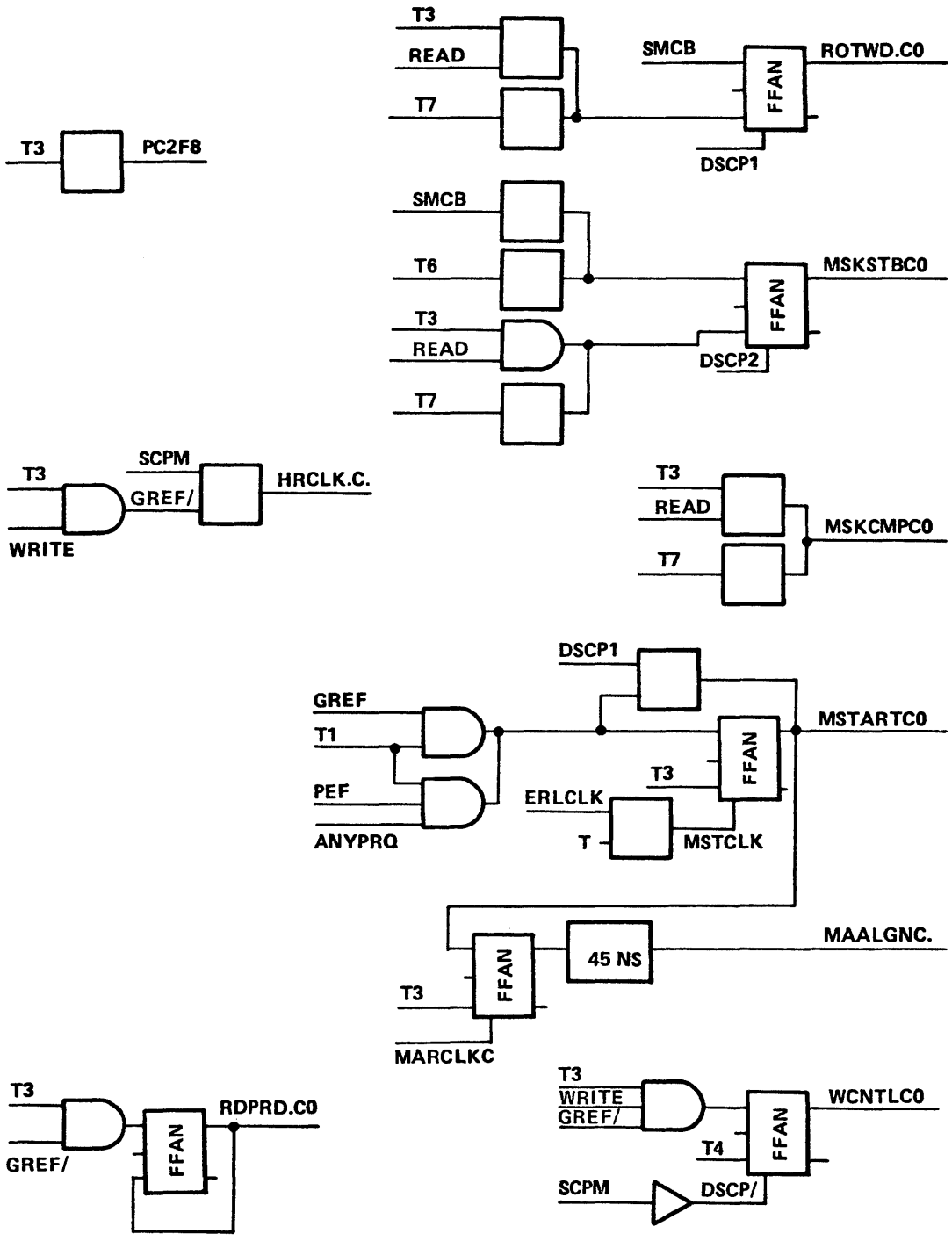


Figure 2-161. The Read/Write Control Levels

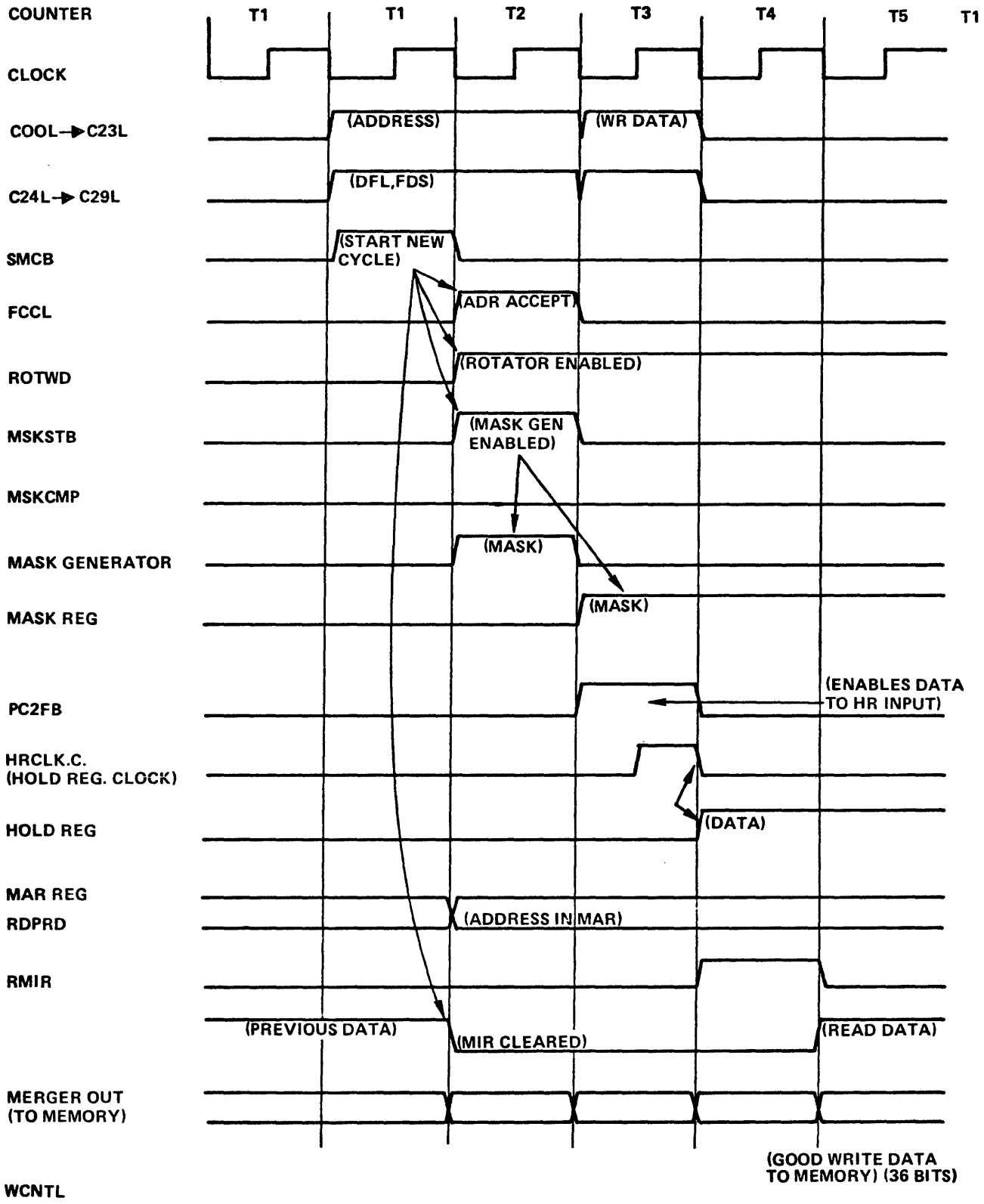


Figure 2-162. The Port Interchange Write Timing

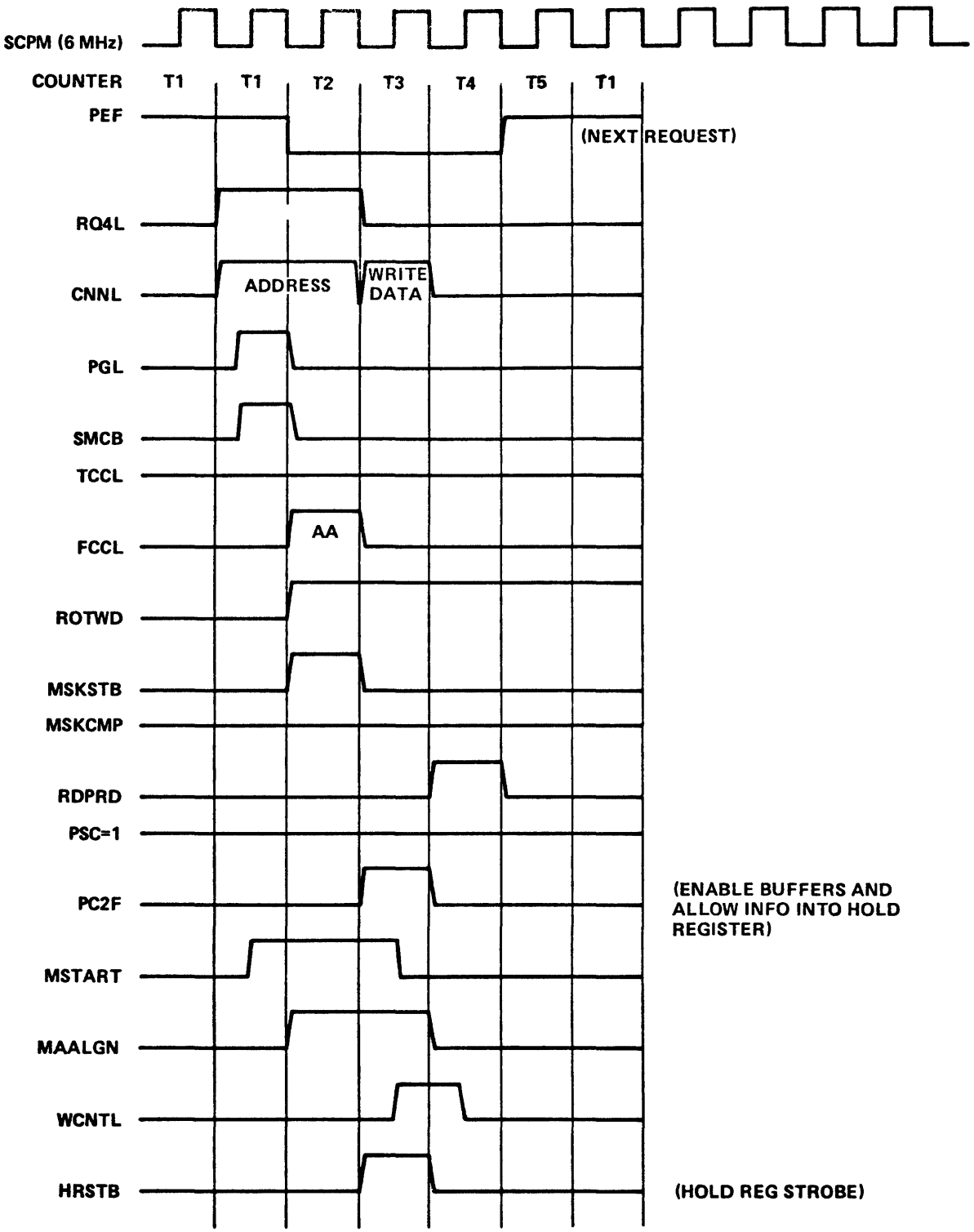


Figure 2-163. The Memory Write Timing

PORT INTERCHANGE READ EXAMPLE

The signal SMCB (start memory cycle) causes the following levels to be true at the trailing edge of the first clock: FCCL, ROTWD, and MSKSTB. Also, the RMIR is cleared, the MAR receives the address and the counter starts. Refer to figures 2-161, 2-164, and 2-165.

At the second clock, when MSKSTB is true and MSKCMP is false, the mask generator develops a mask according to the DFL.

The signal ROTWD enables the mask generator as input to the rotator. The mask, via rotator, reaches the input of the mask register, and at the end of the second clock the mask register is set.

During the third clock signal, MSKCMP comes true, causing the mask register to operate in the complement mode. Since MSKSTB and ROTWD continue true through the third clock, the mask register is complemented at the end of T3.

At the beginning of the fourth clock, the merger receives the mask complement from the mask register but the read data from RMIR is not yet present. This data is present at the trailing edge of T4.

The RDPRD is true during the fourth clock and enables the generation of the RMIR STROBE which is true during the second half of T4.

At the end of RMIR STROBE (beginning of T5) the read data is present in the merger input. Now the merger output contains the read data ready to be used by the port device. An example of the memory swap operation is not included. However, memory swap timing is shown in figure 2-166.

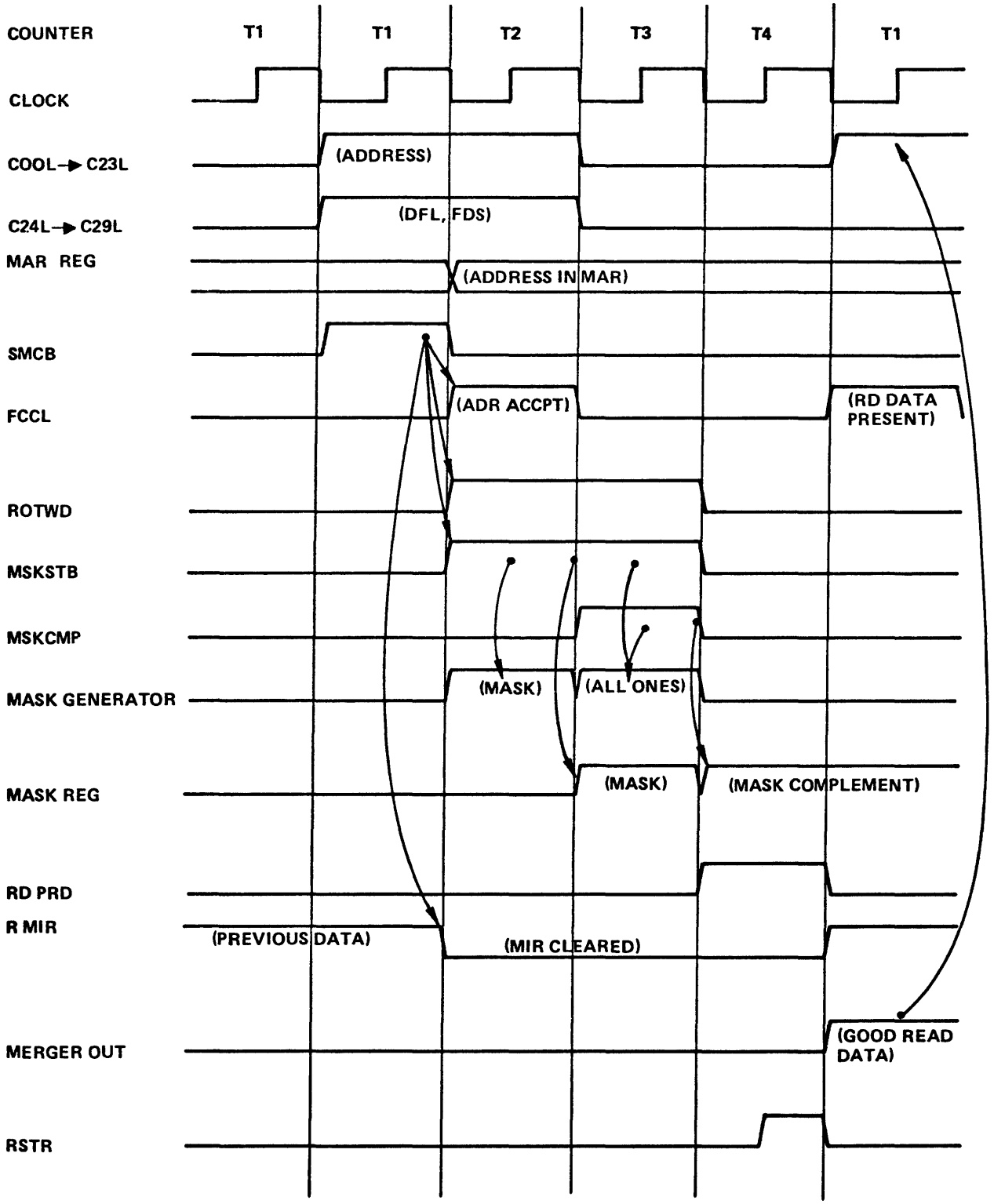


Figure 2-164. The Port Interchange Read Timing

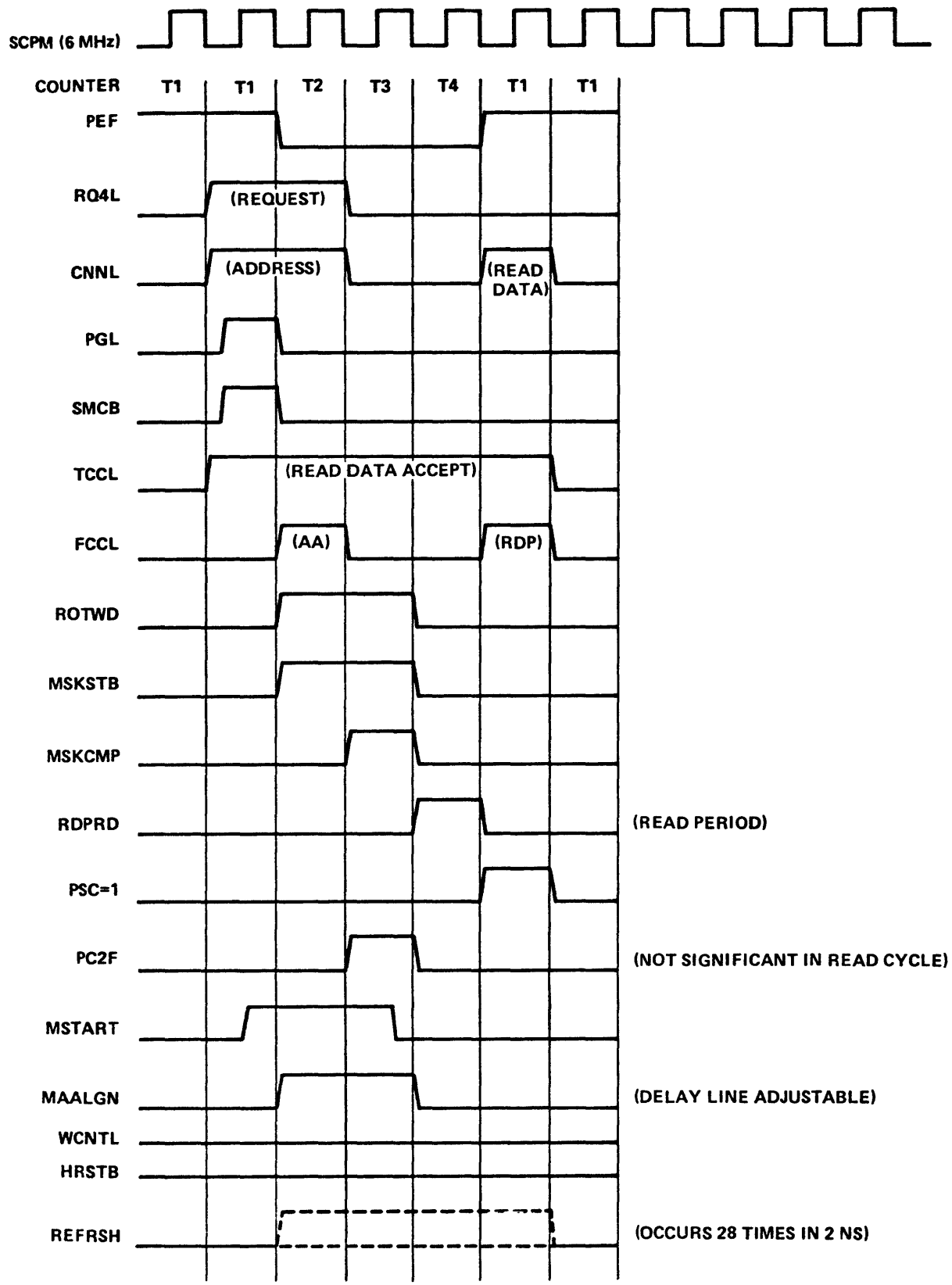


Figure 2-165. Memory Read Timing

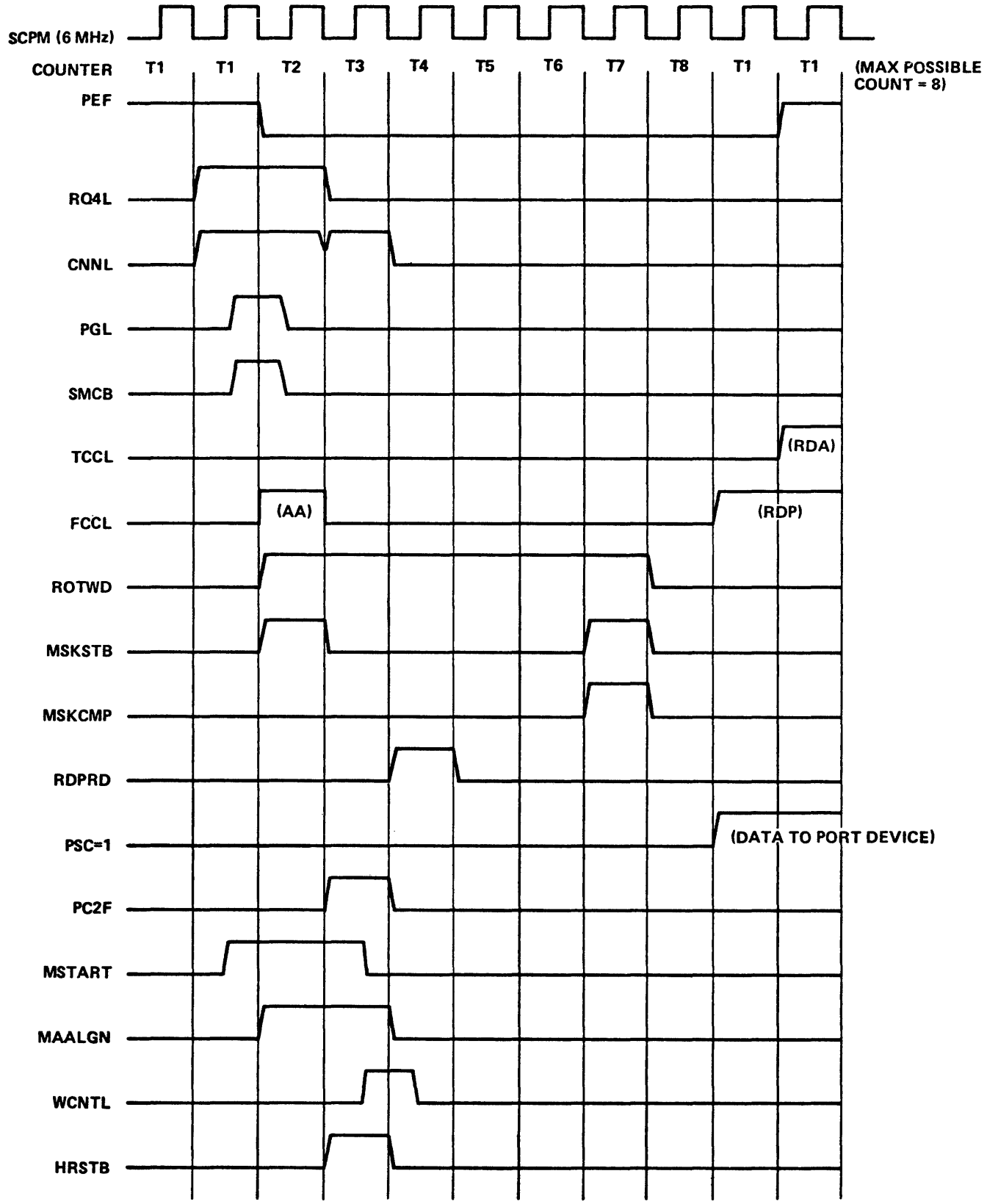


Figure 2-166. The Memory Swap Timing

DISPATCH REGISTER

The dispatch register (figure 2-167) is a 14-bit control register in the port interchange which is used to control port-to-port communication. The register itself stores control levels affecting such operations, while the actual message going from one port device to another is always stored in address zero of S-memory. In use the register accepts control information from a source port device during a write (dispatch) operation, and holds it until cleared by a read and clear operation performed by the destination port device.

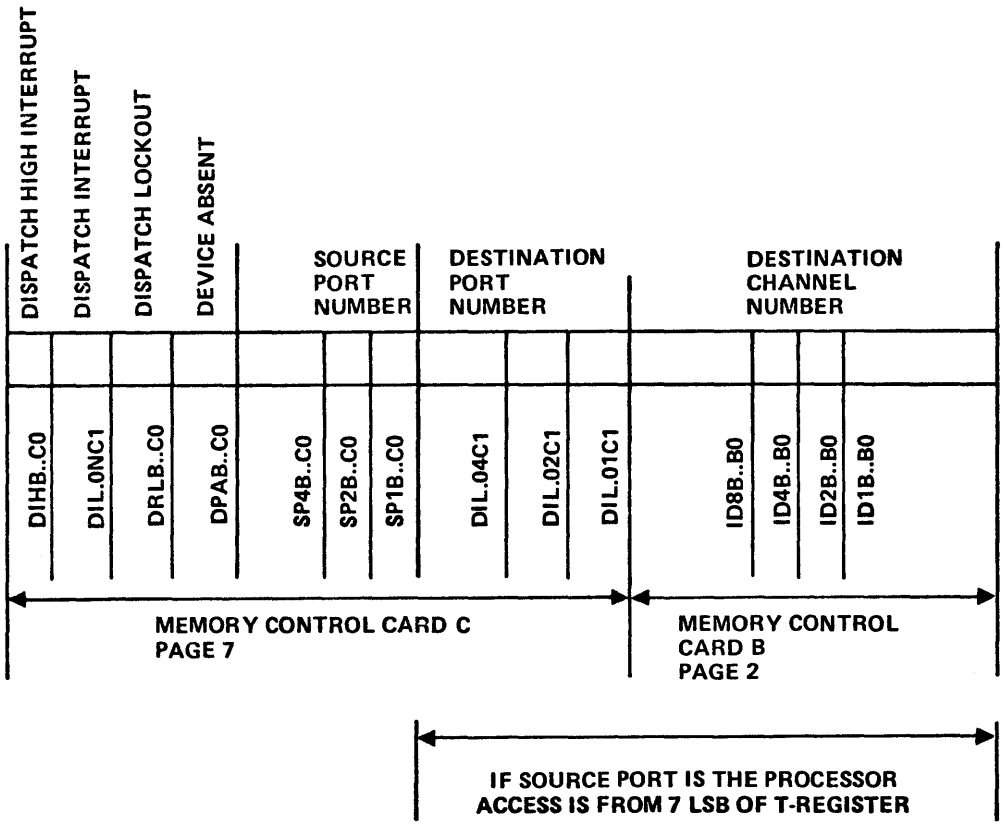


Figure 2-167. The Dispatch Register Structure

Dispatch Register Functional Detail

The dispatch register consists of one RFBN and two RFAN register chips, four FFAN flip-flops, and associated control logic (figure 2-168). It is used to store the control levels shown in table 2-14. The diversity in storage elements employed is due to the sectional nature of the register which is not addressable as a whole. Since the bits stored in the register come from a variety of sources, the derivation of each is described separately.

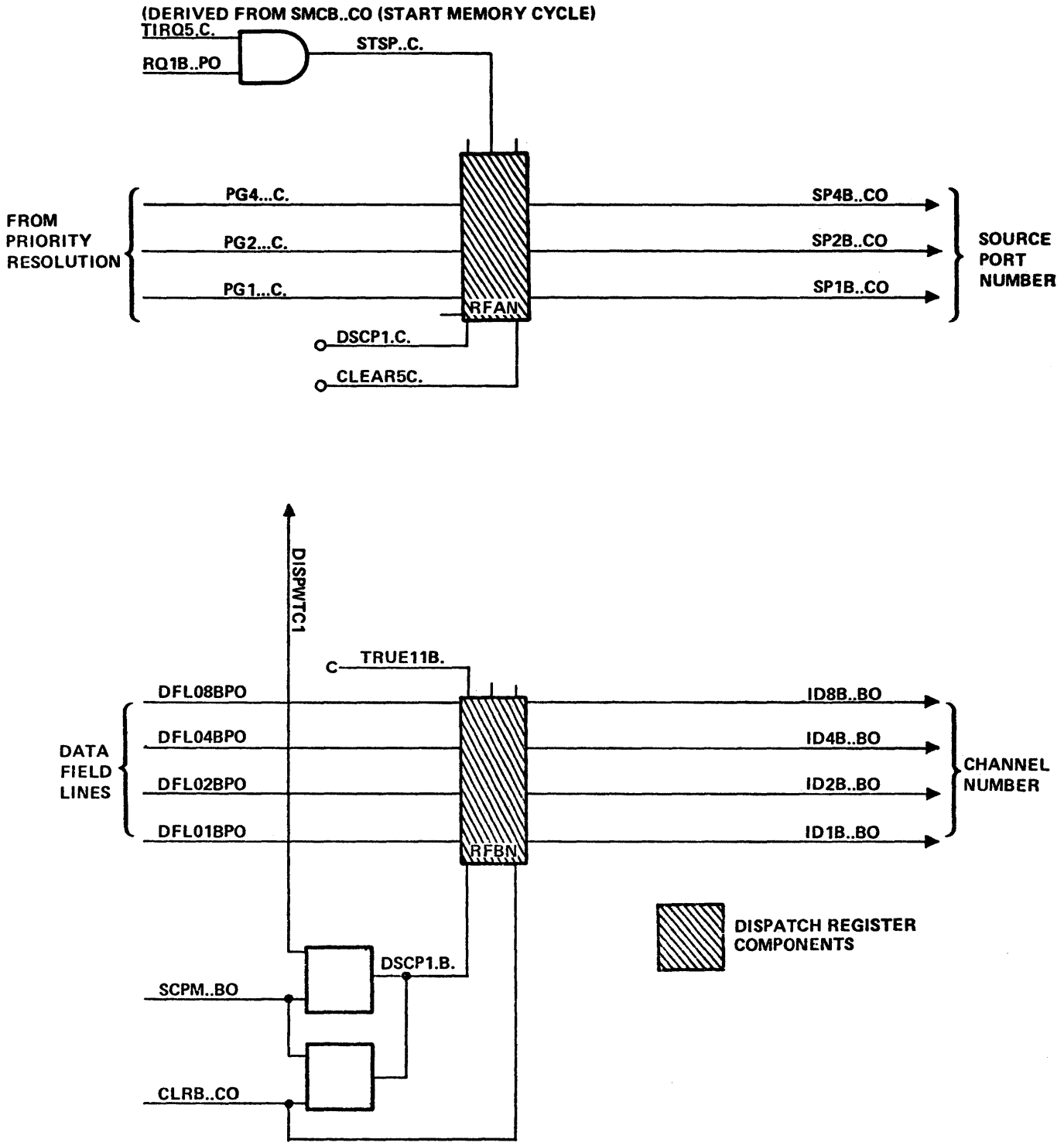


Figure 2-168. Dispatch Register (Sheet 1)

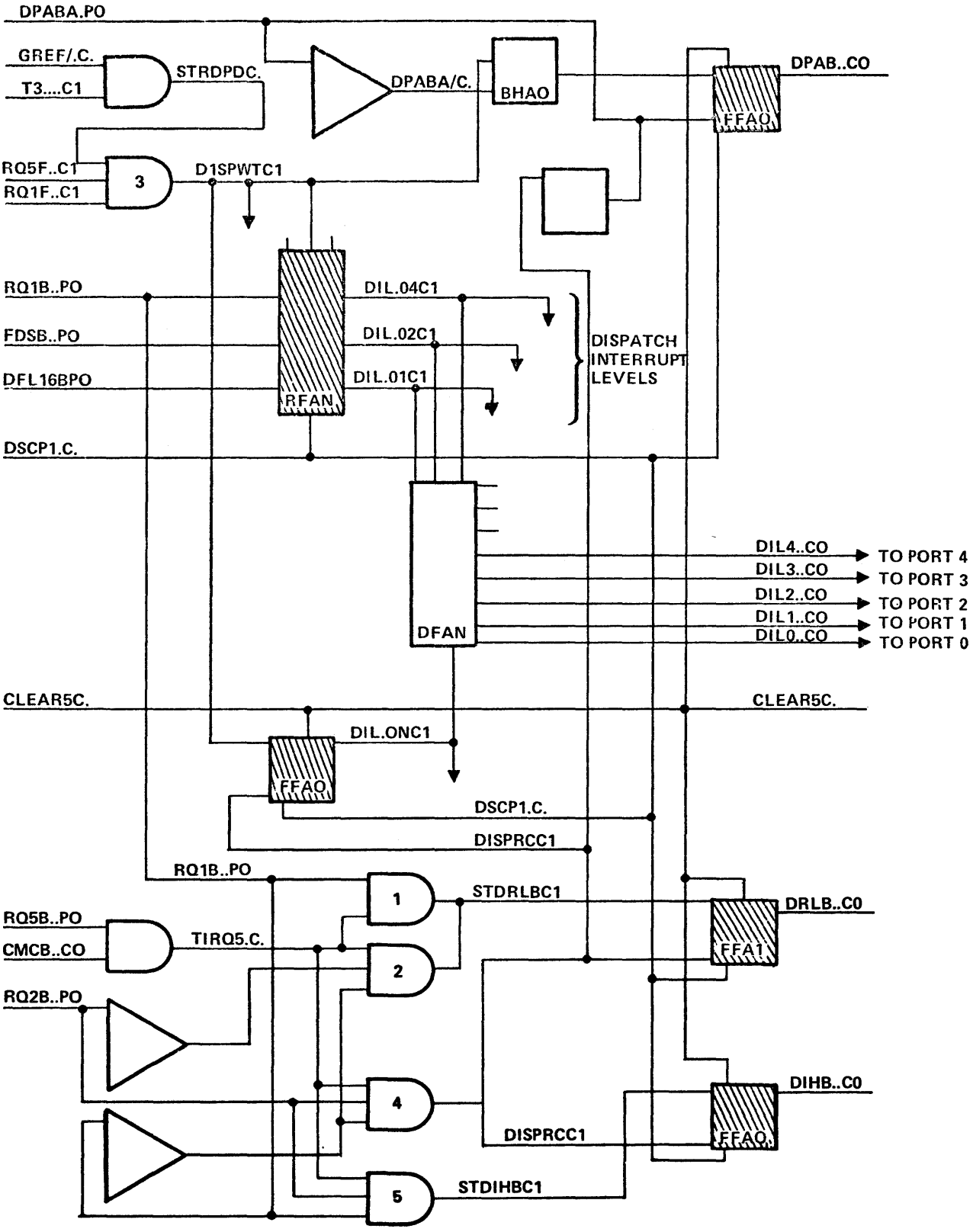


Figure 2-168. Dispatch Register (Sheet 2)

Table 2-14. Dispatch Register Contents

Mnemonic	Function	Storage Medium
ID1B..B0 ID2B..B0 ID4B..B0 ID8B..B0	Destination Channel Number	RFAN H9 in Fig. 2-169
SP1B..C0 SP2B..C0 SP4B..C0	Source Port Number	RFAN L9 in Fig. 2-169
DIL.01C1 DIL.02C1 DIL.04C1	Destination Port Number	RFAN J6 in Fig. 2-169
DPAB..C0	Destination Port Absent Level	FFA0 K7 in Fig. 2-169
DRLB..C0	Dispatch Register Locked Level	FFA1 K9 in Fig. 2-169
DIL.ONC1	Dispatch Interrupt	FFA1 K7 in Fig. 2-169
DIHB..C0	Dispatch High Priority Interrupt	FFA0 K9 in Fig. 2-169

DPAB..C0 (Destination Port Absent): This flip-flop is set when DISPWTC1 is true if the level DPABA.P0 from the destination port is false. When DPAB..C0 is set, this means that one port device has tried to send a message to another which is missing. The flip-flop may be reset by a read and clear operation (gate 4, figure 2-168) from the source port, or by another write operation from the same source port to another existing port. DPAB..C0 is sent to all ports.

DIL.ONC1 (Dispatch Interrupt Low): This flip-flop is set whenever DISPWTC1 is true (dispatch write, low or high). DIL.ONC1 enables DFAN H6 which decodes the destination port number. One of the levels, DILM..C0, is true, and this signals the appropriate port (each level is routed to a different port).

DIL.01C1, DIL.02C1, and DIL.04C1 (destination port number): These bits are set directly into the RFAN when DISPWTC1 is true, and originate in the sending port device. This occurs during the third clock (T3) of a dispatch write operation. Note that the input lines to the RFAN are shared with other logic, and contain different information during different phases of an operation. Their mnemonic identifiers (RQ1B..P0, FDSB..P0, and DFL16BP0) reflect the contents of these lines during the first clock period (T1) of the operation.

The contents of RFAN J6 may be changed by another write dispatch operation. The only protection against overwrite is the lockout bit, and it is up to each port device to refrain from writing if this bit is set prior to its request being acknowledged.

ID1B..B0, ID2B..B0, ID4B..B0 and ID8B..B0 (Channel Number): Like the port number, these bits are set directly into the RFAN when DISPWTC1 is true. The RFAN is always in the D-set mode, with DISPWTC1 enabling the clock. The input lines are shared, serving to transport the 4LSB of data field length at T1 time and the channel number at T3. The channel number itself usually represents the destination channel when written by the processor, and the source channel when written by another port device. Channel number 15 is reserved for use as an indicator that a memory parity error was detected during the fetch of an I/O descriptor.

SP1B..C0, SP2B..C0 and SP4B..C0. (source port number): These bits are set directly into the RFAN when STSP.C. is true, placing the register chip in the D-set mode. The bits constitute the binarily encoded source port number, and are derived directly from the EFAN in the priority resolution logic. STSP..C. comes true only when SMCB..C0 (start memory cycle) is true. This occurs during the first clock (T1) of the operation.

Dispatch Register Operation

Like other memory operations, a dispatch operation begins when SMCB..C0 (start memory cycle) is true. At this time (T1) the source port number (derived from the priority resolution logic) is loaded into the dispatch register. This occurs because RQ1L (indicating a write operation) and RQ5L (indicating a dispatch operation) are both true. The lockout bit (DRLB..C0) and the high interrupt bit (DIHB..C0) are also loaded at T1 time, if present.

At the third clock (T3) the remaining bits are loaded into the dispatch register. These are channel numbers (ID1B..B0 through ID3B..B0), destination port numbers (DIL.01C1 through DIL.04C1), destination port absent (DPAB..C0), and dispatch interrupt (DIL.ONC1). At this time, memory lines CP00 through CP23 contain the message to be written into memory at address zero. Dispatch write timing is shown in figure 2-169.

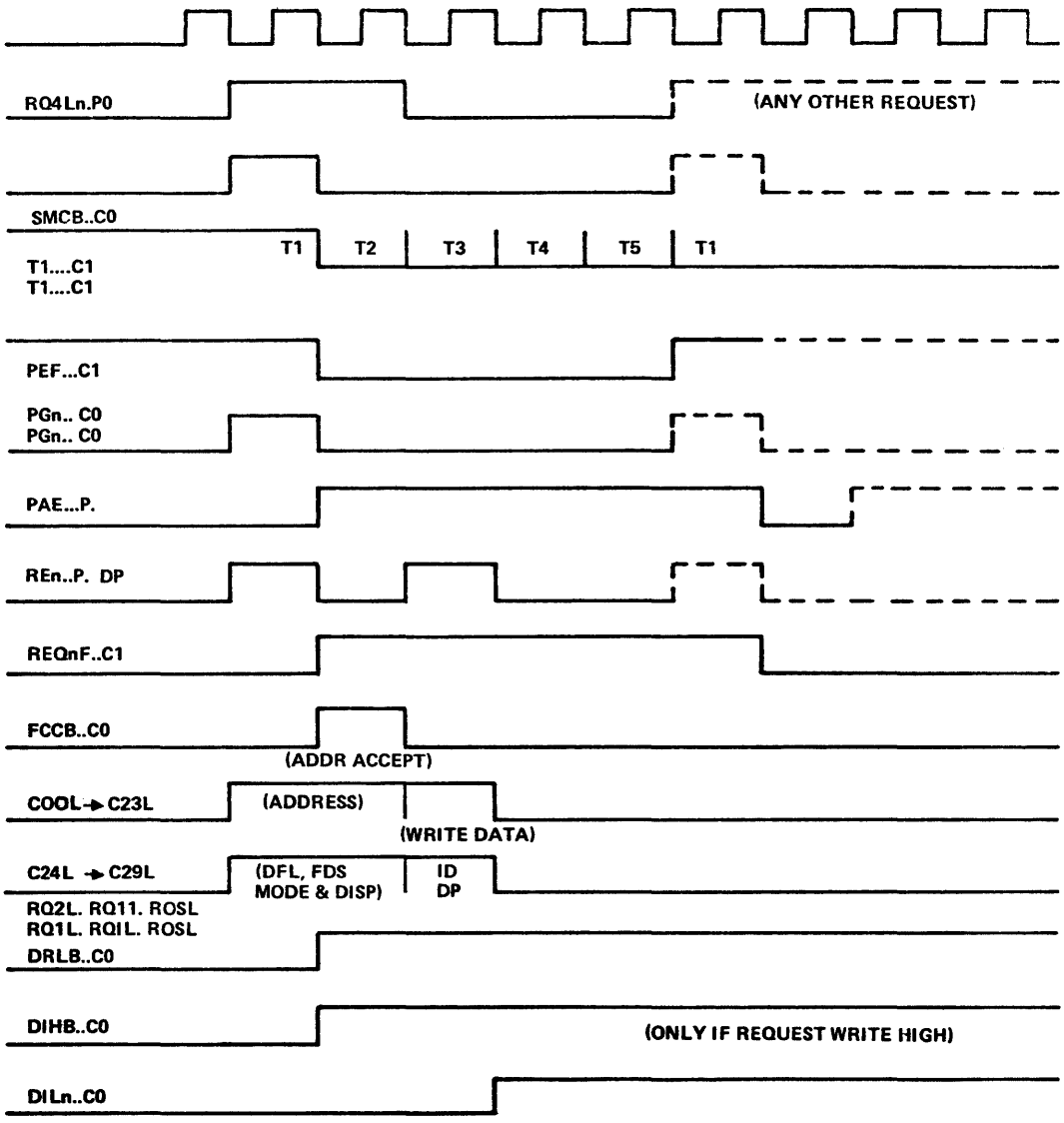


Figure 2-169. The Dispatch Write Timing Chart

PARITY AND OUT OF BOUNDS LOGIC

As a means of guarding against errors incurred in the process of storing and retrieving data from S-memory, parity generation and checking are provided. One parity bit is generated and stored along with each byte (8 bits) of data which is written into memory. This bit is retrieved along with the data byte when a read operation is performed, and is then used for checking purposes. The type of parity employed is odd; meaning that the sum of all true bits within the data byte plus the parity bit must be an odd number. Detected parity errors are sent through the port adapter to the port device for handling. In the case of the processor, a read parity error causes bit 3 of the CD register to be set. If a read parity error is detected during a micro fetch, the processor halts, and the console PARITY lamp is illuminated.

Although parity is generally reported as detected, provisions have been made for forcing good and bad parity under certain conditions. Good parity may be forced by selecting a field length of 25 bits (24 is the maximum number which may be read and gated to a port device in one operation). Likewise, bad parity may be forced with a selected field length of 26. Performing a read at an address which includes bytes beyond the physical limit of memory results in zero fills in the missing bytes, with good parity being forced for them. However, if all bytes addressed are outside the physical limit of memory, a parity error is generated.

Parity and Out-of-Bounds Logic Functional Detail

To accomplish the function required of it, the parity and out-of-bounds logic consists of several sections (figure 2-170): out-of-bounds detection, field length generation, parity check logic, and parity generation logic. These are integral portions of the port interchange and function as part of its operation.

Parity checking for a read operation consists of determining which portion of the 32 bits read from memory have been addressed, checking parity on those bits within the bounds of memory, and forwarding the results to the port adapter. Good parity is forced for those addressed bytes which are beyond the physical limit of memory. Note that such bytes always contain all zero bits. Inputs to the parity check logic include all 36 lines from RMIR (32 data plus 4 parity), 4 lines from the out-of-bounds logic, plus 2 from the field length generator.

Parity generation for writes involves determining which bytes of the four bytes read from memory have received new data, and generating an appropriate parity bit for them. Bytes not affected (if any) are re-written into memory with original parity unchanged. If any old data byte read during the read portion of the write cycle has bad parity, it is written back into memory with bad parity, and the parity error is reported. Inputs to the parity generator are the 32 data lines from the merger, and 4 parity lines from the parity check logic.

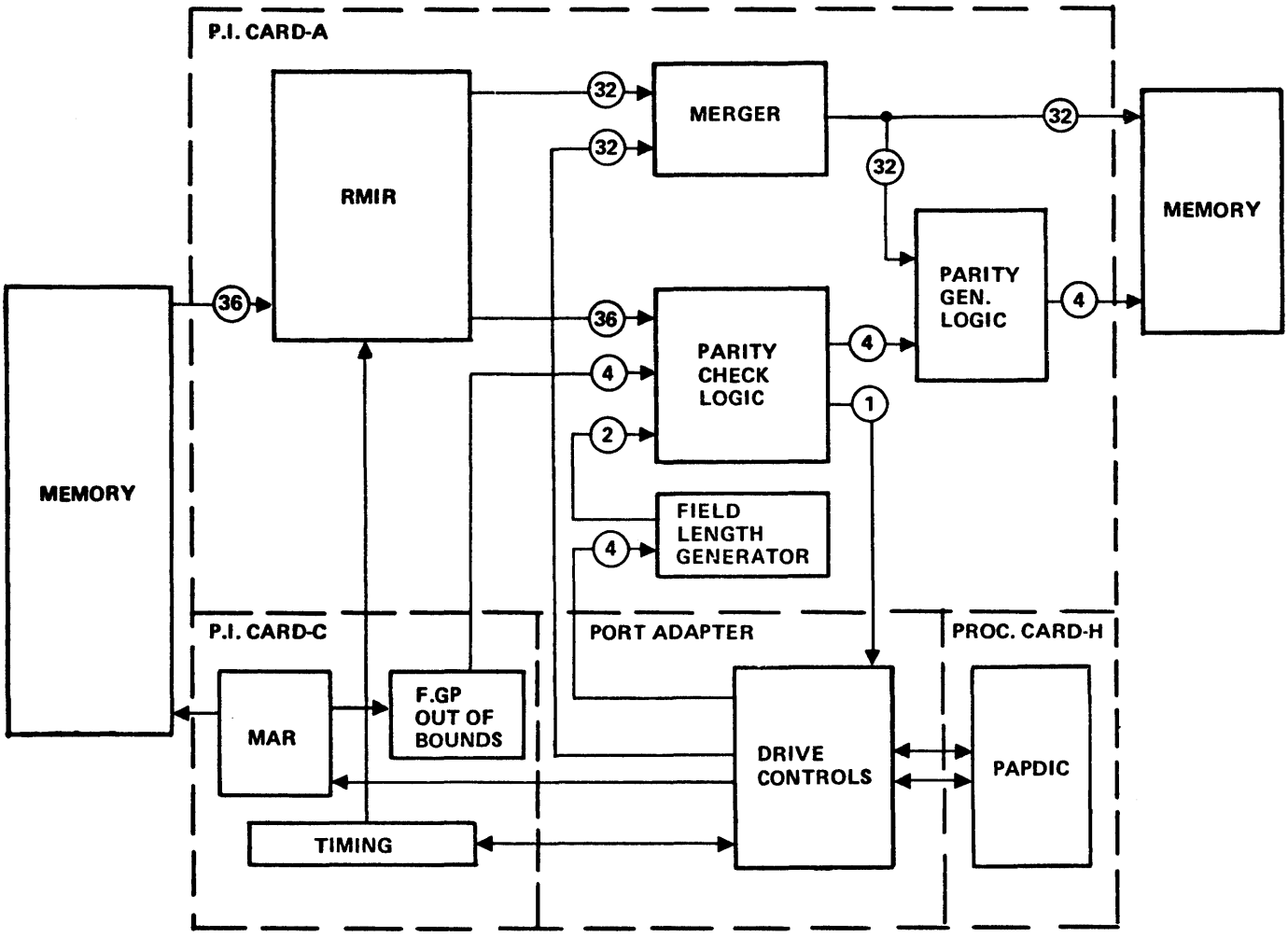


Figure 2-170. Memory Parity and Out-of-Bounds Logic Block Diagram (Showing Relationship to Other Sections of Port Interchange)

Parity Check Logic

When read data is present in RMIR, parity is checked for each byte by four parity check circuits. Refer to figure 2-171. Each of these circuits consists of four DFAN's which provide logic for decoding the eight data bits and the parity bit. The parity is checked by dividing the nine bits into three parts, and gating each three-bit portion through a DFAN. The DFAN outputs are connected in such a way that if none or any combination of two inputs are true, a true output is produced (indicating an even count of true bits). The outputs of DFANS 1, 2, and 3 are fed to DFAN 4, which is connected in the converse manner, such that a true output (PC.0.A1) is produced when any one or all three of its inputs are true (indicating an odd count of true inputs). The net result is that an even number of true input bits causes detection of bad parity, with an odd number producing a good parity indication.

The output of the circuit (PARITYA0) may be forced either true or false by additional means. Good parity may be forced by FGP00.C. or FL.25.A. being true. Either of these signals disables DFAN 4, rendering its output false. Bad parity is forced by FL.26.A., which forces PC.D.A1 true regardless of other conditions which may exist.

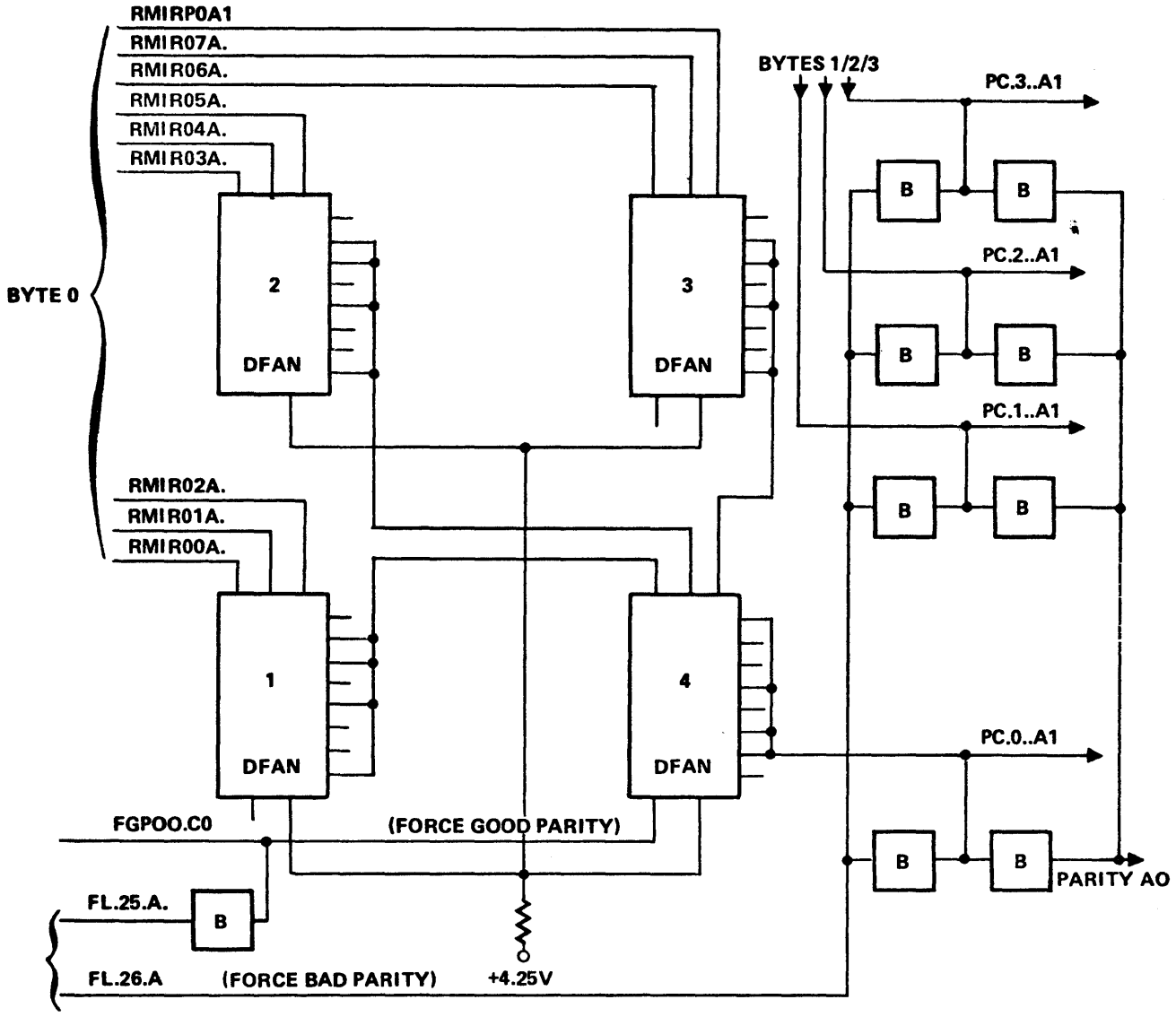


Figure 2-171. Main Memory Parity Check Logic

Special Data Field Length Logic

Selecting a memory cycle field length of 25 forces the generation of good parity during write cycles and prevents reporting of parity errors during read cycles. This is used primarily for establishing good parity throughout memory via load routines immediately after powering up the system. Selecting a field length of 26 forces generation of bad parity during write cycles, and indicates a parity error during any read cycle. Forced bad parity is used by software to verify proper functioning of the parity logic. The data field length as specified by the processor is detected by the logic shown in figure 2-172. Inputs are the C24L..H, C25L..H, C27L..H, and C28L..H lines, which contain the field length bits of 1, 2, 8, and 16 binary weight, respectively. This field length information is clocked into an RFBN when the start memory cycle signal (SMCB..C0) is true. The outputs of the RFBN are then decoded as FL.25.A. and FL.26.A., which are inputs to the parity check logic. The contents of the RFBN may only be changed by a new memory cycle, or cleared during a clear operation.

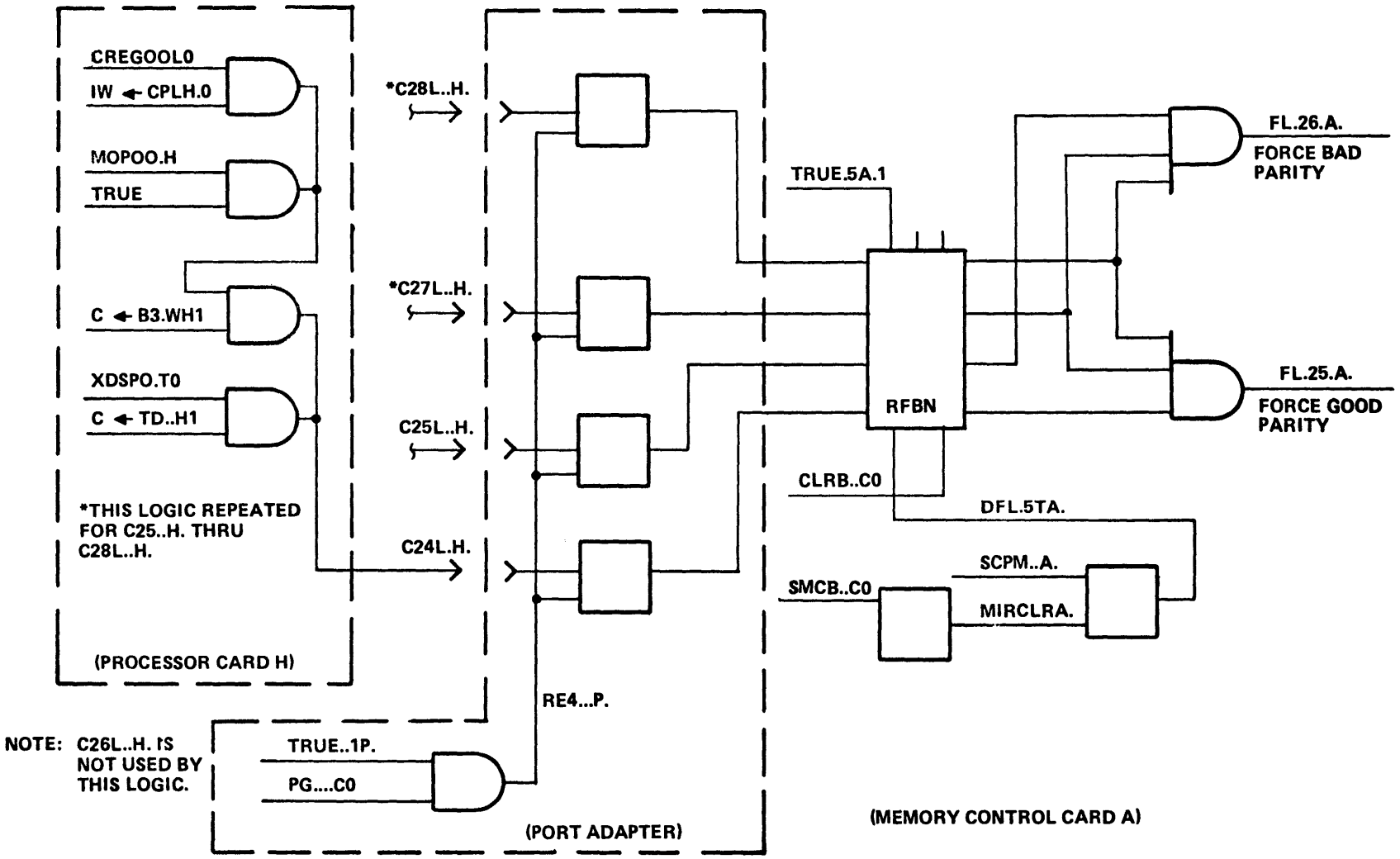


Figure 2-172. Special Data Field Length Logic

RMIR Parity Bit Storage

The parity bit of each of the four data bytes read from memory is stored in a separate RFBN, which operates continuously in the D-set mode (see figure 2-173). The parity bits are strobed into the RFBN's during the fourth clock (T4) of the memory cycle, which is the same as the RMIR register itself. This may occur at the clock pulse or be delayed by 40 nanoseconds (late strobe) depending on the operation of the corresponding portion of RMIR. The 4 RFBN's are cleared during the first clock (T1) of each memory cycle by SMCB.C0, or when CLRB...C0 is true. The latter signal occurs during the initial power up sequence, and when the CLEAR button on the processor console is pressed.

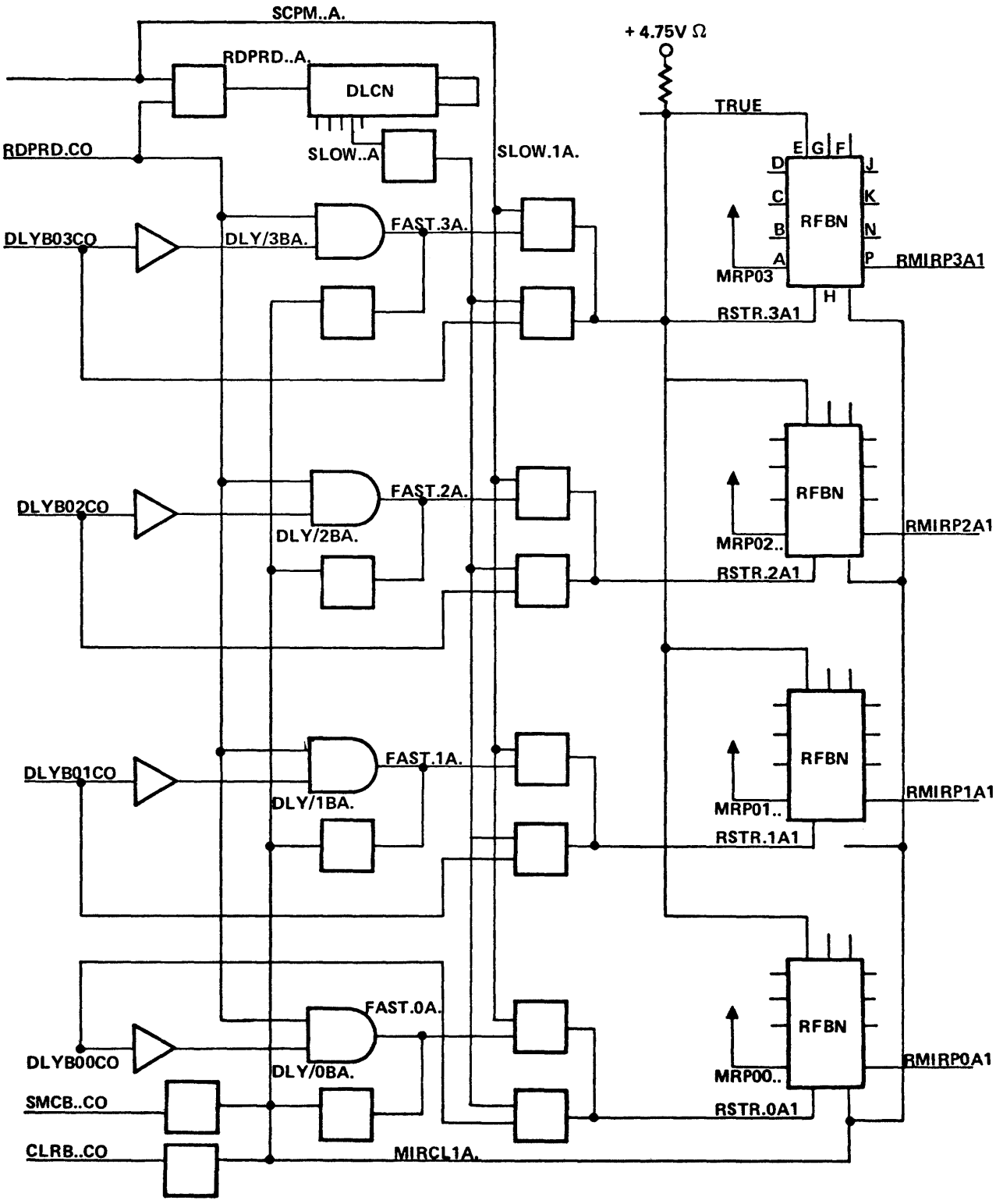


Figure 2-173. RMIR Parity Bit Storage

Parity Error Transfer Time

The parity error signal (PARITYA0) is transferred from the port interchange via the port adaptor on the REQUEST-2 bus to the processor. Refer to figure 2-174. It is gated onto the bus at T1-time after T4-time of a memory read cycle or at T1-time after T8-time of a memory swap cycle. It is controlled by the port adaptor enable signal PAE . . . P. and the output of a flip-flop set with T4-time and T8-time respectively. See figure 2-175.

Parity Generate Logic

The parity generate logic consists of four identical circuits, one for each of the four bytes to be written into memory. Refer to figure 2-176. The parity generation logic for each byte consists of four DFAN's, which form a decoding network to generate an odd parity bit. The inputs to DFAN's 1, 2, and 3 are taken from the MERGER, whose outputs are the 32 bits of new and old data to be written into memory. Like the parity check logic, the nine-bit byte is divided into three portions. Any even count of true bits requires a parity bit, and generates an output. The parity control level PC.0..A1 is normally false and good parity is written. This level is true if a field length of 26 is selected, or if a parity error was encountered for this byte on the read portion of the write cycle being performed. In both cases bad parity is written.

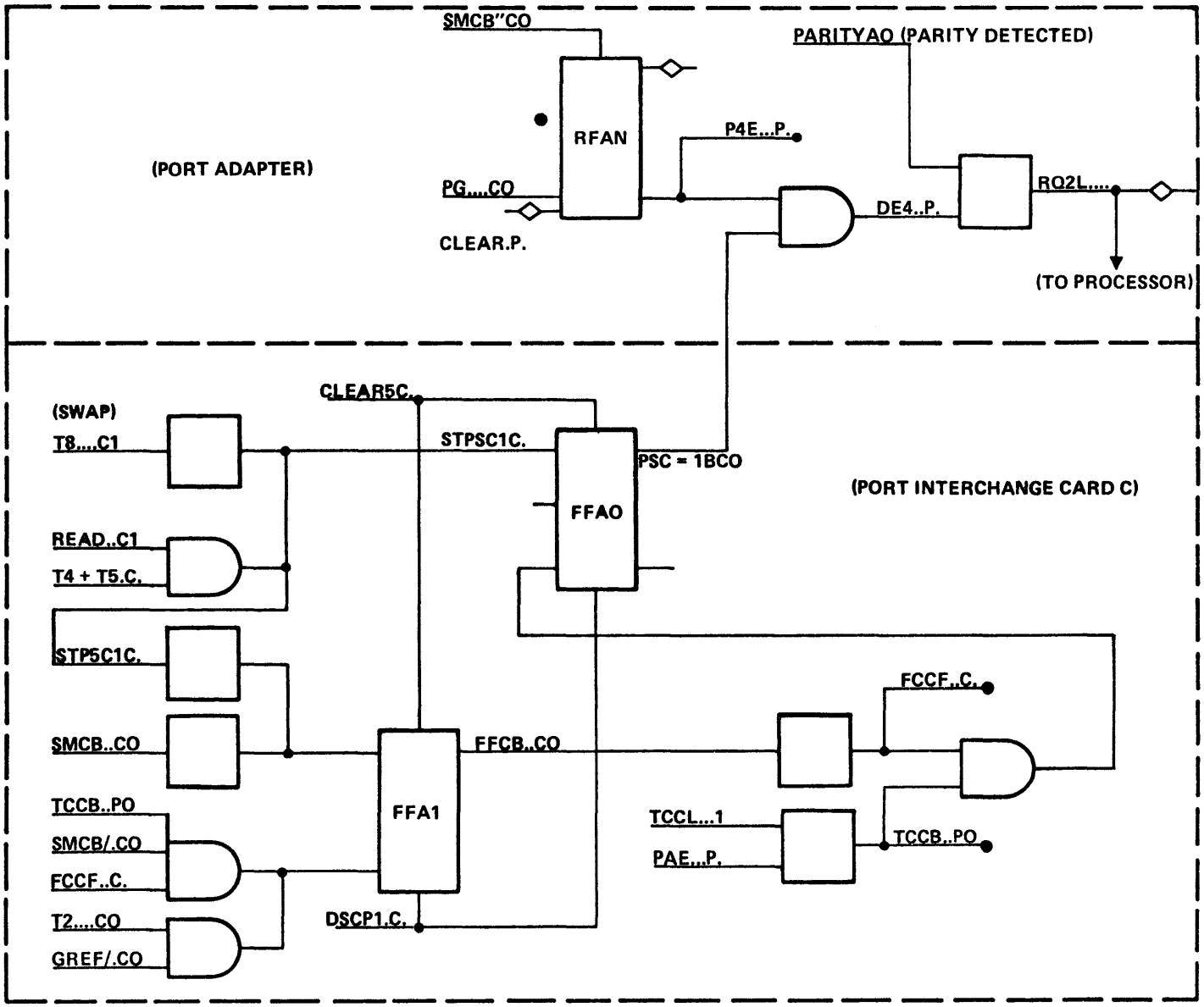


Figure 2-174. Parity Error Transfer Time Logic

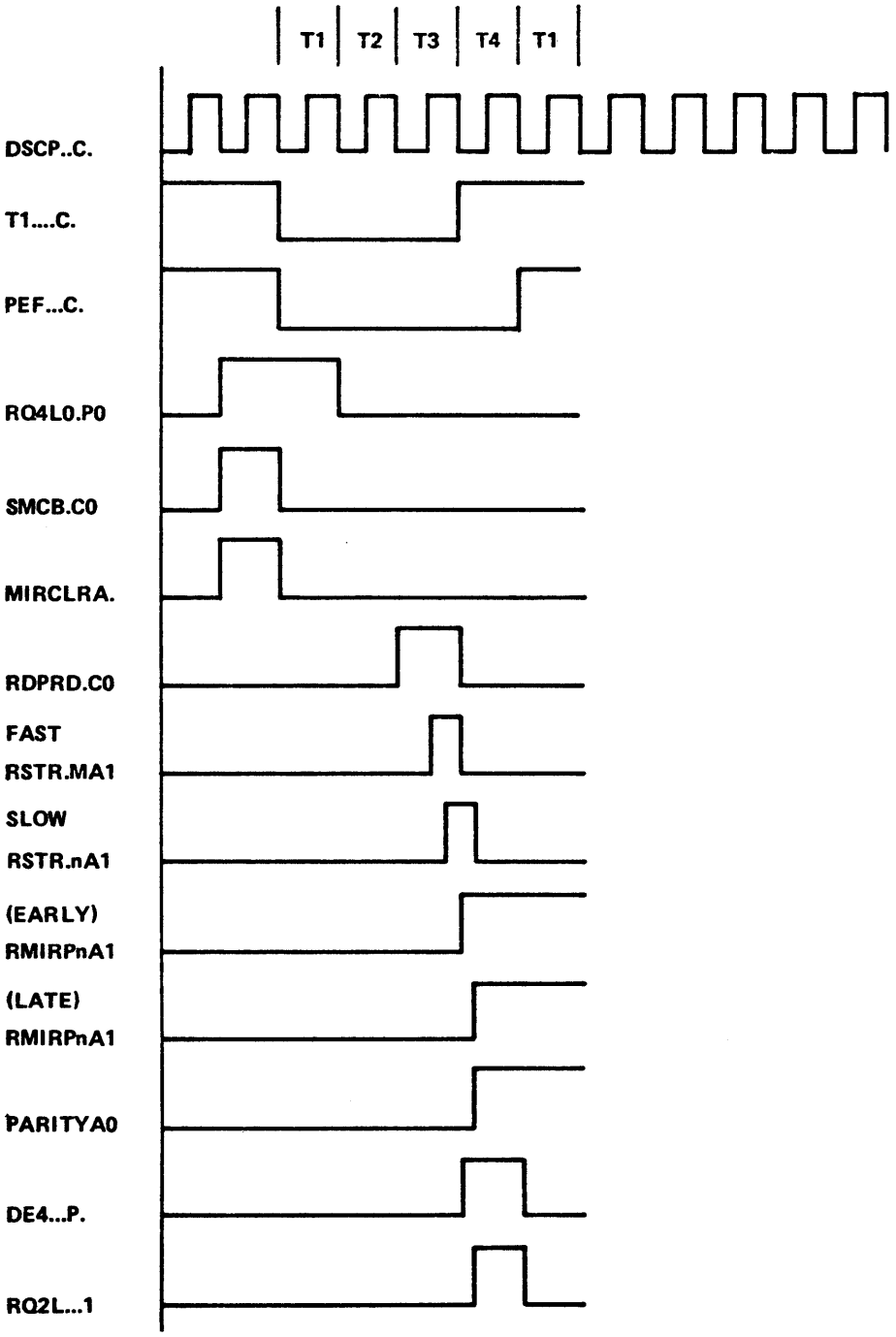


Figure 2-175. Parity Transfer Timing Chart

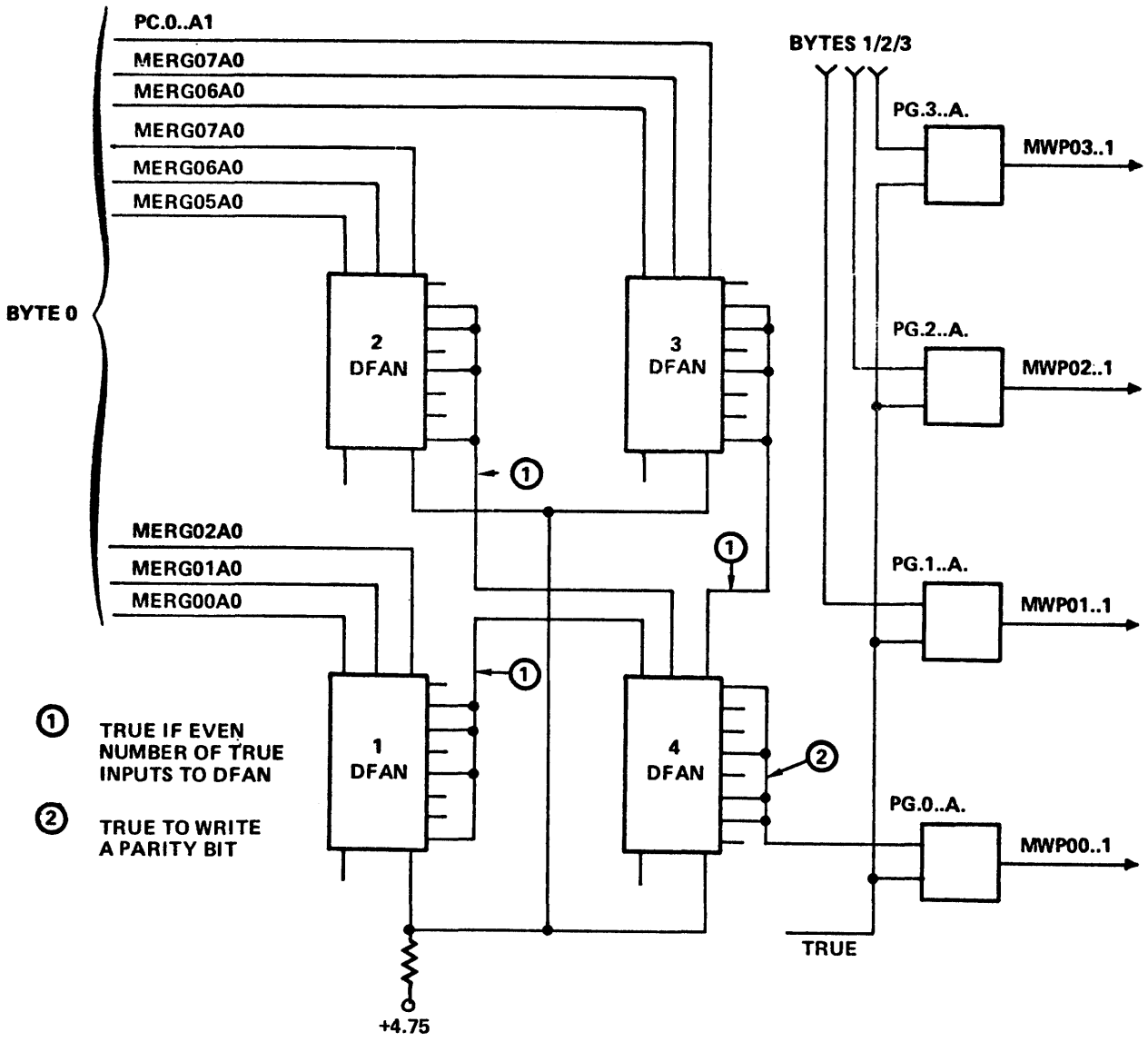


Figure 2-176. Main Memory Parity Generate Logic

Memory Out-of-Bounds and Force Good Parity Logic

Memory out-of-bounds detection is provided by a comparator circuit (figure 2-177) which monitors the input address and a preset jumper chip. The latter is hardwired to indicate the upper boundary of installed memory. The jumper configuration is compared with the memory address bits as follows:

Jumper	Memory Address Bit	Significance
R-S	15	Chip Row Select
G-H	16	
F-J	17	Card Group Select
E-K	18	
D-L	19	Memory Unit Select
C-M	20	

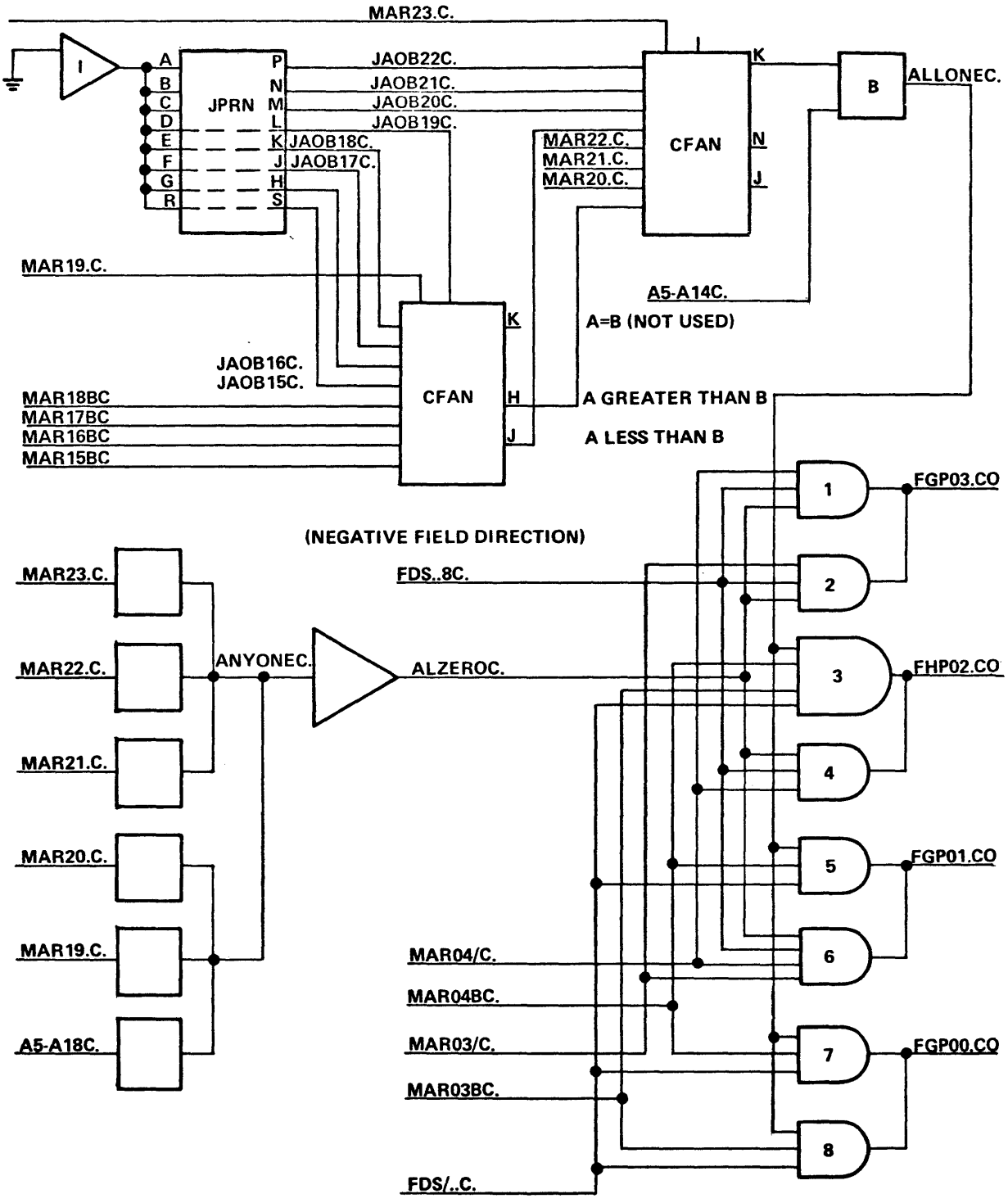
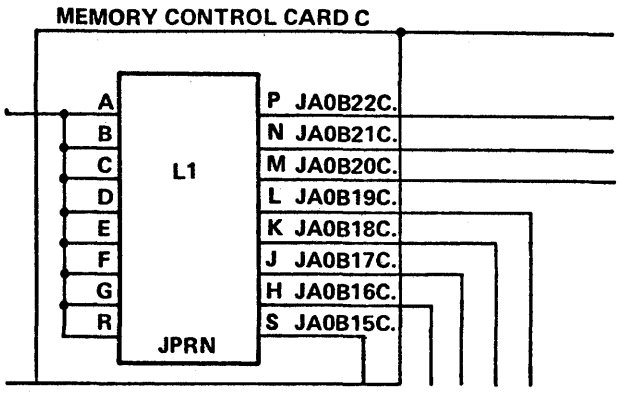


Figure 2-177. Memory Out-of-Bounds and Force Good Parity Logic

Figure 2-178 illustrates a jumper chip wired for a memory size of 128K bytes. The upper boundary of memory in this case would be the top of memory unit 01 (second unit of memory). If any of the last four bytes at the top of memory is addressed, MAR bits 05 through 19 are true.

FORCE GOOD PARITY

JUMPER	BYTES OF S-MEMORY	BITS OF S-MEMORY
A–P (L1)	512K	4096K
B–N (L1)	256K	2048K
C–M (L1)	128K	1024K
D–L (L1)	64K	512K
E–K (L1)	32K	256K
F–J (L1)	16K	128K
G–H (L1)	8K	64K
R–S (L1)	4K	32K
A–P (L4)		

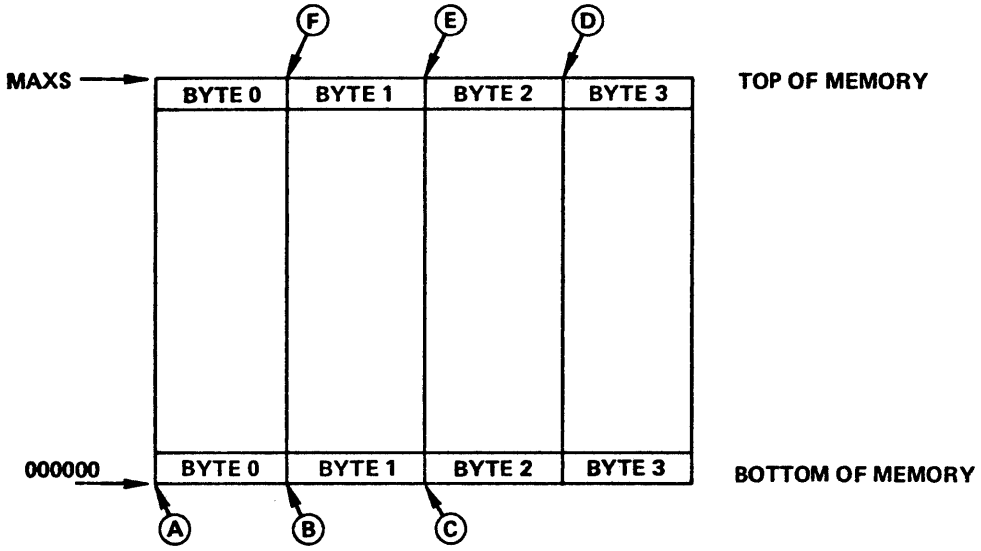


FORMULA:
WIRE JUMPERS CORRESPONDING TO SUM OF INSTALLED MEMORY MINUS 4K.

Figure 2-178. Force Good Parity Jumper Chip Wiring (64K Configuration Shown)

Both comparator chips produce a true output on pin K (A equal B). Likewise, buffer B is enabled, since all bits from MAR 05-MAR 14 are true. This combination generates the signal ALLONEC, which enables a set of gates to force good parity for the bytes addressed out of bounds. Note that only gates 3, 5, 7, and 8 (corresponding to bytes 0, 1 and 2) are enabled since in no event is good parity forced for byte 3. This is true because a valid address can overlap the upper boundary of memory by no more than 3 bytes. The force good parity gates may also be enabled when bytes are addressed out of bounds at the bottom of memory. In this case the output of the inverter (ALZEROC) in figure 2-177 is true, indicating that the bits in MAR05 through MAR23 are all zeroes. Note that ALLZEROC enables gates 1, 2, 4, and 6 corresponding to bytes 1, 2, and 3.

The bytes addressed out of bounds are determined by the status of MAR03, MAR04, and the field direction sign. With key byte addresses kept inside the physical boundaries of memory, there are six possible ways of addressing non-existent out of bounds bytes which, while retrieved as all zeroes, require a parity bit to prevent detection of an unwanted parity error. Figure 2-179 illustrates the six variants of addressing non-existent bytes from within the physical boundaries of memory. Note that if the MAR address itself points outside the physical bounds of memory, the signal ALLONEC does not come true, and a parity error is generated. Note that the jumper is strapped to give the same memory limit as MAXS in the processor.



TYPE	MAR 3	MAR 4	FIELD DIRECTION	FORCE GOOD PARITY FOR BYTES:	GATES USED
A	0	0	NEG	1/2/3	6/4/2
B	1	0	NEG	2/3	4/1
C	0	1	NEG	3	2
D	1	1	POS	0/1/2	8/5/3
E	0	1	POS	0/1	7/5
F	1	0	POS	0	8

Figure 2-179. Force Good Parity Variants

REFRESH LOGIC

Due to the type of memory chips used in S-memory, it is necessary to perform a "refresh" cycle on the whole of memory at least once every two milliseconds. Refresh is accomplished by doing a read operation, but not utilizing the data thus produced. Addressing for the read operations is provided by a refresh address counter (figure 2-180) which is incremented each time a refresh cycle occurs, thus advancing through memory. Refresh is a continuing process, into which the periodic memory access cycles which occur must be interleaved.

The basic refresh interval is divided into 128 separate read cycles of approximately 15 microseconds each. Memory addressing for refresh is a composite of sequential and simultaneous actions. Due to the characteristics of the memory chips, refresh for an entire chip may be accomplished by addressing only the first 32 bit locations. In practice, the refresh address is fed simultaneously to all chips on a given card group. The address counter is advanced after each refresh read, proceeding from 0 through 31. This process is continued by repeating it sequentially for each card group (maximum 4) within the memory unit, for a total of 128 refresh reads occurring within the 2-millisecond time limit (128 x 15 microseconds = 1.920 milliseconds). The basic 128 read cycle refresh process occurs simultaneously in each unit of installed memory, thus encompassing the whole of memory.

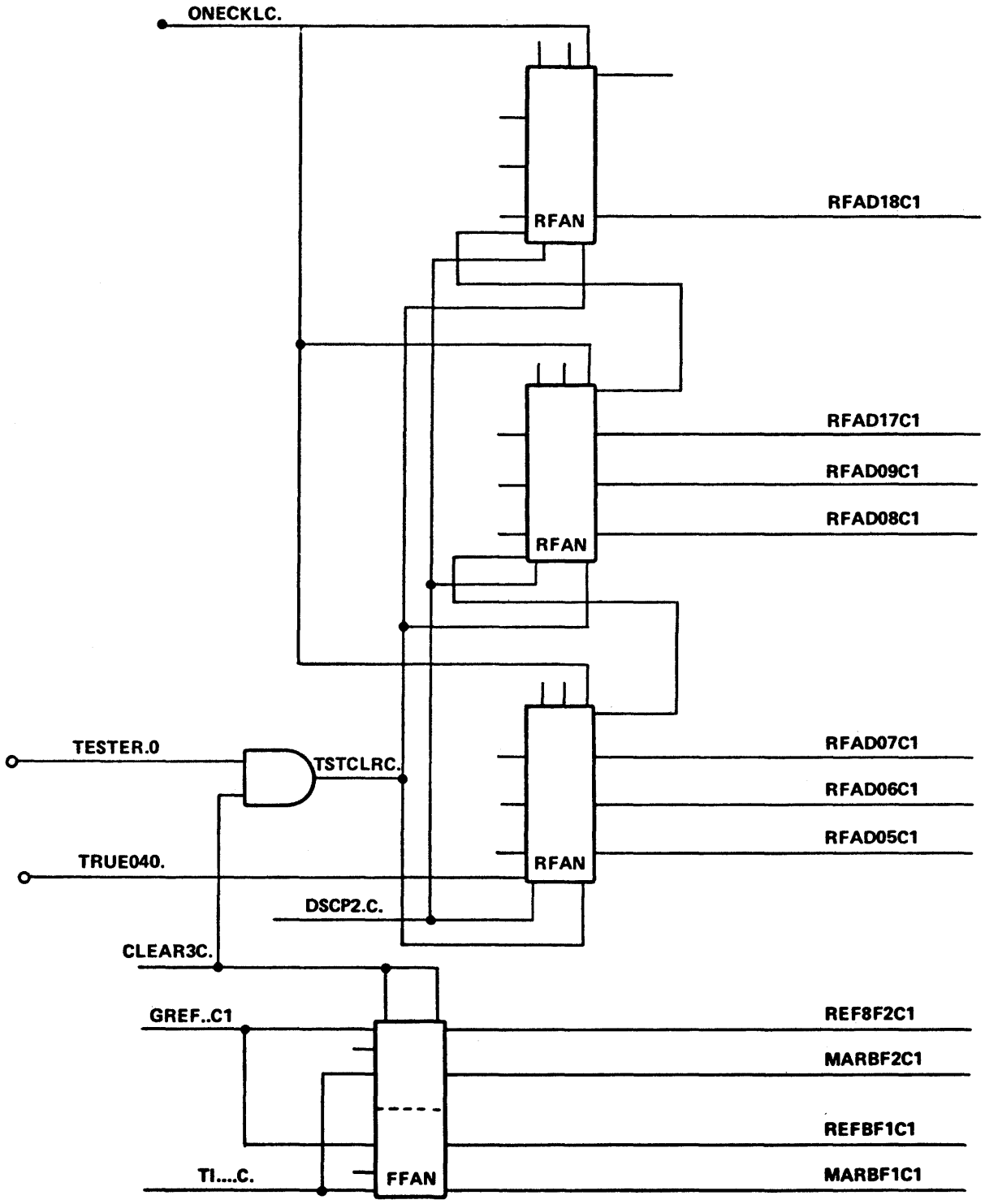


Figure 2-180. The Refresh Address Logic

Timing for memory refresh is provided by a refresh signal generator (figure 2-181) which operates in two stages. Essentially, the circuit is arranged to produce the signal GREF..C1 once within each 15-microsecond time period. The first 10 microseconds of the period serve as a buffer during which a refresh occurs only if a normal memory cycle is not taking place (ANYPRQC. false). If a refresh does not occur during the 10 microseconds, generation of GREF..C1 is forced, which terminates the action in progress by resetting the memory cycle timing counter. The refresh then occurs, after which normal memory operation resumes.

Refresh Logic Functional Detail

A 5-nanosecond pulse (RFMVIC). occurring at 15-microsecond intervals is generated by two TAON chips. When this level is true, flip-flop 1 is set at the next system clock pulse, generating HRF.FFC. Flip-flop 2 follows one clock later, generating HRF1FFC1. Four microseconds thereafter RFMV1.C goes false, and on the next system clock pulse flip-flop 1 is reset. At this time, CRF.FFC1 and HRF1FFC1 are both true, enabling gate 3, and producing a single pulse called ONECKLC. ONECKLC sets flip-flop 4 (which operates in the JK mode), generating DRF...C1 (demand refresh). DRF...C1 stays true until GREF..C1 (generate refresh) appears.

The level LRF.FFC1 stays true for 10 microseconds, until the appearance of the next RFMV1C. pulse. If during this period the level ANYPRQC (any memory request) goes false, the gate labeled low priority in figure 2-181 is enabled and the level GREF..C1 goes true. GREF..C1 resets flip-flop 4 at the next system clock, and is latched into flip-flop 5.

With GREF..C1 true, a read memory cycle is started. The memory cycle timing counter proceeds to T4, at which time it is cleared, flip-flop 5 (in figure 2-181) is reset, and the output of the refresh address counter is sent to memory. Since the memory cycle is terminated at this point, no action is taken regarding the read data produced thereby. Note that the refresh address counter is upcounted by one at each ONECKLC. pulse.

If the level ANYPRQC. stays true throughout the 10 microsecond "grace" period, the low priority gate is never enabled. DRF...C1 (demand refresh) stays true for the entire period awaiting a refresh cycle. At the next RFMV1C. pulse (15 microseconds following the previous one) HRF.FFC. is again set. With DRF...C1 and HRF.FFC both true, the high priority gate is enabled at the end of the currently-executing memory cycle (when the memory cycle timing counter is reset, producing T1...C1), forcing GREF..C1 true. Refresh timing is shown in figure 2-182.

S-MEMORY

S-memory consists of a network of 1 x 1024 bit RAM memory chips, plus addressing and interfacing logic. The storage elements of memory are arranged to provide a 36-bit width (32 data bits and 4 parity bits), with the length expandable to any amount desired (within practical limits). In hardware terms, this memory configuration involves rows of 36 RAM chips which are divided between two standard size logic cards. Each pair of cards (card group) contains four such chip rows for a total storage capacity of 16,384 data bytes (4 x 1024 bytes per row x 4 rows). The card groups are further combined to form memory units which consist of four (maximum) card groups. Each memory unit is provided with its own field address and interface control logic. The entire memory subsystem may comprise up to four such memory units, which are connected in "daisy-chain" fashion, with each unit being uniquely addressed, and all being simultaneously accessed.

To allow the size of memory to be set to any desired amount, several increments are provided. The memory units need not be full, but may contain from one to four card groups. In addition, card groups may be composed of half (two chip rows) populated storage cards. This provides memory size changes in 8192(8K) byte increments. In all cases, the smallest unit of storage elements must serve as the top of memory. For example, if an 81,920 (80K) byte memory is desired, it consists of one full memory unit (64 K bytes) of four fully-populated card groups, and one additional memory unit containing one fully-populated card group (16K bytes). If it is desired to expand this memory to 88K (90,112 bytes), one half-populated card group is added to the second unit. Only a single half-populated card group may be used in a memory subsystem.

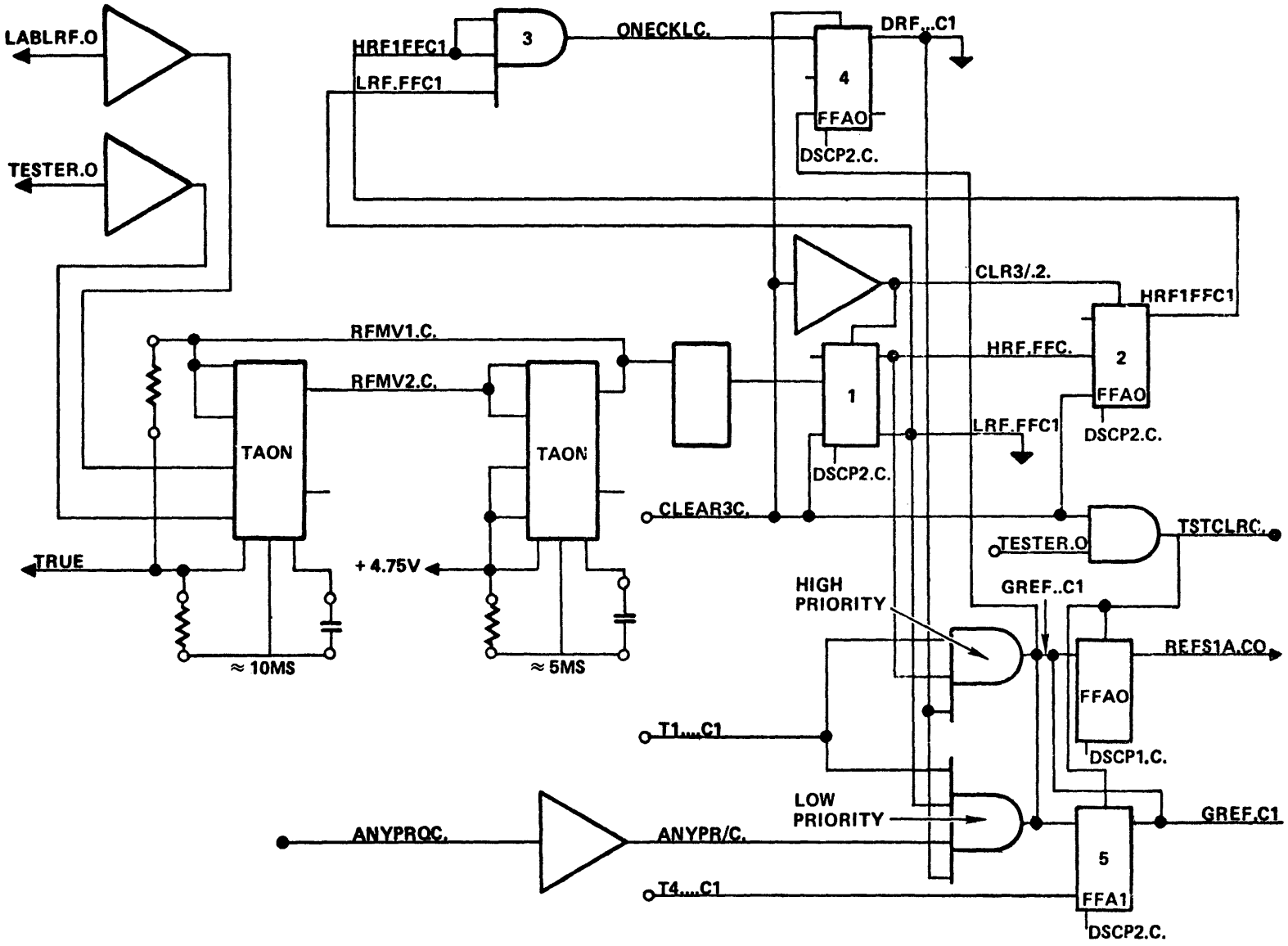


Figure 2-181. Refresh Signal Generation

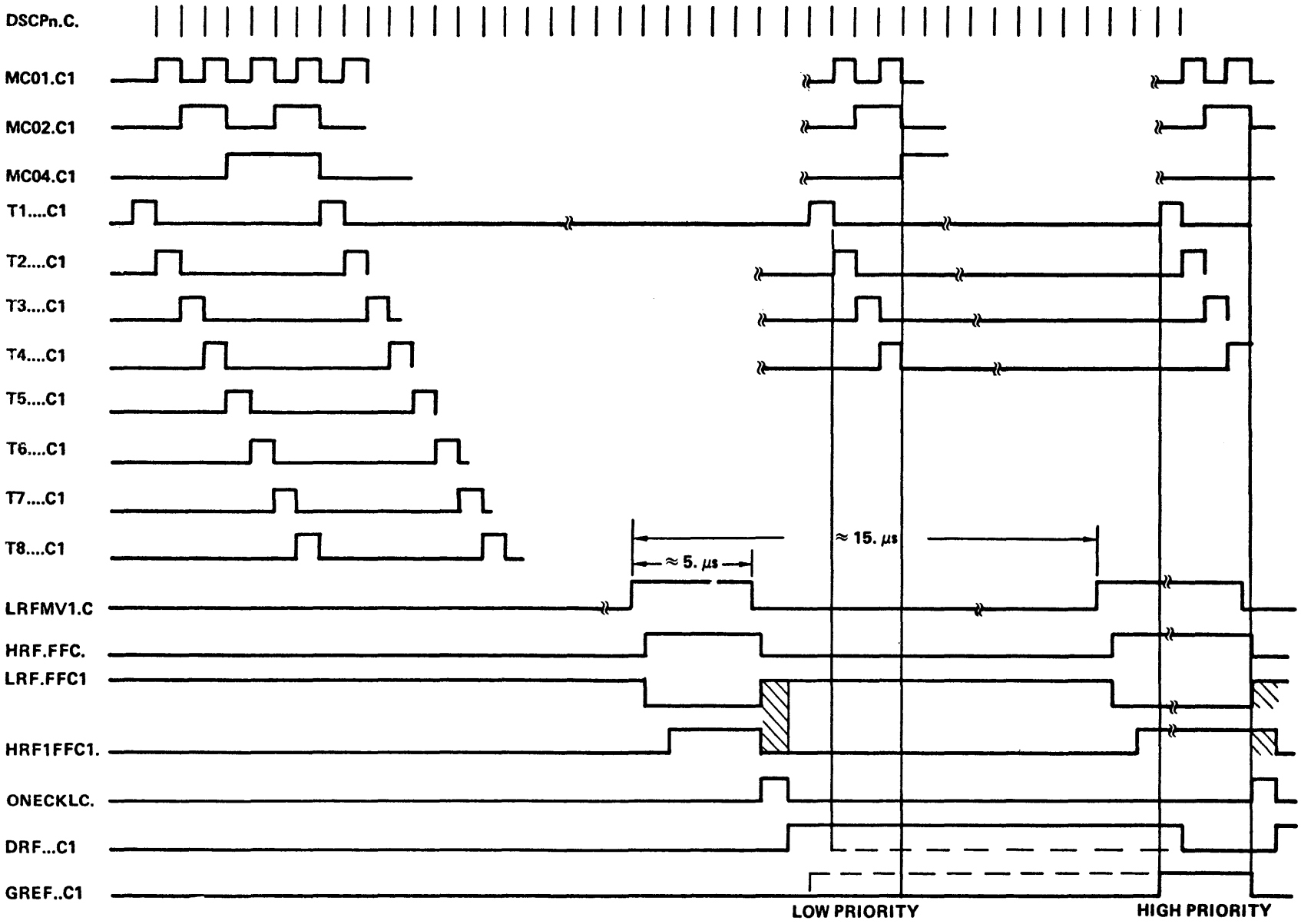


Figure 2-182. Memory Cycle and Refresh Timing Chart

S-Memory is bit-addressable, meaning that a selected field of data (of up to 24 bits in length) may be read or written beginning at any desired bit location within memory. This capability is provided by a combination of byte boundary addressing and data rotation, with the latter function being performed externally in the port interchange. Since the data field length of a read or write operation may involve up to three bytes (24 bits) of data, it is necessary to access four bytes of storage to allow the bit addressing (via data rotation) process to take place. Addressing within memory itself involves determining (from the address supplied) which four-byte storage locations are to be accessed, and producing the appropriate control signals for doing so. Because the sequential bytes are, more often than not, located in different words of memory, modifying the basic address supplied is a necessary part of this function.

MEMORY ADDRESSING

On each memory operation the port device, by the way of the port interchange, sends a 24-bit binary address which is received by the field address card in each S-memory unit. In actuality, only the 21 most-significant bits of the address reach memory itself, the 3 least-significant bits being used exclusively by the rotator in the port interchange.

The 21 address bits received identify a key byte in memory. This is the unique memory byte within which the bit location specified by the full 24-bit address is contained. Also sent along with the address is a field direction sign bit which determines whether the three higher-addressed bytes (forward) or three lower-addressed bytes (backward) are to be accessed in addition to the key byte. All memory cycles involve the accessing of four bytes of data. Isolation of a smaller field (of up to 24 bits) within these four bytes for reading or writing is accomplished in the port interchange.

Address modification is controlled by the contents of MA03, MA04 (memory address lines), and the field direction sign. When reading/writing in the forward direction, it may be necessary to provide byte 0, bytes 0 and 1, or bytes 0, 1, and 2 with incremented addresses, depending on the location of the key byte. Likewise, when reading/writing in the reverse direction, byte 3, bytes 3 and 2, or bytes 3, 2, and 1 may require a decremented address. This is shown in figure 2-183. Note that byte 3 never requires an incremented address, and byte 0 never a decremented address. For address modification purposes the word, row, and group portions of the memory address are fed simultaneously to four "channels" which correspond to the four byte columns in memory layout. These channels, in which the address may be individually incremented or decremented, drive the address line inputs of the appropriate portions of memory.

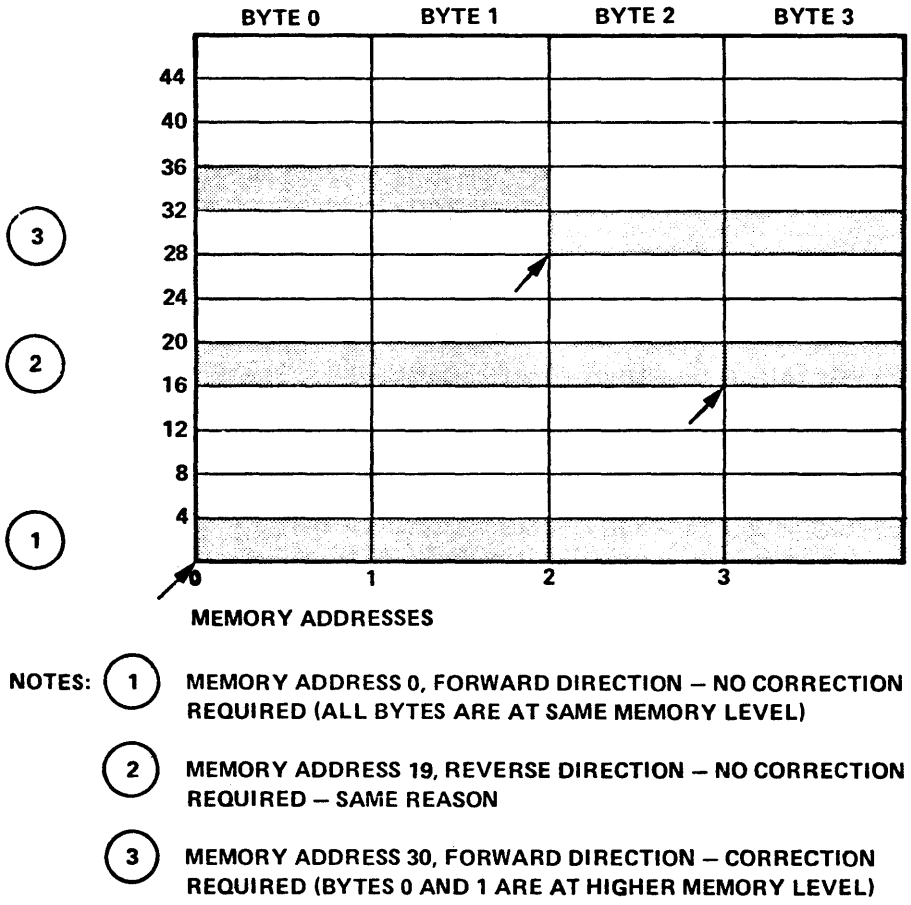


Figure 2-183. Address Modification

S-MEMORY STORAGE CARD

The standard memory storage card is illustrated in figure 2-184. The memory chips on it are divided into two groups (X and Y) which correspond to the byte columns in memory layout. Since a card group is made up of two such cards, chip group X may represent byte 0 or 2, and Y byte 1 or 3, depending upon the card's position within the group. Note that there are nine chips in each half row, representing eight data bits and one parity bit. The balance of the card consists of address buffers, inverter drivers, and sense amplifiers. A functional block diagrams of the card is shown in figure 2-185.

For addressing purposes, each such storage card receives two sets of address lines which contain address bits 05 through 18. Each address input furnishes the addressing information for one of the two chip groups, this separation being necessary because independent byte address control is a required feature of the addressing scheme employed. Address bits 05 through 14 are inverted to produce CA0X through CA9X, which go to all 36 chips within a group to select one of 1024 bit locations (actually words of memory) in each. Only one chip row is enabled at any given time, this being controlled by address lines A15 and A16. Lines A15 and A16 also enable the signals precharge, chip enable, and write enable.

Input (write) data to the card is passed through TTL inverter-drivers which provide the proper input for the TTL storage chips employed. Output (read) data is amplified by way of SBMN sense amplifiers, which convert it into CTuL logic levels.

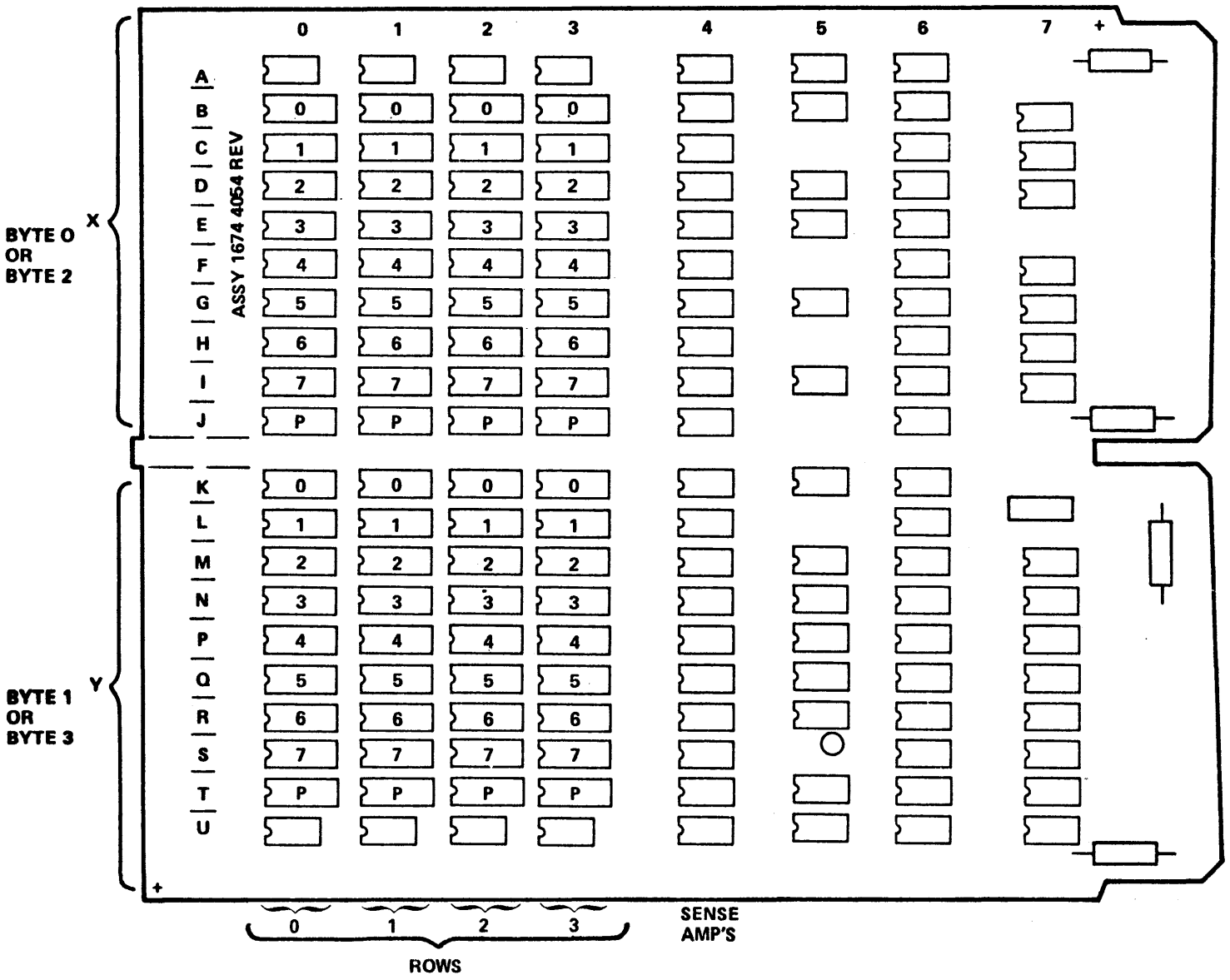


Figure 2-184. S-Memory Storage Card (Level B)

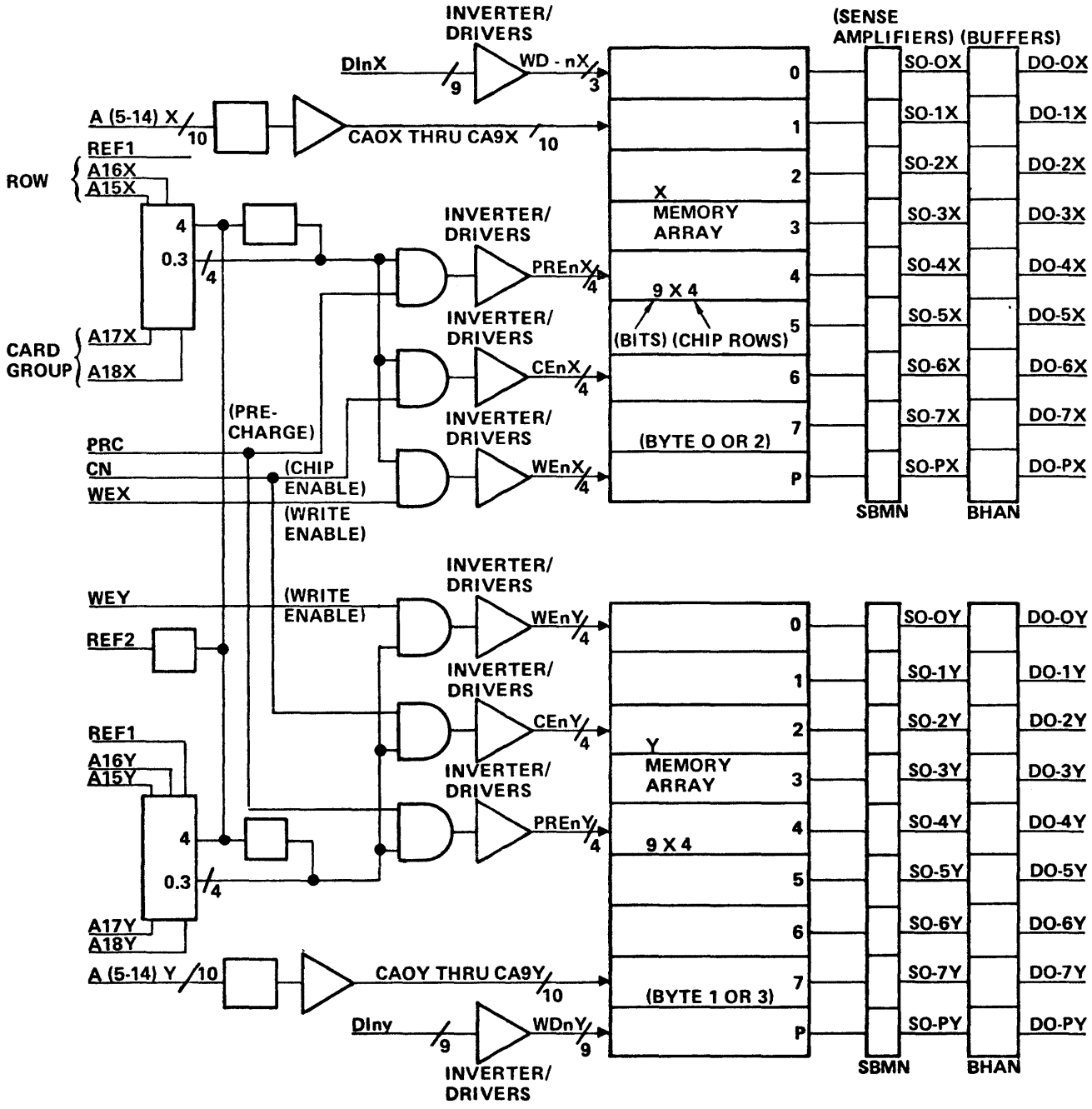


Figure 2-185. Storage Card Functional Block Diagram

S-MEMORY UNIT

Individual storage cards are assembled (in groups of two) into memory units, which are, as the name implies, self-contained units of electronic memory. Each memory unit may contain from one to four card groups, and may, therefore, represent from 8192 (8K) to 65,536 (64K) bytes of data storage. In the minimum configuration, such a memory unit would contain one group of half-populated storage cards, whereas the maximum would be four groups of fully-populated cards. In all cases, each memory unit occupies a 10-card backplane, and is powered by its own memory power supply. Each unit also includes a field address (FA) card and an interface control (IC) card which contain the logic necessary to interface the storage cards in the unit with the rest of the system. Physically, the memory unit is assembled as shown in figure 2-186. The FA and IC cards always occupy the two center slots, with the storage cards distributed on either side. The storage cards are located as shown to equalize propagation times.

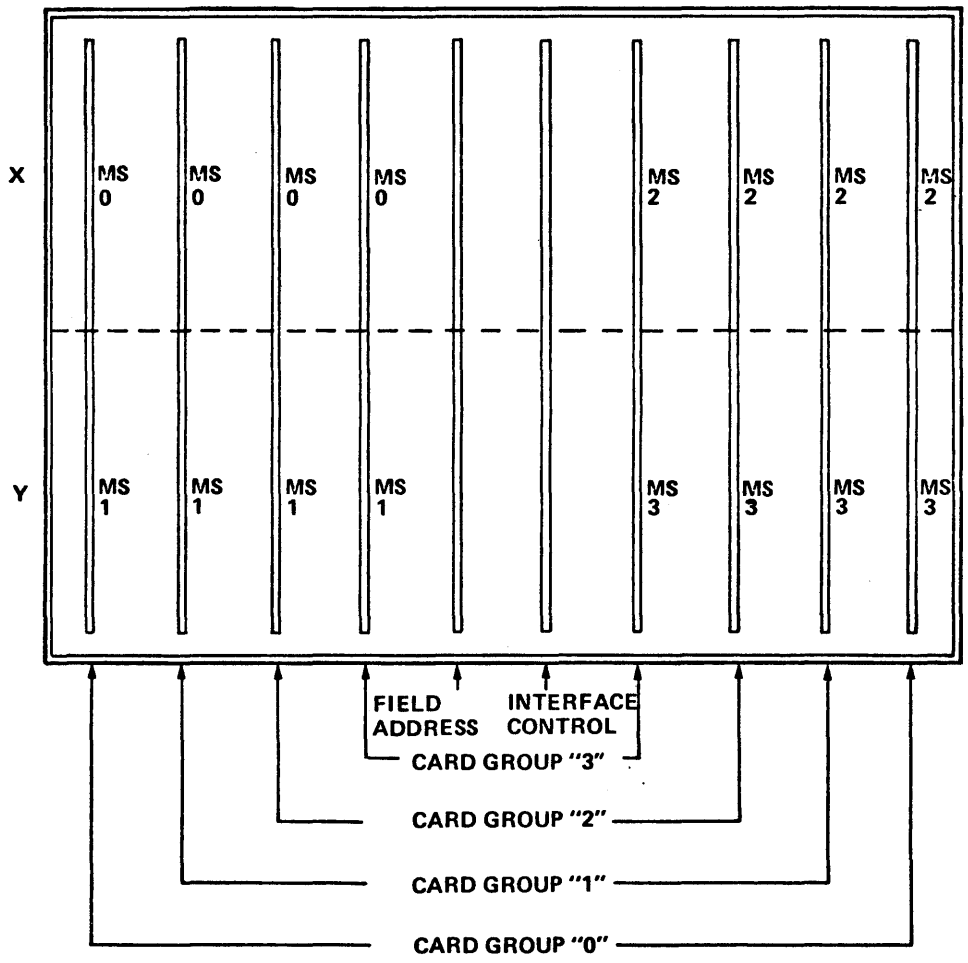


Figure 2-186. The 64K Byte S-Memory Unit

FIELD ADDRESS LOGIC

The field address logic performs the enabling and address control functions for the storage elements within its own memory unit. Essentially, the logic is equipped to determine whether or not the memory address furnished is located within its portion of memory and, if so, to produce the proper enabling signals for gaining access thereto. To perform its function, the field address logic consists of elements for byte decoding, address gating and correction, and unit enabling.

Since each memory cycle involves access to four sequential bytes which may or may not be located at the same word level in memory, address correction is an important part of the field address logic's function. Implementing such a system involves separating the address information furnished to each of the byte columns (chip groups within the memory unit) into individual channels which may be independently incremented or decremented. Control of this process is dependent upon the location of the key byte, and the field direction sign. Also included in the field address logic is a unit enabling circuit which has provisions for activation concurrent with a higher or lower memory unit when a data field spanning unit boundaries is encountered. A functional block diagram of the field address logic is shown in figure 2-187.

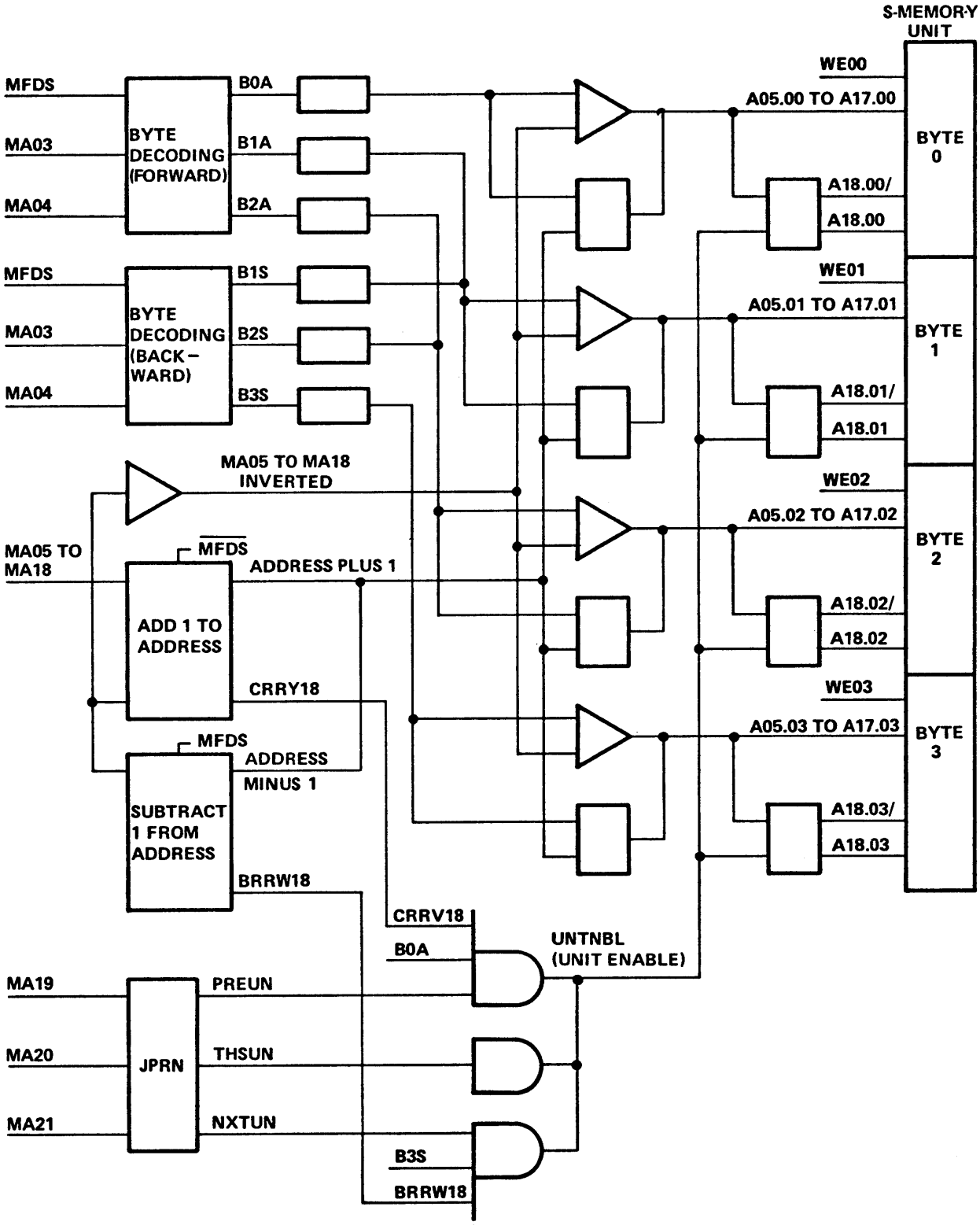


Figure 2-187. Field Address Logic

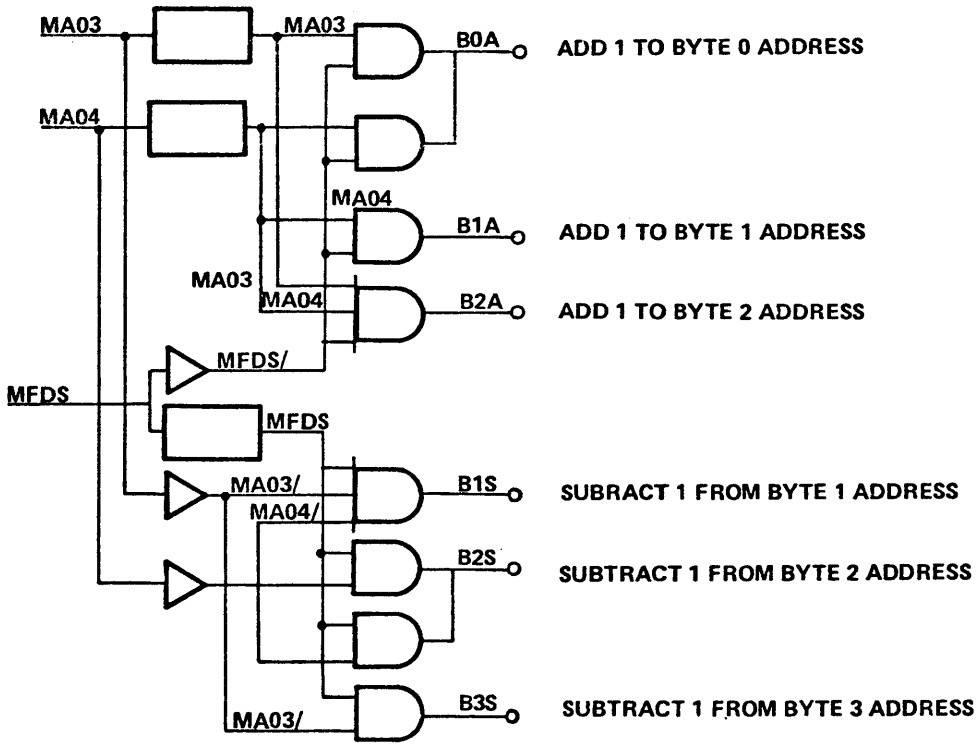
Byte Detection

The determining factor in whether or not address correction is required is the location of the key byte. The key byte's location is dependent upon the contents of memory address lines 03 and 04. If the byte address is 00 and the field direction sign is false (indicating a forward read or write), all bytes to be accessed are located in the same word of memory. The same is true when the byte address is 03 and the read/write is to be done in the reverse direction. In all other cases, the required bytes are located in different words of memory, and address correction is required.

Byte detection is provided by logic shown in figure 2-188. This circuit decodes the contents of MA03, MA04, and MFDS producing the basic add and subtract levels which control address gating to the storage elements:

B0A	add 1 to byte 0 address	MFDS false
B1A	add 1 to byte 1 address	
B2A	add 1 to byte 2 address	
B1S	subtract 1 from byte 0 address	MFDS true
B2S	subtract 1 from byte 1 address	
B3S	subtract 1 from byte 2 address	

Note that B0A, B0A, and B1A, or B0A, B1A, and B2A may be true if MFDS is false, depending on the location of the key byte. Likewise, B3S, B3S and B2S, or B3S, B2S, and B1S may be true if MEDS is true and the appropriate key byte is selected.



CARRY LOGIC

CRRY06 = MA05 * MA06
CRRY07 = MA05 * MA06 * MA07
CRRY08 = MA05 * MA06 * MA07 * MA08
•
•
CRRY17 = MA05 * MA06 * – – – * MA16 * MA17
CRRY18 = MA05 * MA06 * – – – * MA16 * MA17 * MA18

BORROW LOGIC [MAn..I..EQUALS MAn INVERTED]

BRRW06 = MA05..I * MA06..I
BRRW07 = MA05..I * MA06..I * MA07..I
•
•
•
BRRW17 = MA05..I * MA06..I * – – – * MA16..I * MA17..I
BRRW18 = MA05..I * MA06..I * – – – * MA16..I * MA17..I * MA18..I

Figure 2-188. Key Byte Decoding

Address Modification Logic

Each memory address line from 05 through 18 (word address) goes through a circuit such as shown in figure 2-189. This circuit, in conjunction with those adjoining, is capable of incrementing the address by one, decrementing by one, or passing it unchanged. Which event occurs is dependent on the outputs of the key byte detection circuit previously discussed. The incremented/decremented addresses simply represent the gated outputs of the next lower-carry logic or next higher-borrow logic which takes the place of the furnished address when the appropriate add or subtract (BmA or BmS) signal is present. Note that separation of the address into four channels occurs at this point. The carry and borrow outputs are derived through simple AND gating (not shown). To provide single signal control of the modification procedure, the furnished address is subjected to a double inversion process.

Additional gating is employed at the MA18 level to produce the signals A18.00 and A18.00/ through A18.03 and A18.03/. These, along with the A17.nn signals, go to the enable and enable not inputs of two DFANs on each storage card.

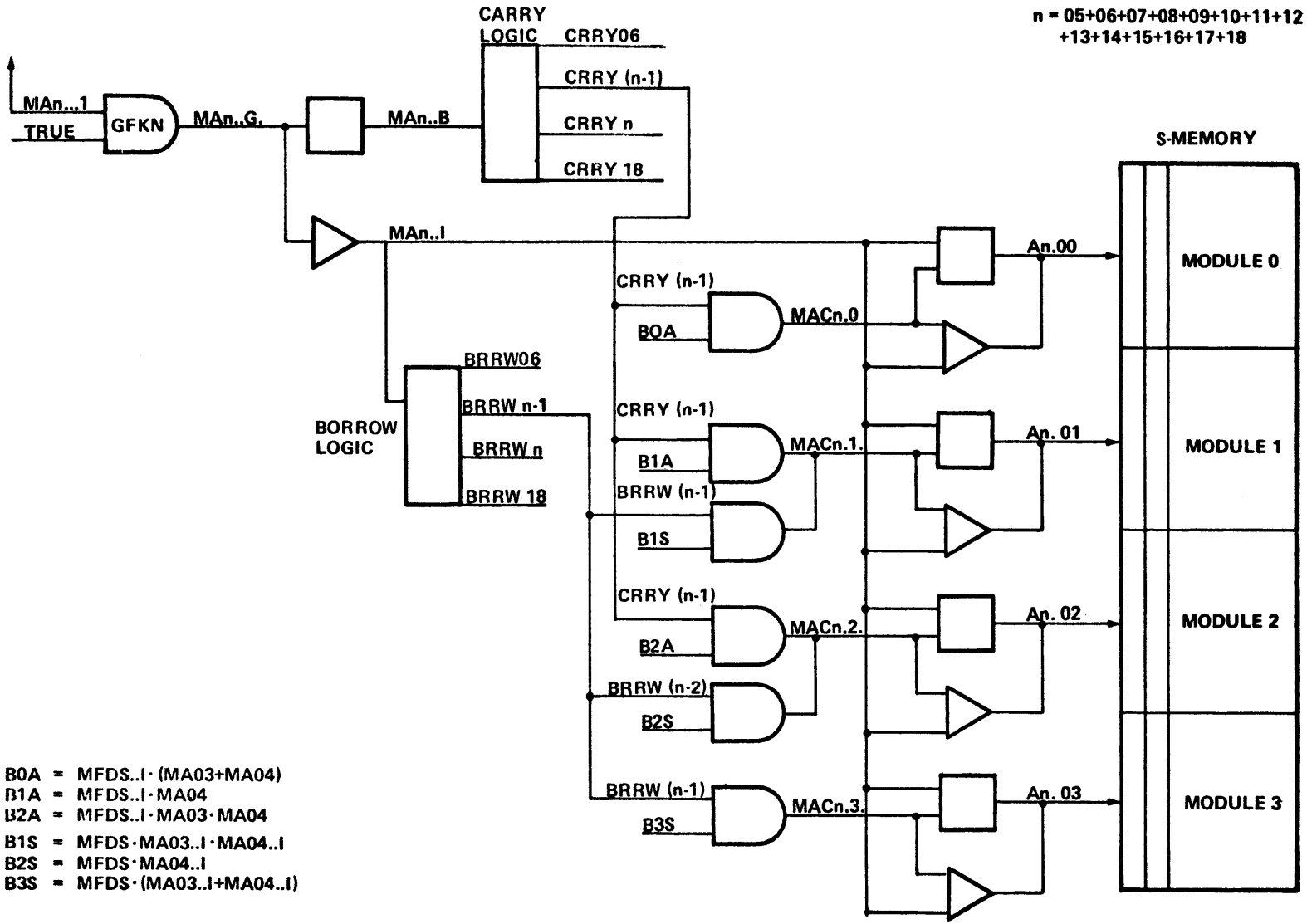


Figure 2-189. Address Modification Logic

Card Group Addressing

Card group addressing is accomplished in the same manner as word addressing, and is, in actuality, an extension of the address modification logic previously discussed. Enabling via four channels on a per-byte basis is continued at the card group level, since the requirement for individual control still exists. Card group enabling is controlled by the (corrected) contents of address lines MA17 and MA18. These go to the enable (E0) and enable not (E1) inputs of two DFAN chips on each storage card, one of which controls gating of the chip enable, precharge, and write enable signals for each byte column of storage elements. To provide the proper control levels for allowing card group selection by the DFANs, additional gating elements are employed on the MA18 line to produce the signals A18.001, A18.011, A18.021, and A18.031., as shown in figure 2-190.

These, along with the A17.nn and A18.nn signals are connected to the card group DFAN inputs in the following configuration (by variation in backplane wiring):

Card group 0: A18.nn/ to enable inputs
A17.nn to enable not inputs

Card group 1: A17.nn to enable inputs
A18.nn to enable not inputs

Card group 2: A18.nn to enable inputs
A17.nn to enable not inputs

Card group 3: A17.nn to enable inputs
A18.nn/ to enable not inputs

Memory Unit Addressing

The memory unit address is contained in MA19 and MA20. These are decoded by way of a DFAN to produce levels with binary weights from 0 to 3 (UN.20, UN.21, UN.22, and UN.23). A jumper chip is wired on each field address card to derive from these levels the signals PREUN (preceding unit), THSUN (this unit), and NXTUN (next unit). The actual connections used vary depending on the unit position within memory, as shown in figure 2-191. If MA19 is true and MA20 is false, the addressed unit is unit 1. UN.21 is true on the field address cards in each unit, making NXTUN true in unit 0, THSUN true in unit 1, and PREUN true in unit 2. Within unit 1 in this example, THSUN true with GRPENBL1 (always true) makes UNTNBL true, disabling the inverter elements which block the A18.mm and A18.nn/ lines (figure 2-190) by forcing all of them true. A18.mm and A18.mm/ then assume their selected levels and address a card group within the unit.

UNTNBL can also be forced true by an address correction condition which occurs when a data field spanning unit boundaries is accessed. When an overlap from a higher unit occurs, UNTNBL is produced by NXTUN·BRR18·B3S; when from a lower unit, by PREUN·CRRY18·B3S.

Memory address lines 21, 22, and 23 are normally false, being of a binary weight beyond the system memory capability. These are decoded in a manner similar to MA18, 19, and 20 to make GRPENBL1 (group enable) true for a value of zero on 21, 22, and 23.

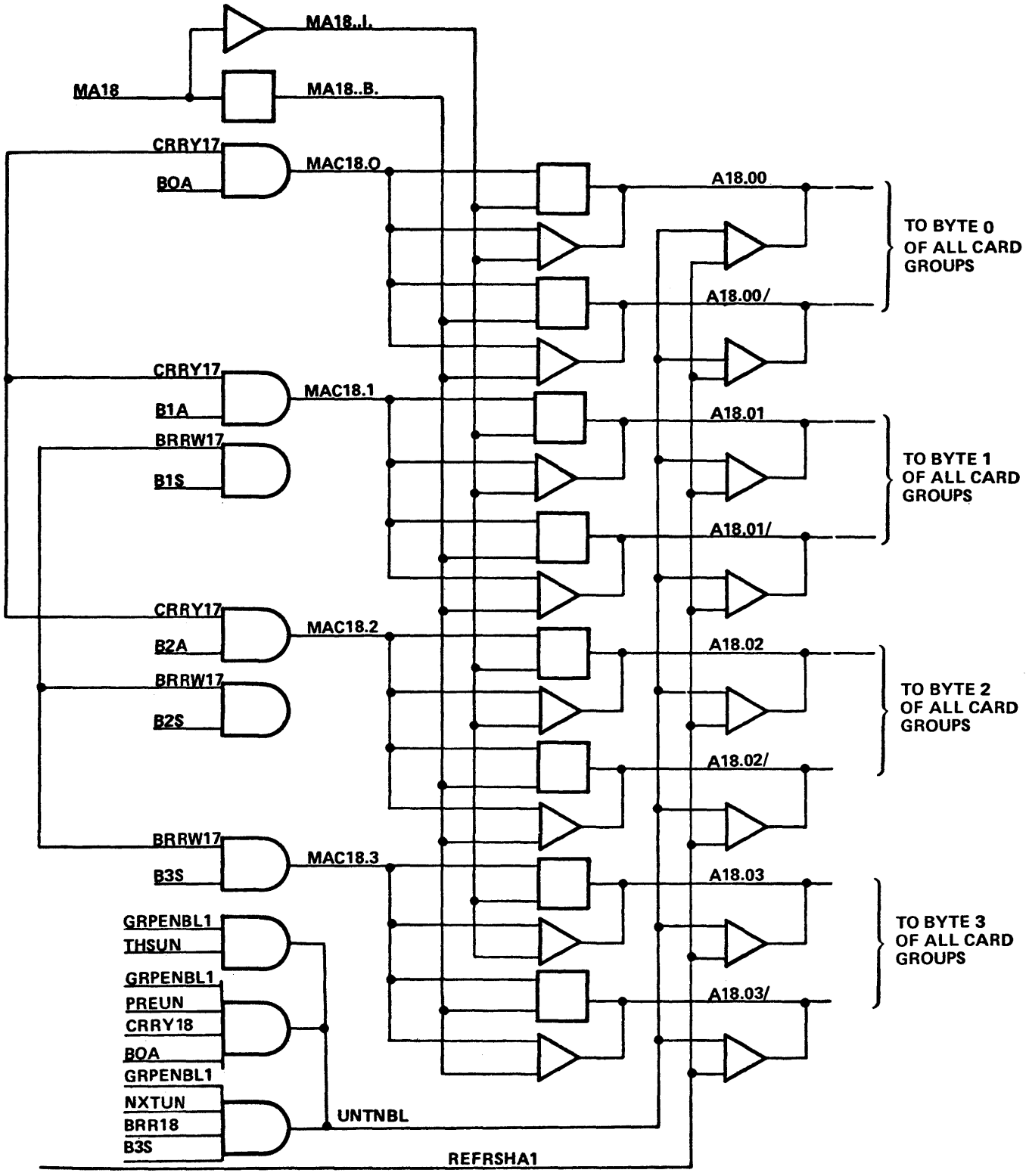


Figure 2-190. Memory Address Line 18 and Unit Enable Logic

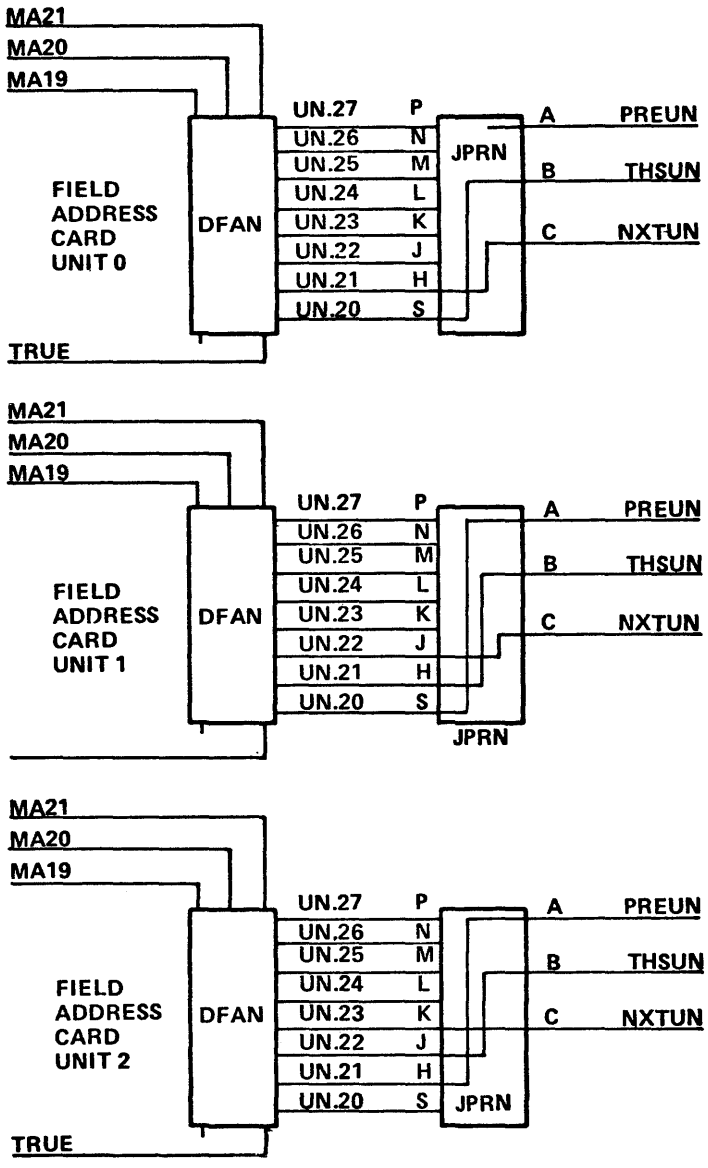


Figure 2-191. Unit Select Logic

This signal is used as an additional AND condition in generation of UNTNBL in all memory units. Therefore, if address lines 21, 22, and 23 contain some value other than zero, GRPENBL1 is not generated, and memory access is precluded. Refer to figure 2-192.

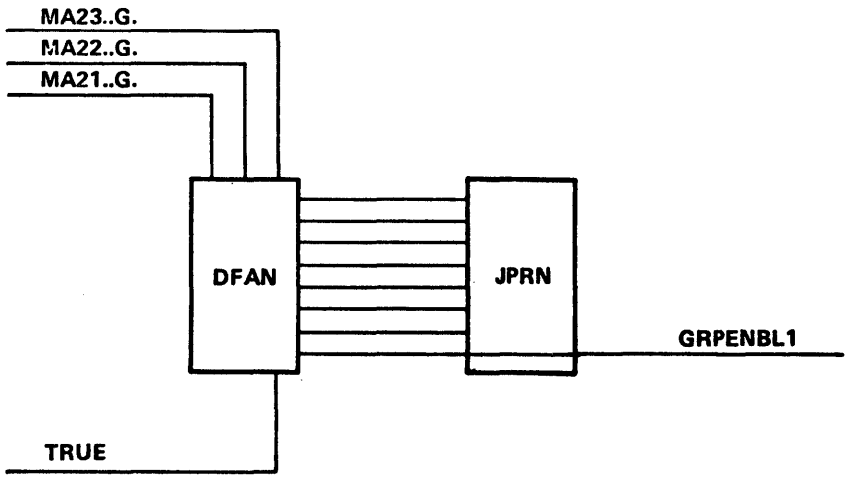


Figure 2-192. Group Enable Logic

INTERFACE CONTROL LOGIC

The interface control card contains the logic to receive and distribute memory write data, and to gate read data from bytes 0, 1, and 2 to the memory read lines. Output gating for byte 3, although a part of the IC logic, is physically located on the field address card. The memory write gating elements are shown in figure 2-193, and read gating in figure 2-194. The enabling signals BnnSEL.1 (read lines) and MWDBnEN. (write lines) are generated externally in the port interchange.

In addition to the above, the interface control card generates the memory clock signals PRE (precharge), CE (chip enable), and WE (write enable). These are all derived from the system clock by way of delay lines and flip-flops as shown in figures 2-195 and 2-196. Memory read timing is shown in figure 2-197.

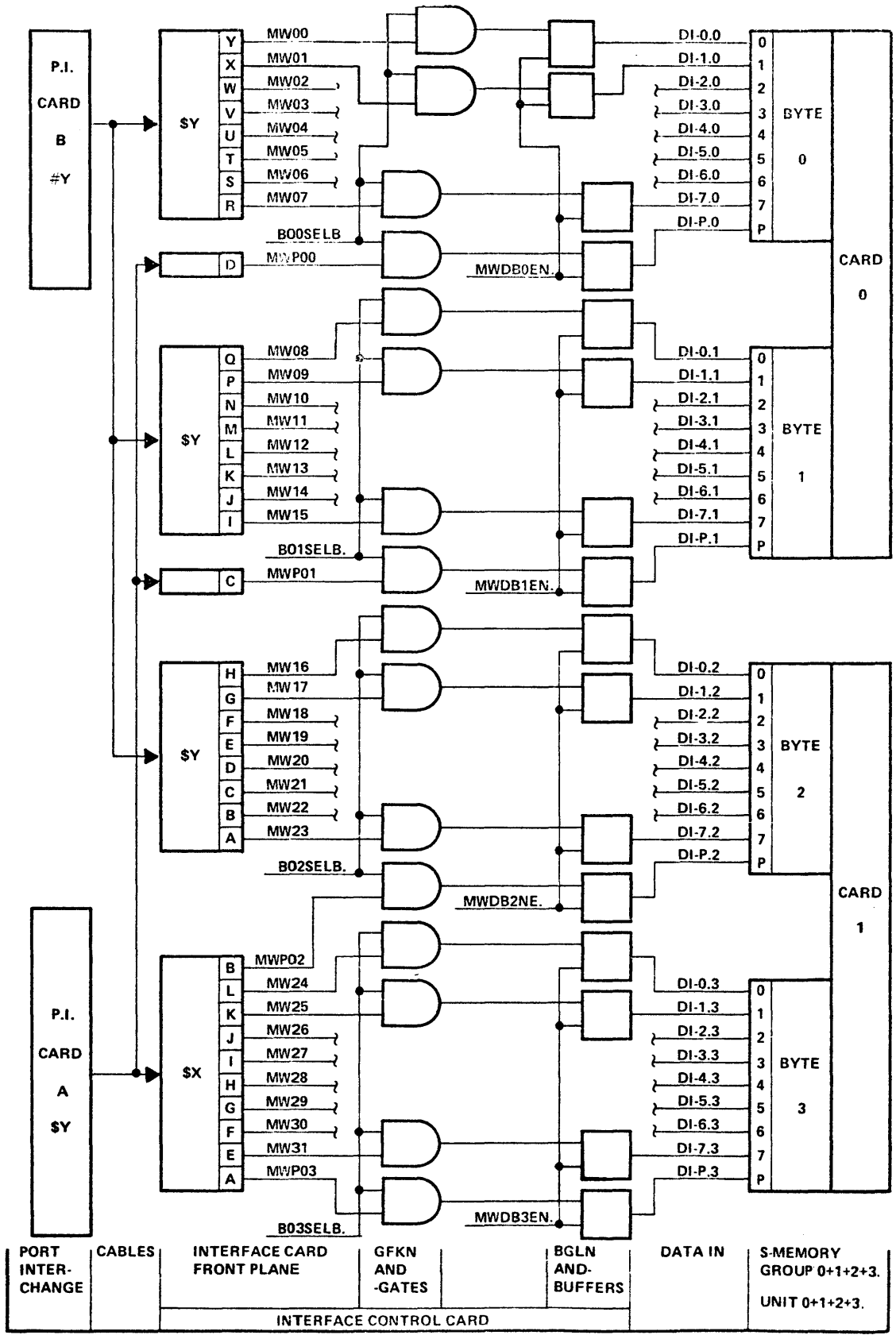


Figure 2-193. Memory Write Lines Distribution

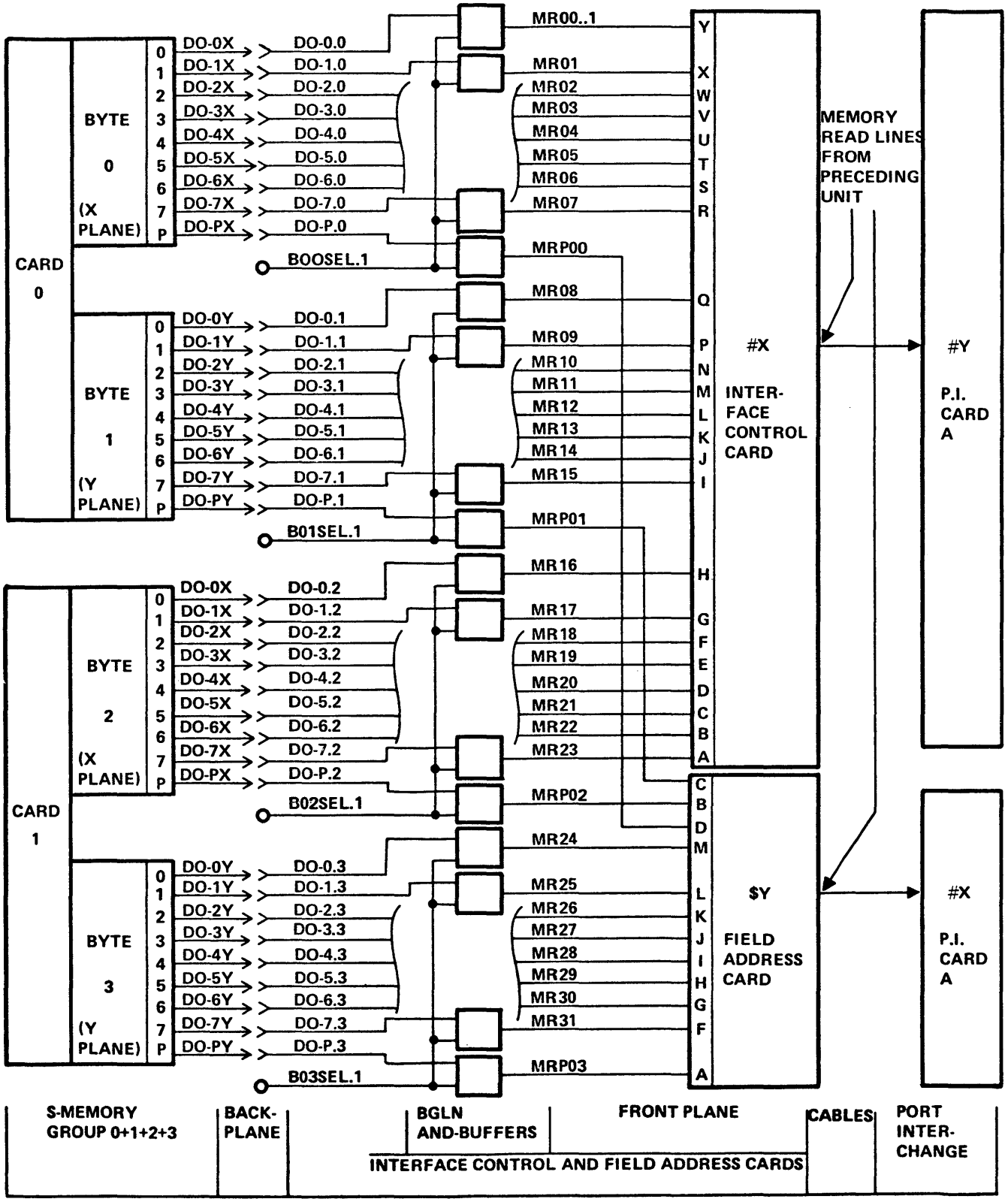


Figure 2-194. Memory Read Lines

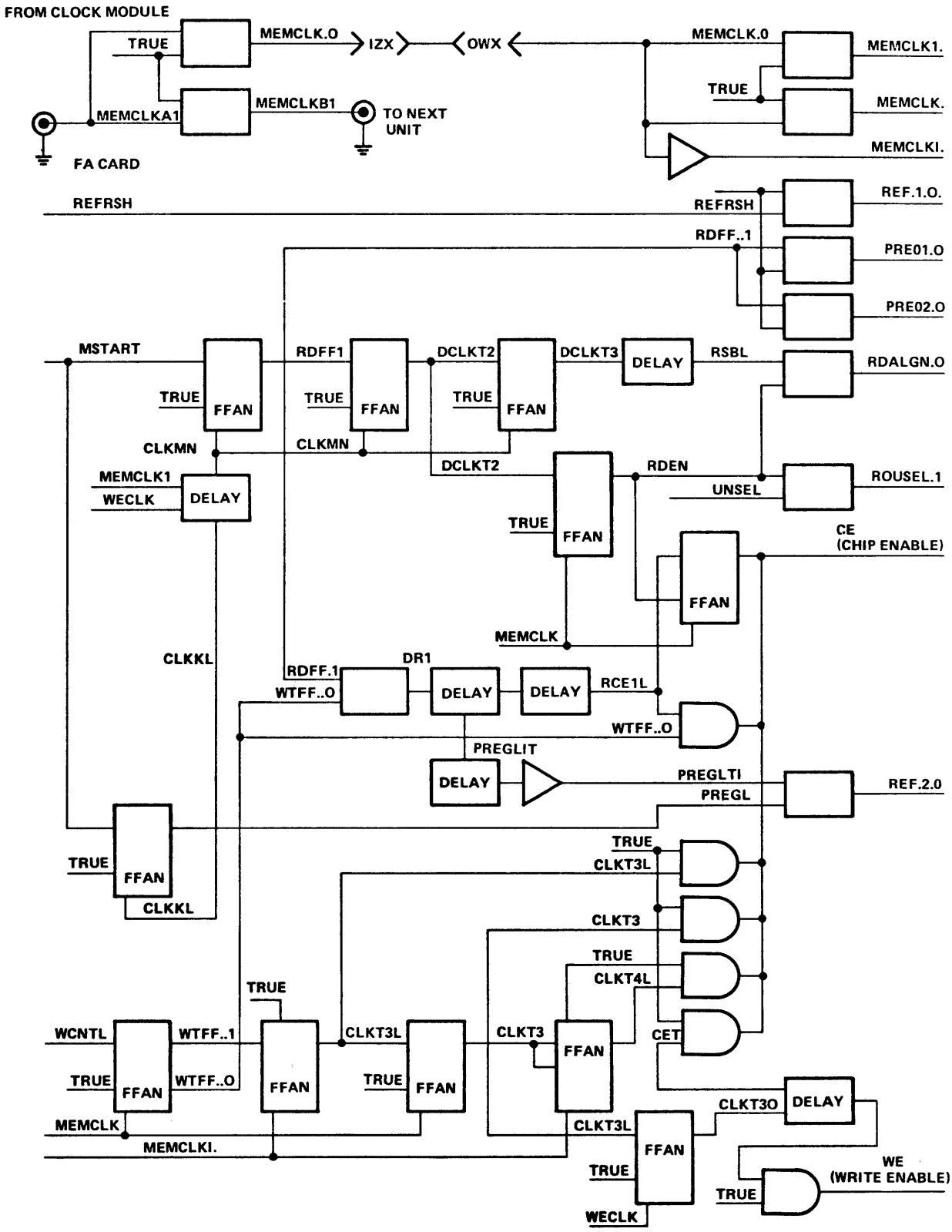


Figure 2-195. Memory Clocks Generation

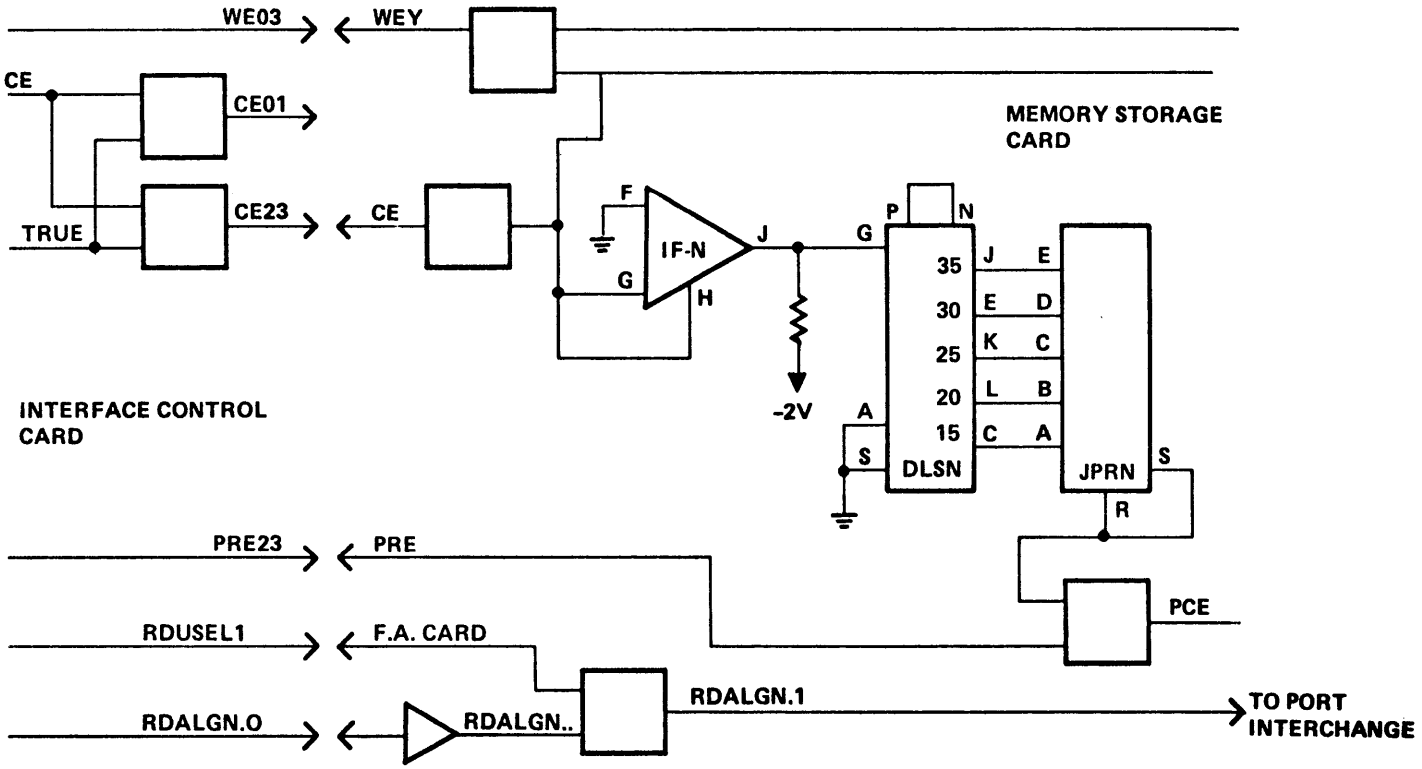


Figure 2-196. Memory Clocks Distribution

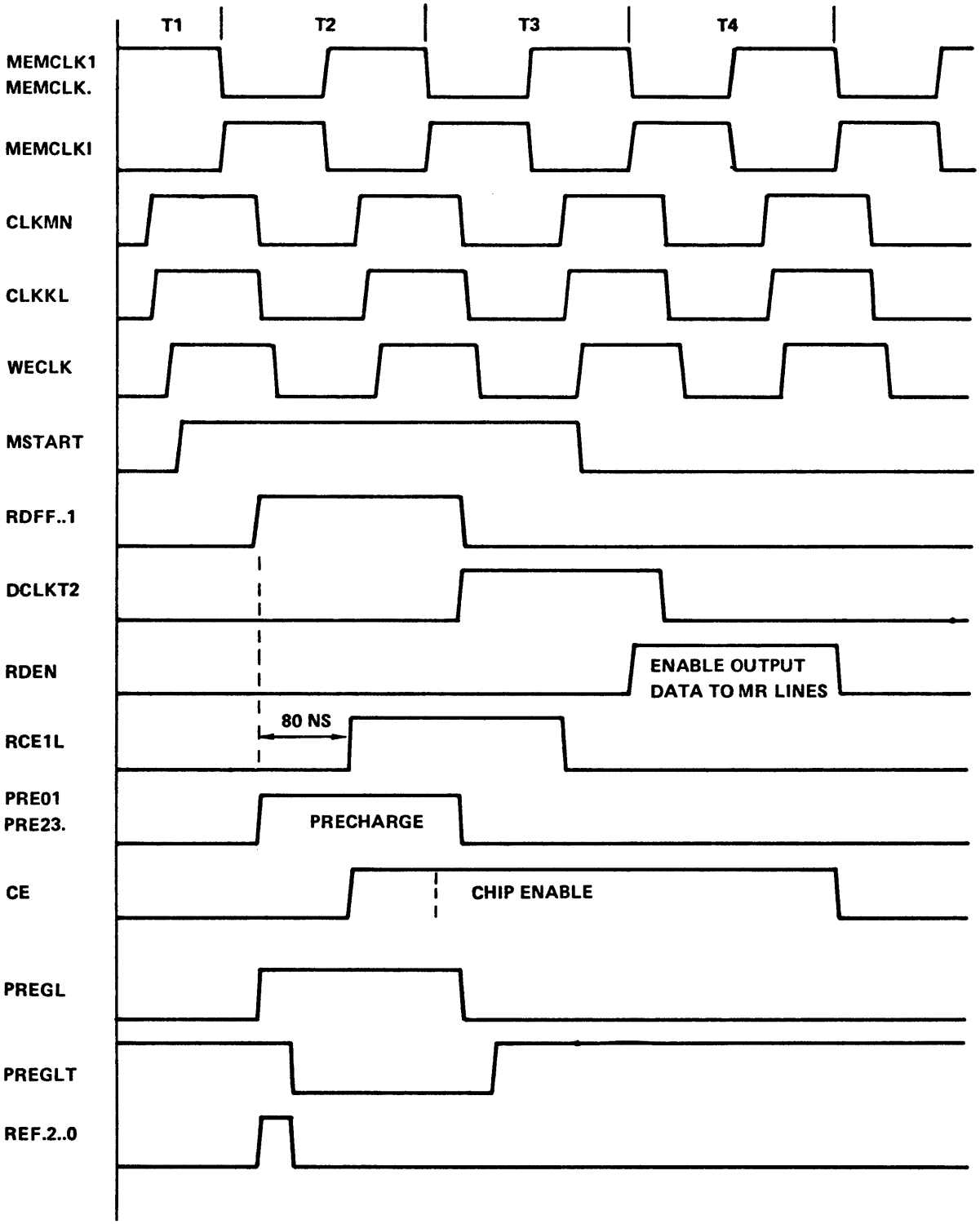


Figure 2-197. Memory Timing - Read

SOFT I/O CONCEPT

Transfer of data between the processor and the I/O subsystem is controlled by a micro program referred to as the I/O driver routine. The I/O driver routine consists of a string of micros which perform the required data movement and checking functions, and is entered whenever an I/O operation is required. Prior to beginning an I/O operation, a 24-bit reference address must be placed in the L-register, and the channel number (of the designated I/O device control) in the T-register. The reference address points to a location in S-memory where all pertinent information referring to the I/O operation is stored.

Once entered, the I/O driver routine continues executing until the conditions specified in the I/O descriptor stored at the reference address in memory have been fulfilled. Although any of the 32 possible micros may be executed within the I/O driver, only two have direct significance to the transfer of data to/from the I/O subsystem. These are the 1C and 2C micros.

I/O BUS

The I/O bus consists of 24 bi-directional lines between the processor and the I/O base. The 24 I/O bus lines are designated as PIOB00Q1 through PIOB23Q1, and, at the processor end, are buffered in both directions (figure 2-198). Data leaving the processor represents the buffered contents of the MEX, with gating to the I/O bus controlled by the signal DRVR=1Q1. Incoming data is gated by the signal RCVR=1Q1, but does not go directly to the MEX. Rather this data is subjected to a second level of gating which is controlled by the signal ALLOW.Q. (figure 2-199).

I/O INTERFACE CONTROL LOGIC

I/O Interface gating is controlled by the same source and sink control logic which selects the other processor registers as a function of micro decoding. This logic is discussed in the Basic Micro and Address Control portion of this section. Refer to that discussion for further information.

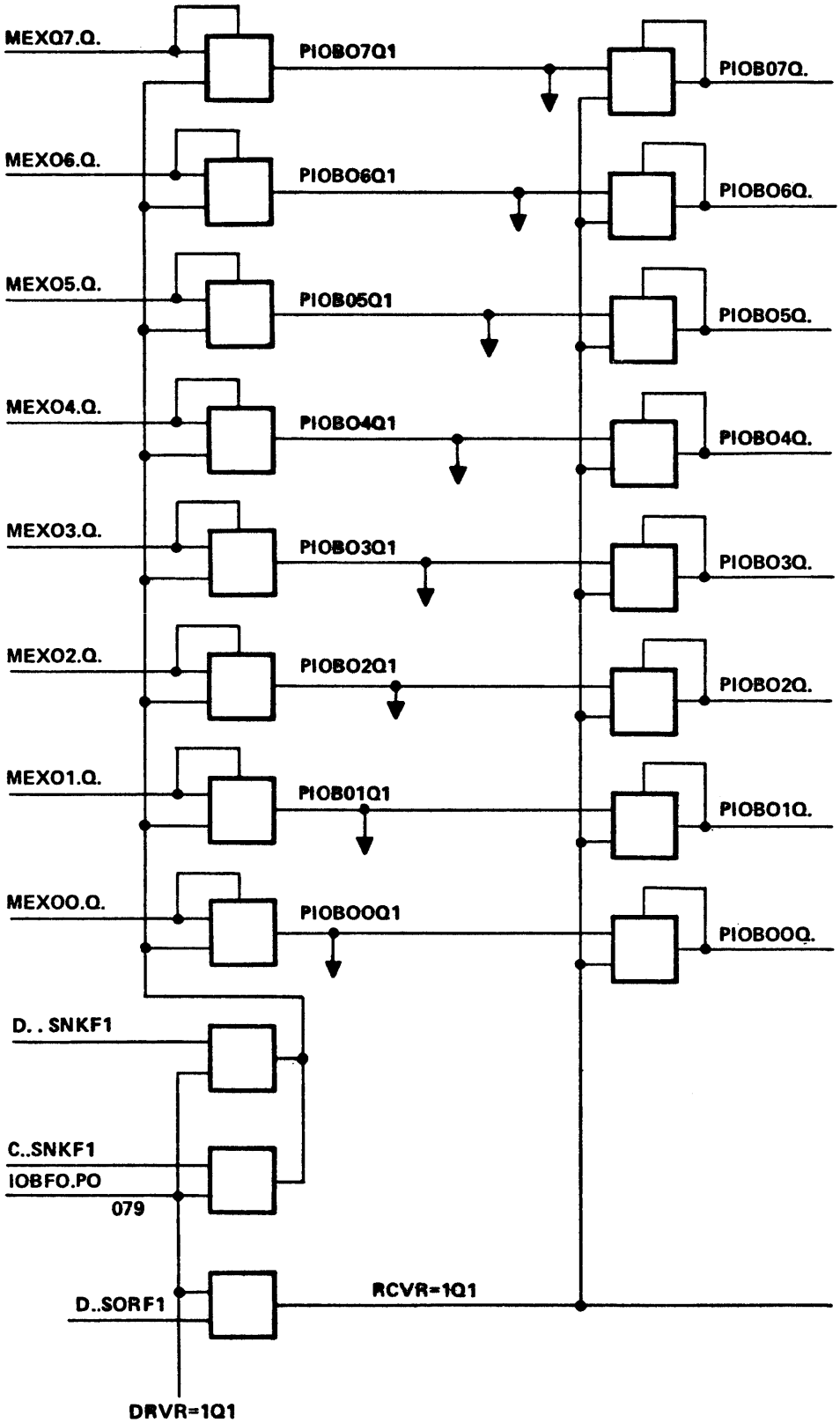


Figure 2-198. MEX/IO Bus Line Drivers and Receivers (Bits 00 - 07 Shown)

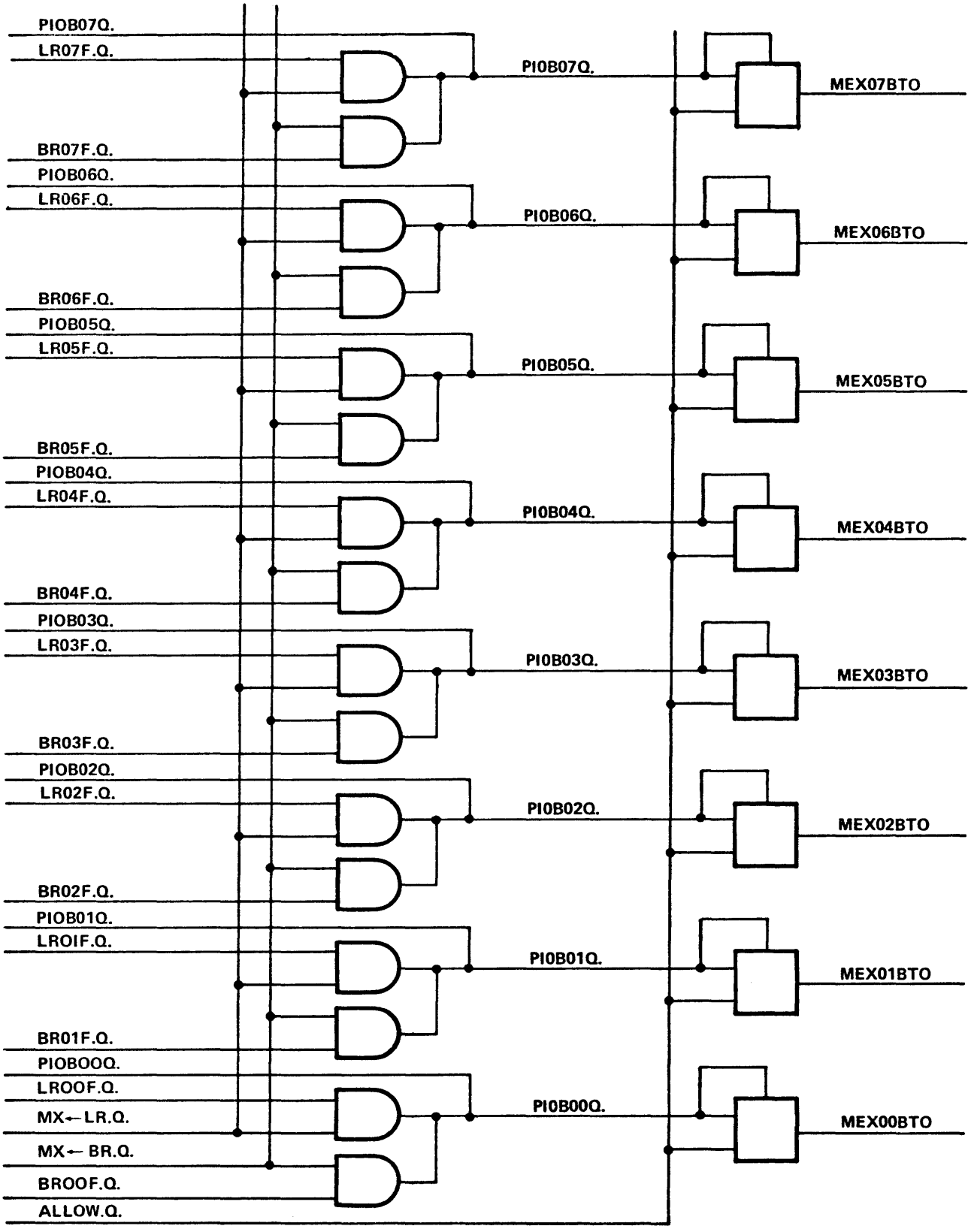


Figure 2-199. BR/LR Registers and I/O Bus to MEX Gating
(Bits 00 - 07 Shown)

SECTION 4
ADJUSTMENTS

INTRODUCTION

This section provides information that enables the Field Engineer to observe, check, and adjust the basic timing signals used in the B 1720 Model 1 processor.

PRELIMINARY STEPS

The following steps must be performed before any system adjustments are begun.

1. Determine that all system voltages are set to nominal values.
2. Use a calibrated oscilloscope that can display 5 nanoseconds per centimeter. (Example: Tektronix 465 at 0.05 microseconds per centimeter with the X10 switch on.)
3. Use two probes identical in type and length. Ground the probes, using short leads, to the backplane.
4. Check the probes and the oscilloscope preamps by attaching both probes to the same system clock backplane pin and overlaying the two traces. Any difference in signal fall time or amplitude must be corrected before proceeding. Use the following procedure:

Trigger	Internal positive
Channel One	System Clock at XEOW *
Channel Two	System Clock at XEOW *
Horizontal	0.05 usec/cm, X10 on (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Trigger Source	Channel One
Display mode	Alternate

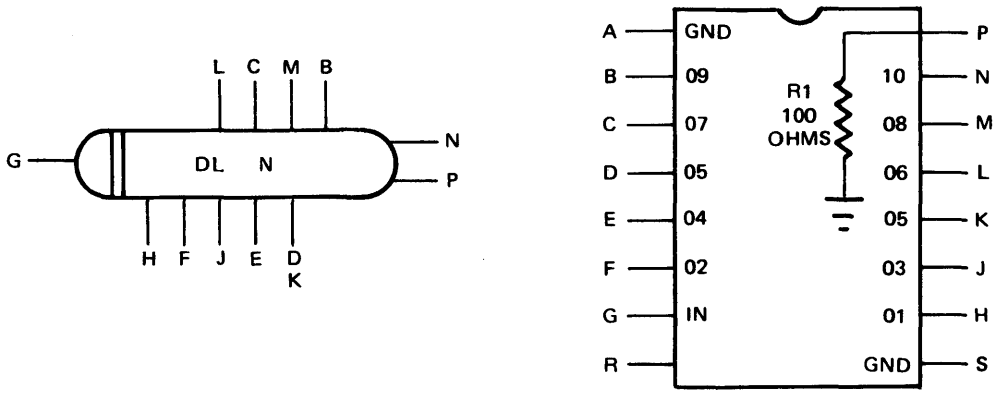
* processor backplane

Adjust ground of each channel 1.0 cm below the centerline so that the +1-volt level is the centerline.

5. Make all measurements in this section at the +1-volt level.
6. When making adjustments, follow the sequences given in this section.
7. Each adjustment in this section calls out "normal" delay line connections. These are listed as starting points only. Because of component variations, some adjustments may vary from the normal pins.
8. All schematics and partial schematics included as figures in this section are examples only. Actual wiring and pin locations are to be obtained from the applicable Field Test and Reference (FT&R) documentation.

DELAY LINES

Figure 4-1 shows the configuration of the three delay lines (DL2N, DL5N, DLCN) used in the B 1720 Model 1. Table 4-1 shows the applicable stepped delays. On wire-wrapped and hybrid cards, the actual delay line pins (taps) are used for the desired delay. On etched cards, header chips (JPRN) are used. To determine which header pin reflects the desired delay line tap, refer to the appropriate schematic.



G10633

Figure 4-1. B 1720 Processor Delay Lines

Table 4-1. B 1720 Delay Line Values (in nanoseconds)

Pin:	H	F	J	E	K,D	L	C
DL2N	2	4	6	8	10	12	14
DL5N	5	10	15	20	25	30	35
DLCN	10	20	30	40	50	60	70

CLOCK ADJUSTMENTS

Steps 1 through 8 provide instructions for checking and adjusting the processor and port interchange clocks. Figures 4-2 and 4-3 show the clock distribution paths within the processor and the port interchange for wire-wrapped and etched cards, respectively. Figures 4-4 through 4-11 are keyed to steps 1-8. Figure 4-12 is a comprehensive clock timing diagram.

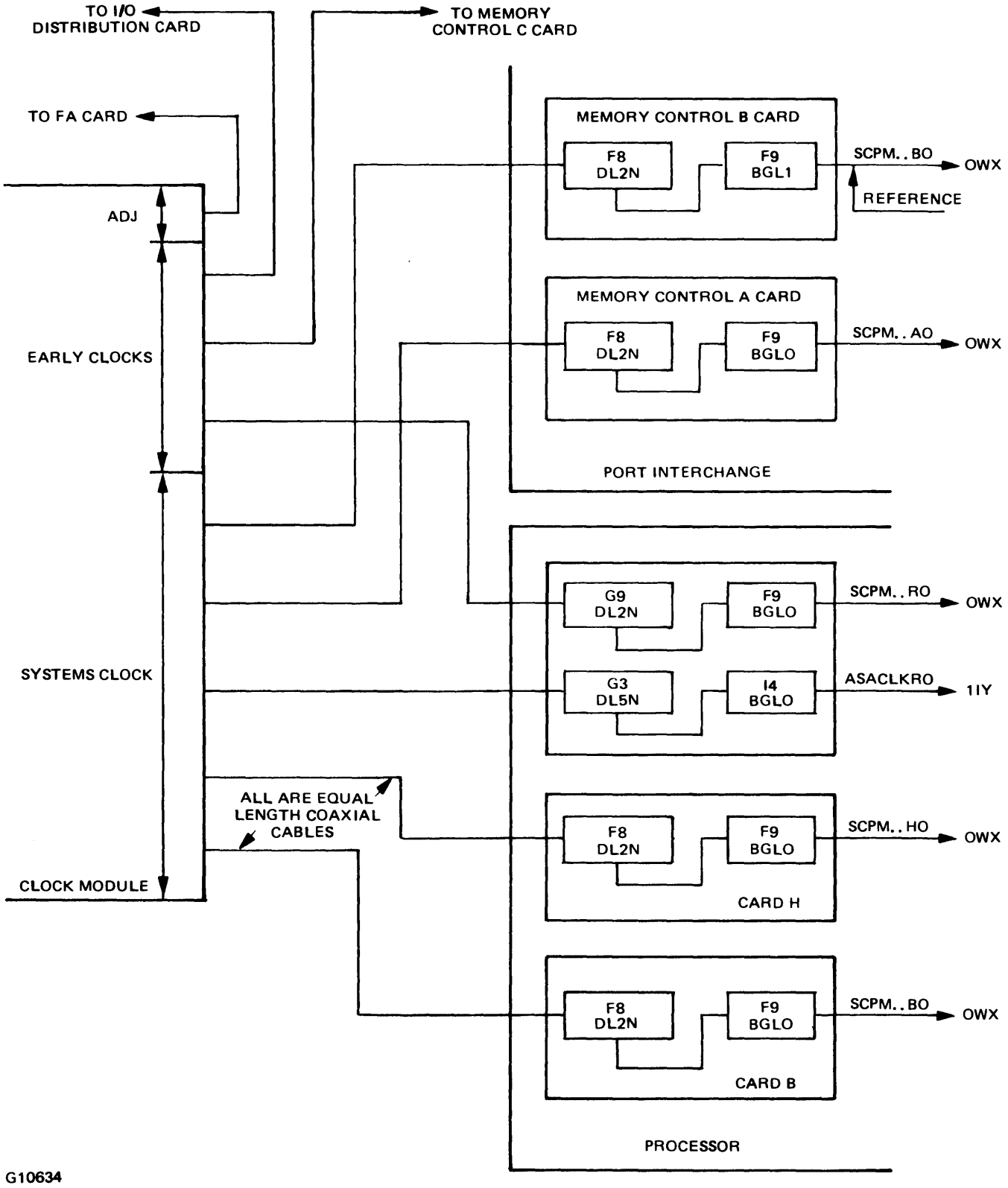


Figure 4-2. B 1720 Clock Circuits, Wire-Wrapped Cards

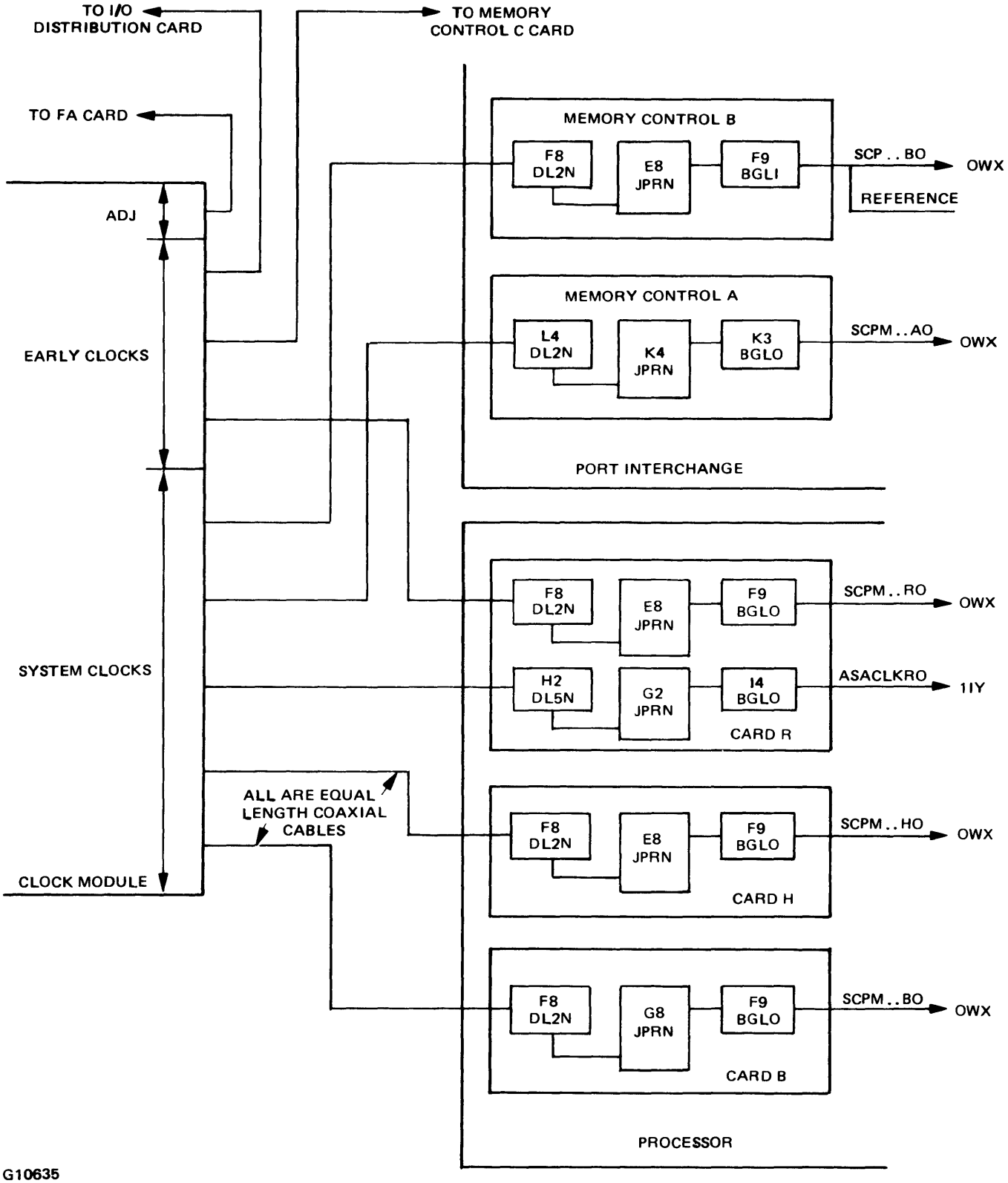


Figure 4-3. B 1720 Clock Circuits, Etched Cards

Step 1: Clock Module System Clock

Figure 4-4 shows the oscilloscope set-up and timing requirements for the System Clock output of the clock module assembly. The purpose of step 1 is to verify the amplitude and time period of this clock. There are no adjustments. The clock module provides System Clock outputs by means of 14 "doghouses"; each one must be checked. On the M1 system, the System Clock is routed to processor cards R, B, and H and port interchange cards A and B. Card 1 of the multi-line control also receives the System Clock.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	System Clock at each clock module doghouse
Channel Two	Not applicable
Horizontal	0.2 us/cm, X10 ON (20 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

Measurements (All at 1-volt level; all system voltages nominal.)

Symmetrical clock at 166.6 ns

Adjustments

none

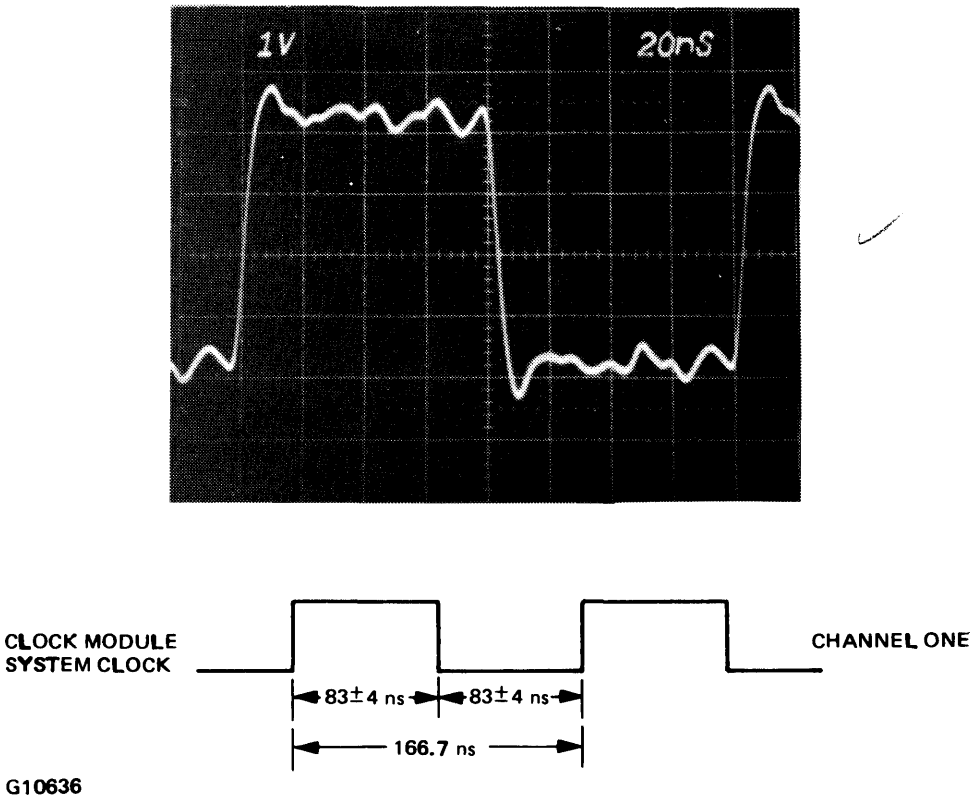


Figure 4-4. B 1720 Clock Module System Clock Timing

Step 2: Clock Module Early Clock

Figure 4-5 shows the oscilloscope set-up, adjustment procedure, and timing requirements for the Early Clock output of the clock module assembly. All 8 Early Clock doghouses must be checked. Early Clock is sent to processor card R, memory control card C, and the I/O distribution card.

Oscilloscope Set-up

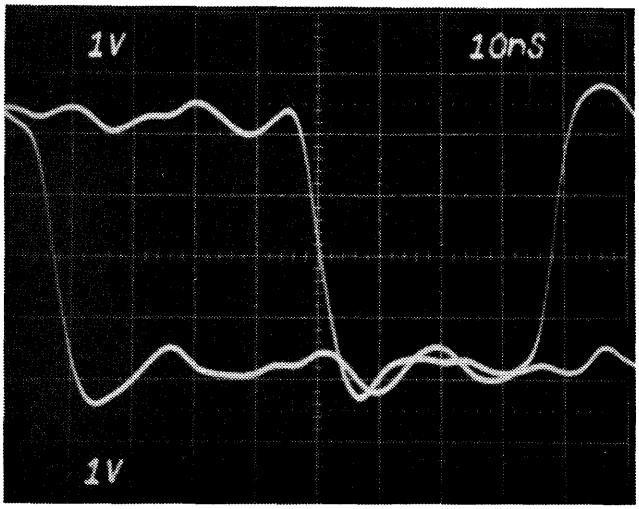
Trigger	Internal positive
Channel One	System Clock at doghouse going to port interchange card B
Channel Two	Early Clock at each doghouse
Horizontal	0.1 us/cm, X10 ON (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

Measurements (All at 1-volt level; all system voltages nominal.)

Trailing edge of Early Clock referenced to trailing edge of System Clock must be 40 ±3 ns.

Adjustments

JPRN G3 (normally G3J to G3P)



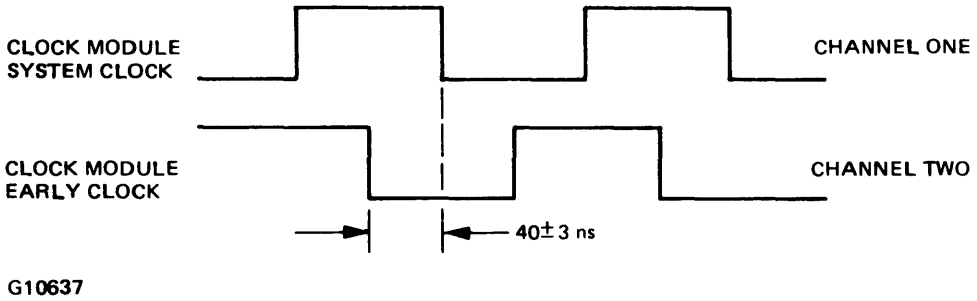


Figure 4-5. B 1720 Clock Module Early Clock Timing

STEP 3: Clock Module Adjustable Clock

Figure 4-6 shows the oscilloscope set-up, adjustment procedure, and timing requirements for the Adjustable Clock output of the clock module assembly. Check both Adjustable Clock doghouses. Adjustable Clock is routed to the field address card in the S-memory base.

CAUTION

Adjustment of the Clock Module Adjustable Clock requires reverification of Step 21 (located near the end of this section).

Oscilloscope Set-up

Trigger	Internal positive
Channel One	System Clock at doghouse going to port interchange card B
Channel Two	Adjustable Clock at each doghouse
Horizontal	0.1 us/cm, X10 ON (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

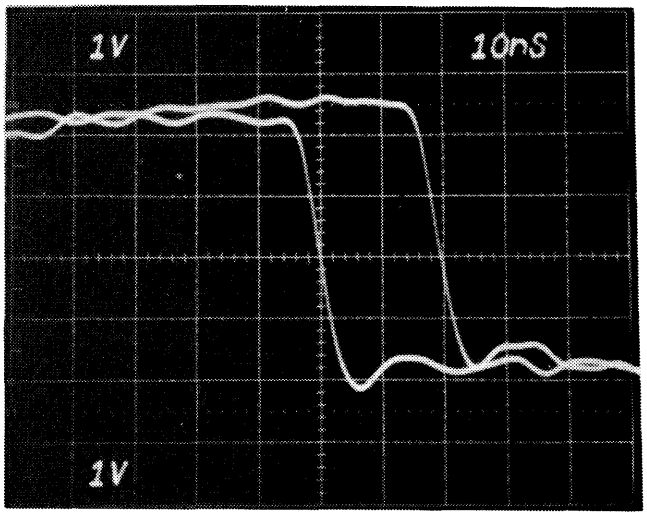
Measurements (All at 1-volt level; all system voltages nominal.)

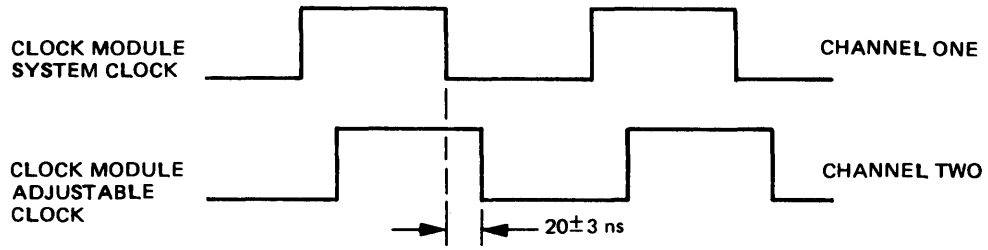
The trailing edge of Adjustable Clock must occur 20 ±3 ns after the trailing edge of System Clock.

Adjustments

JPRN E3 (normally E3K to E3L) ✓

ok. 1.2.90 Bu





G10638

Figure 4-6. B 1720 Clock Module Adjustable Clock Timing

STEP 4: Port Interchange Card B System Clock

Figure 4-7 shows the oscilloscope set-up, adjustment procedure, and timing requirements for signal SCPM..B0 on port interchange card B. SCPM..B0 becomes the reference clock for the rest of the system. The purpose of this step is to insure that the reference clock has been set so that other timing adjustments referencing SCPM..B0 will have enough tolerance (delay line taps) to permit required adjustments.

CAUTION

Adjustment of SCPM..B0 (System Reference Clock) requires verification of all other system timings including I/O clocks.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	clock module System Clock (any doghouse)
Channel Two	SCPM..B0, port interchange backplane XA3W
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edge of SCPM..B0 must occur 25 ±3 ns after the trailing edge of clock module System Clock.

Adjustments

Wire-wrapped card: DL2N F8 (normally F8K to F9H)

Etched card: JPRN E8 (normally E8E to E8R)

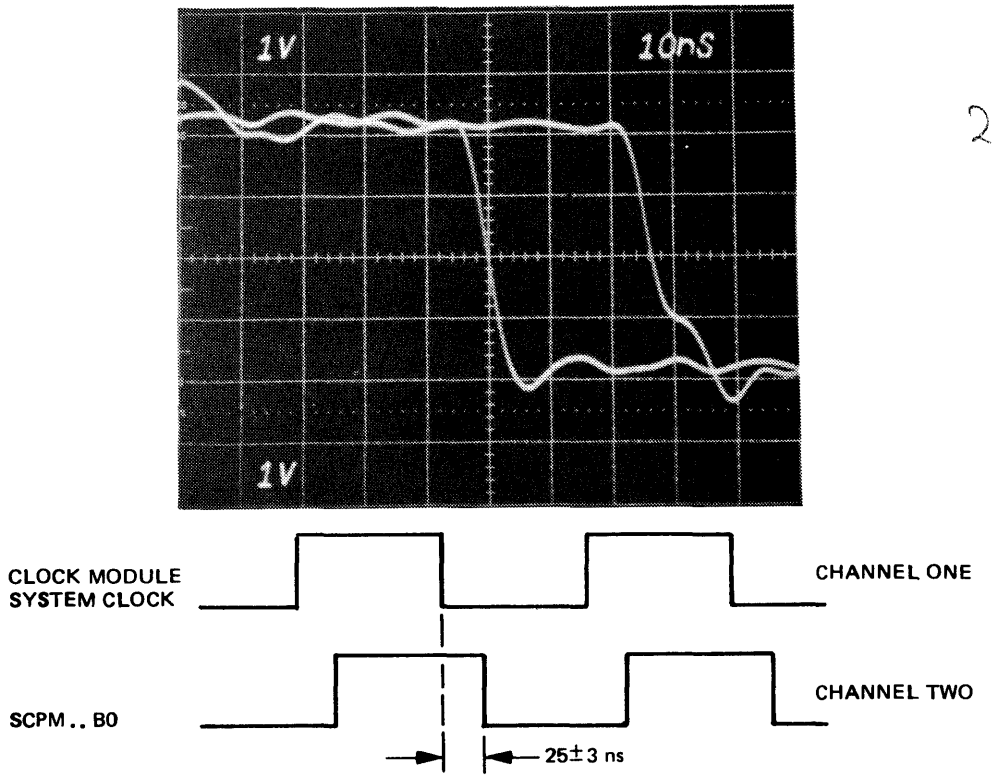


Figure 4-7. B 1720 Port Interchange Card B System Clock Timing

Step 5: Port Interchange Card A System Clock

Figure 4-8 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the System Clock on port interchange card A.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..B0, port interchange backplane XA3W
Channel Two	SCPM..A0, port interchange backplane XA6W
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

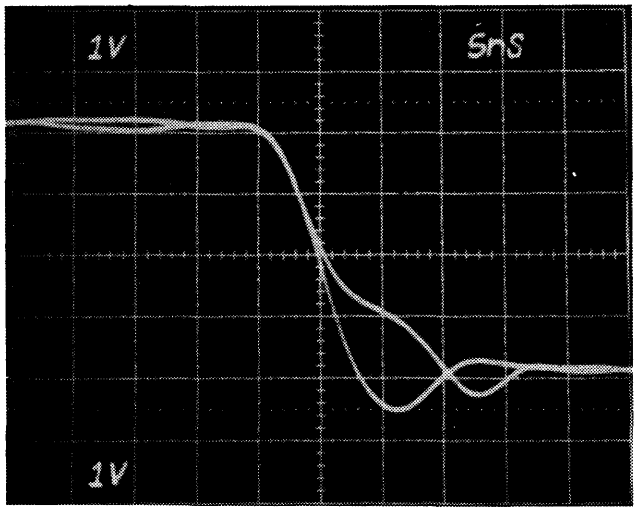
Measurements (All at 1-volt level; all system voltages nominal.)

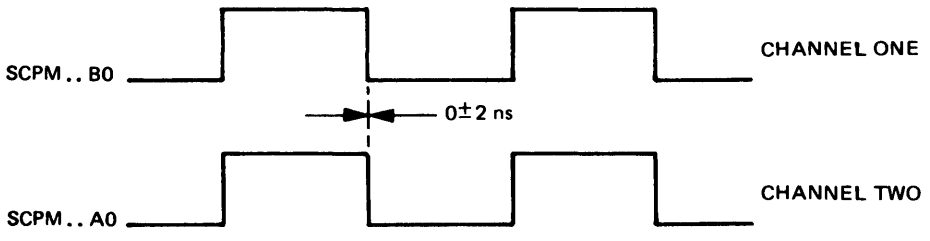
The trailing edges of SCPM..A0 and SCPM..B0 should be coincident (0 ±2 ns).

Adjustments

Wire-wrapped card: DL2N F8 (normally F8K to F9B)

Etched card: JPRN C2 (normally C2E to C2R)





G10640

Figure 4-8. B 1720 Port Interchange Card A System Clock Timing

Step 6: Processor Card B System Clock

Figure 4-9 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the System Clock on processor card B. Card B has two System Clock outputs and drives logic on cards B, A, J, K, and L.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..B0, port interchange backplane XA3W
Channel Two	SCPM1.B0, port interchange backplane XC1Z
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

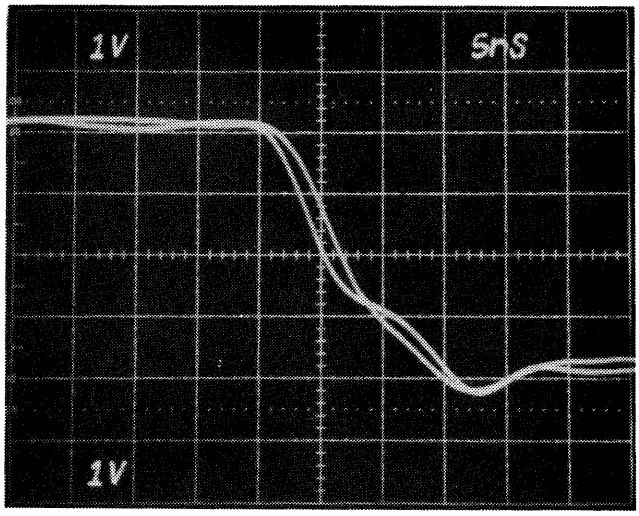
Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edges of both card B outputs and port interchange SCPM..B0 must be coincident (0 ±2 ns).

Adjustments

Wire-wrapped card: DL2N F8 (normally F8K to F9P)

Etched card: JPRN G8 (normally G8K to G9P)



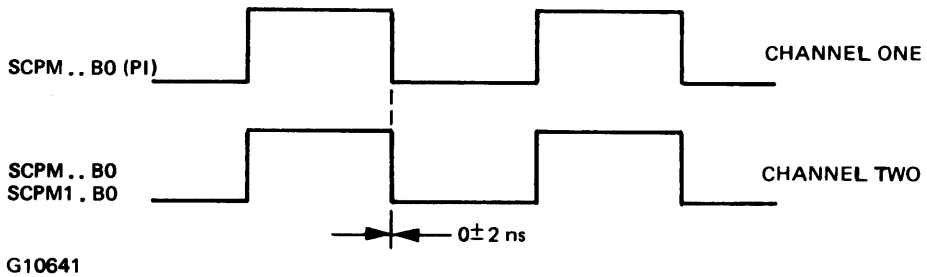


Figure 4-9. B 1720 Processor Card B System Clock Timing

Step 7: Processor Card H System Clock

Figure 4-10 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the System Clock on processor card H. SCPM..H0 drives logic on processor cards H, E, F, G, P, and Q.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..B0, port interchange backplane XA3W
Channel Two	SCPM..H0, processor backplane XA3W
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

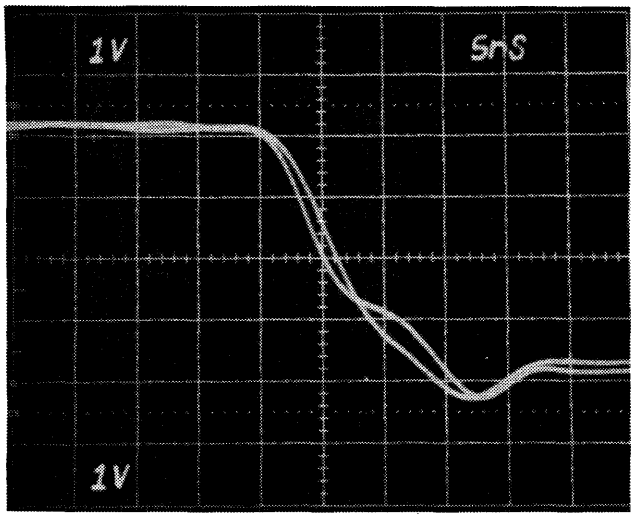
Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edges of SCPM..H0 and SCPM..B0 must be coincident (0 ±2 ns).

Adjustments

Wire-wrapped card: DL2N F8 (normally F8K to F9B)

Etched card: JPRN E8 (normally E8E to E8A)



2 ns

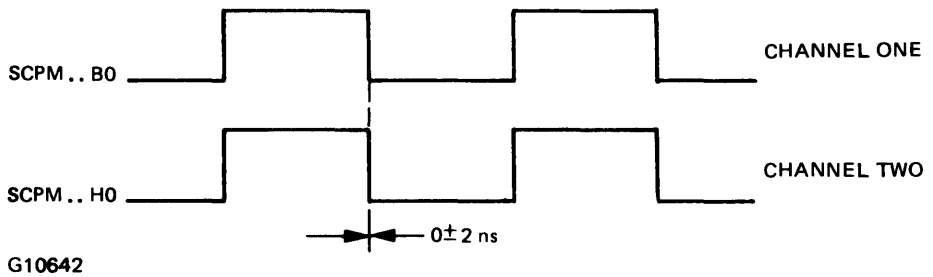


Figure 4-10. B 1720 Processor Card H System Clock Timing

Step 8: Processor Card R System Clock

Figure 4-11 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the System Clock on processor card R. SCPM..R0 drives logic on cards R, C, D, M, and N. Refer to figure 4-12 for a comprehensive clock timing example.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..B0, port interchange backplane XA3W
Channel Two	SCPM..R0, processor backplane XE0W
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

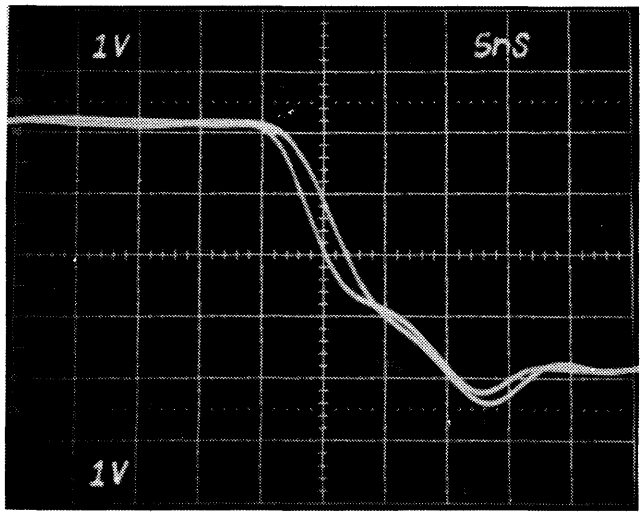
Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edges of SCPM..R0 and SCPM..B0 must be coincident (0 ±2 ns).

Adjustments

Wire-wrapped card: DL2N F8 (normally F8K to F9B)

Etched card: JPRN E8 (normally E8E to E8R)



-2ns

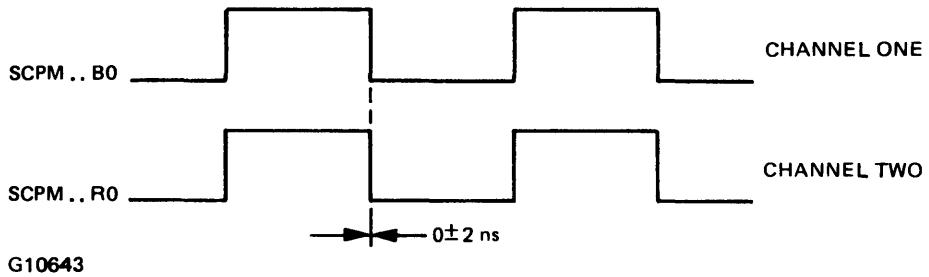
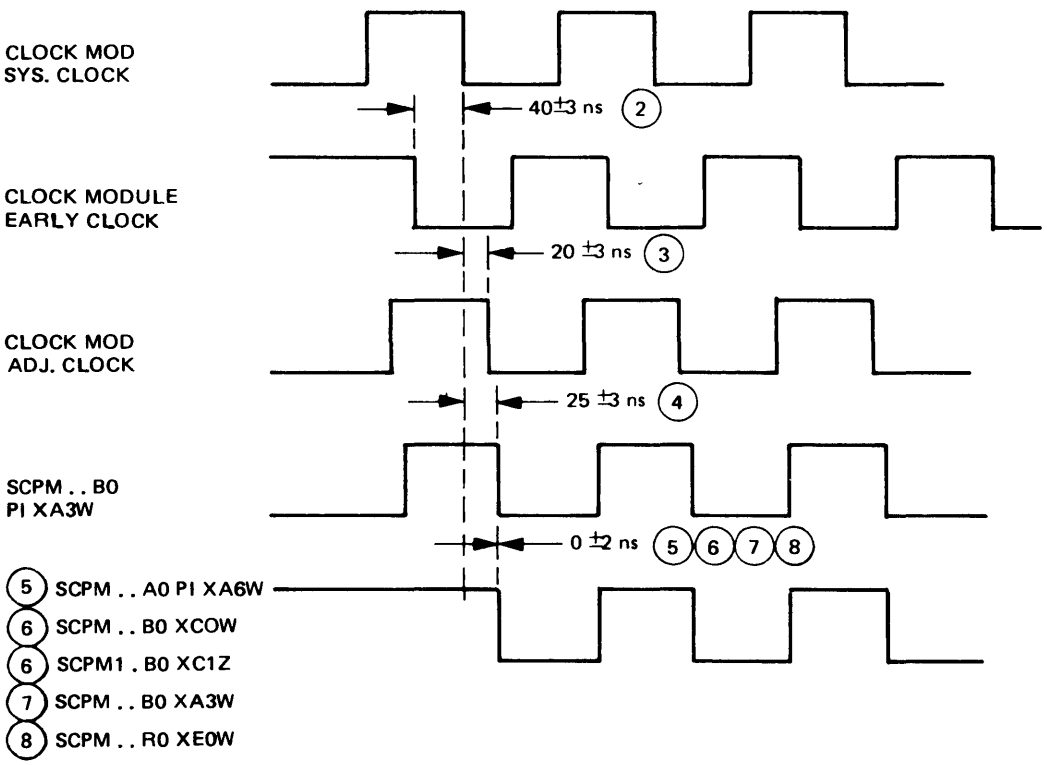


Figure 4-11. B 1720 Processor Card R System Clock Timing



NOTE: THE CIRCLED NUMBERS REFER TO THE STEPS IN THE ADJUSTMENT PROCEDURE.

G10644

Figure 4-12. B 1720 Comprehensive Clock Timing

MEMORY WRITE ENABLE CHECK; SCRATCHPAD AND A-STACK ALIGNMENT

Steps 9 through 15 provide information for checking and adjusting M-memory Write Enable, Scratchpad, and A-Stack. The programs listed in tables 4-2 and 4-3 are referenced in these procedures. Figures 4-13 through 4-18 are keyed to steps 9 through 15. Figure 4-20, at the end of this subsection, provides comprehensive Scratchpad and A-Stack timings.

Table 4-2. B 1720 Scratchpad Write Routine

A Register	MSM Register	Description
000000	0900	Monitor
000001 000002	07F0 07F0	Exchange DPW source address of 0, sink address of 15
000003 000004	070F 070F	Exchange DPW source address of 15, sink address of 0
000005	8400	Lit 0 to A register.

Table 4-3. B 1720 A-Stack Write Routine

A Register	MSM Register	Description
000000	10AB	Move X register to TAS
000001	8400	Lit 0 to A register

Step 9: M-String Write Enable

Figure 4-13 shows the oscilloscope set-up and timing requirements for MSWREN0 referenced to SCPM..R0. MSWRENM0 is checked for both trailing edge alignment and negative-going pulse width. There is no adjustment; if the specified measurement is not met, faulty components must be suspected.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..R0, processor backplane pin XE0W
Channel Two	MSWRENM0, processor backplane pin XF0Z
Horizontal	0.1 us/cm, X10 (10ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle; TRUE is forced on processor backplane pin XF0F to bring WRITMSD0 TRUE

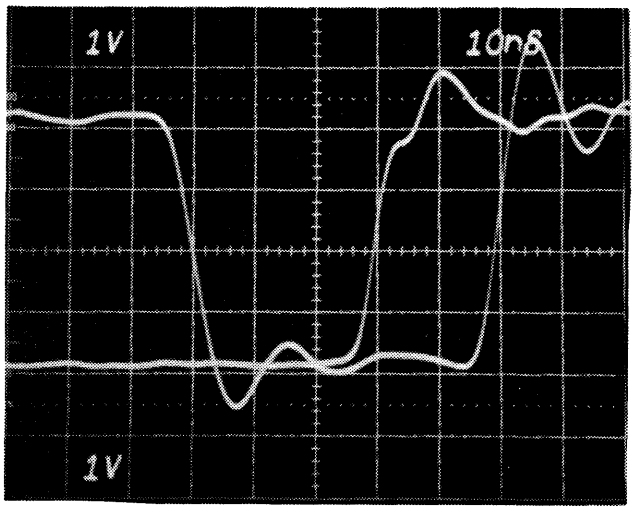
Measurements (All at 1-volt level; all system voltages nominal.)

MSWRENM0 must rise 14 ±7 ns after the trailing edge of SCPM..R0.

MSWRENM0 negative-going pulse width must be 57 ±8 ns.

Adjustments

None. (Check components if readings indicate error.)



20ns ok.

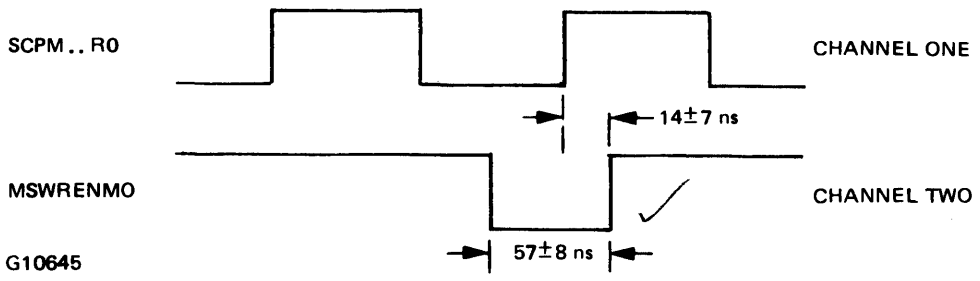


Figure 4-13. B 1720 M-String Write Enable Timing

Step 10: A-Stack Address Clock

Figure 4-14 shows the oscilloscope set-up, adjustment procedures, and timing requirements for ASACLKR0 referenced to the trailing edge of SCPM..R0. ASACLKR0 is derived by delaying the Early Clock received on processor card R from the clock module assembly.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..R0, processor backplane XE0W
Channel Two	ASACLKR0, processor backplane YE1I
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

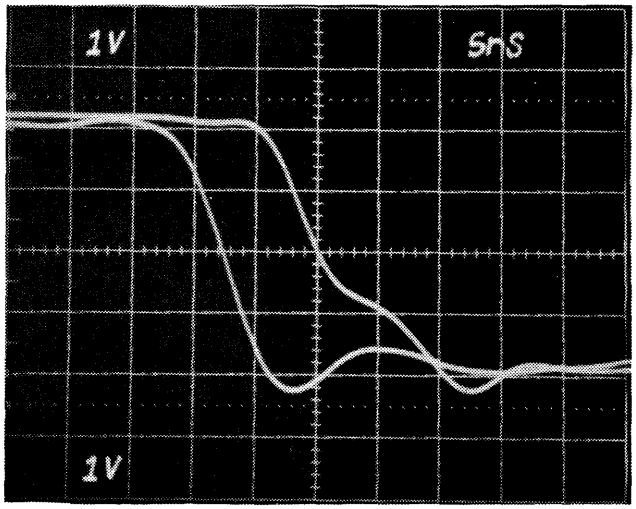
Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edge of ASACLKR0 must occur 8 ±4 ns before the trailing edge of SCPM..R0.

Adjustments

Processor card R, wire-wrapped: DL5N G3 (normally G3B to I4P)

Processor card R, etched: JPRN G2 (normally G2C to G2S)



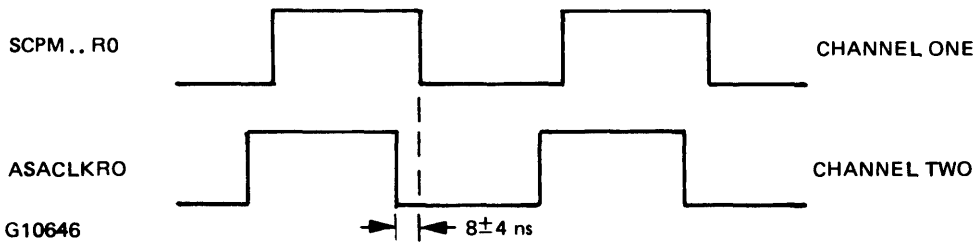


Figure 4-14. B 1720 A-Stack Address Clock Timing

Step 11: Scratchpad Write Enable

Figure 4-15 shows the oscilloscope set-up, adjustment procedures, and timing requirements for Scratchpad Left Write Enable and Scratchpad Right Write Enable referenced to SCPM..R0.

Oscilloscope Set-up

Trigger	External positive
Channel One	SCPM..R0, processor backplane XE0W
Channel Two	PLWE/.R0, processor backplane YE1F PRWE/.R0, processor backplane YE1S
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Net MNTR0.P0, processor backplane XBOH
Machine State	Scratchpad Write Routine (table 4-2) executing

Measurements (All at 1-volt level; all system voltages nominal.)

The negative-going edge of Scratchpad Write Enable (PLWE/.R0 and PRWE/.R0) must be 10 ±5 ns after the trailing edge of SCPM..R0.

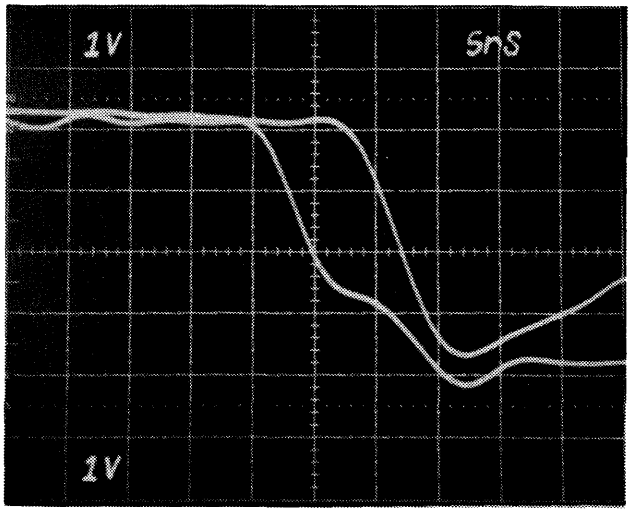
Adjustments

was 4ns and H3G - H4R

Processor card R, wire-wrapped: DL5N H3 (normally H3F to H4R) ↓ changed to H3F - H4R

Processor card R, etched: JPRN G3 (normally G3G to G3R)

now +9ns OK



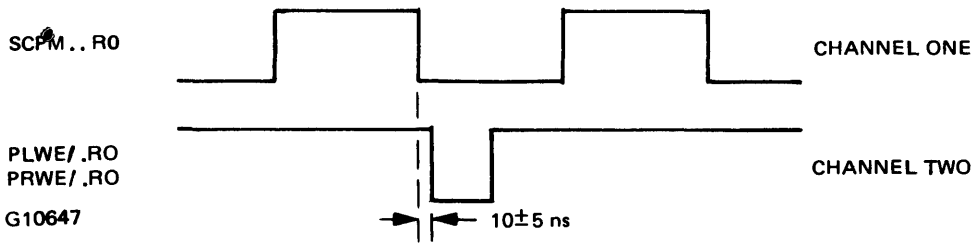


Figure 4-15. B 1720 Scratchpad Write Enable Timing

Step 12: Scratchpad Write Enable Pulse Width

Figure 4-16 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the Scratchpad Left Write Enable and Scratchpad Right Write Enable pulse widths. Adjustment of the pulse widths of these nets also adjusts the negative transition of Scratchpad Address lines SPA0..R0, SPA1..R0, SPA2..R0, and SPA3..R0.

Oscilloscope Set-up

Trigger	External positive
Channel One	SCPM..R0, processor backplane XE0W
Channel Two	PLWE/.R0, processor backplane YE1F PRWE/.RO, processor backplane YE1S
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Net MNTR0.P0, processor backplane XB0H
Machine State	Scratchpad Write Routine (table 4-2) executing

Measurements (All at 1-volt level; all system voltages nominal.)

The width of Scratchpad Write Enable (PLWE/.R0 and PRWE/.R0) must be greater than 30 ns.

Adjustments

Processor card R, wire-wrapped: DL5N H3 (normally H3M to H4B)

Processor card R, etched: JPRN G3 (normally G3C to G3S)

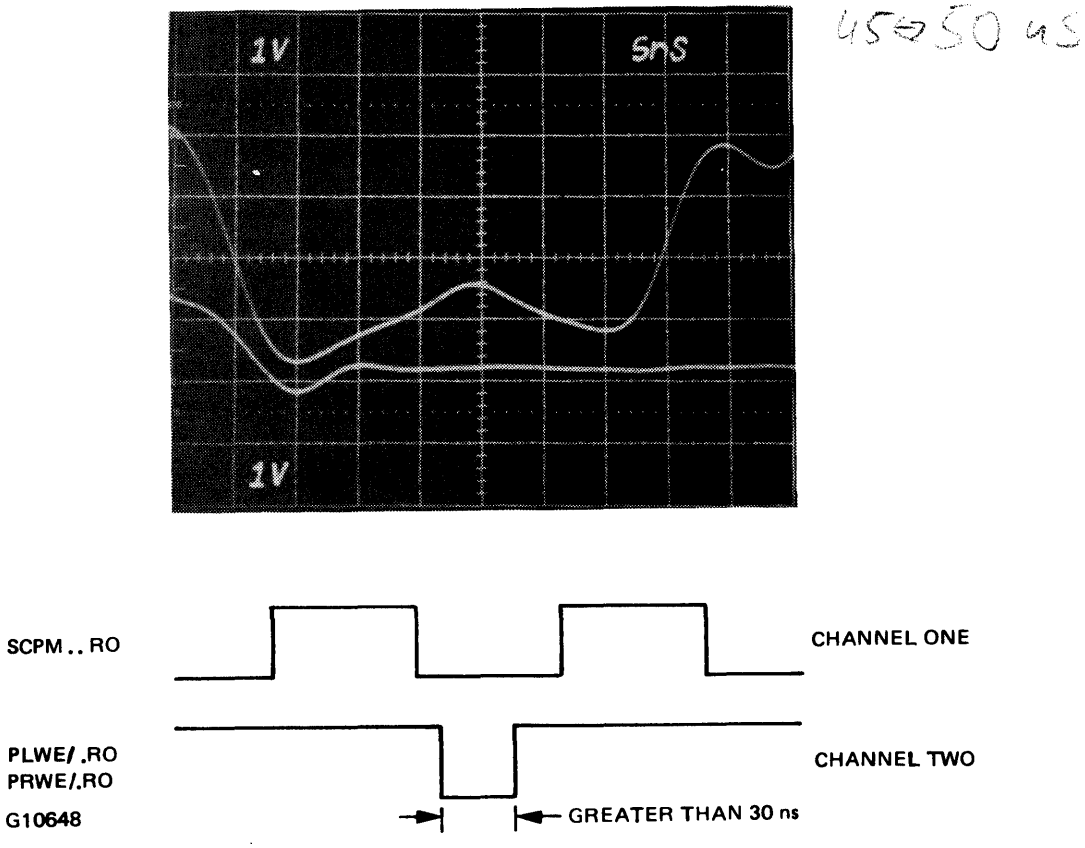


Figure 4-16. B 1720 Scratchpad Write Enable Pulse Width

Step 13: Scratchpad Address Line Alignment

Figure 4-17 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the four Scratchpad Address lines (SPA0..R0, SPA1..R0, SPA2..R0, and SPA3..R0) referenced to the trailing edge of SCPM..R0. The address line leading edge adjustment is included.

Oscilloscope Set-up

Trigger	External positive
Channel One	SCPM..R0, processor backplane XE0W
Channel Two	SPA0..R0, processor backplane YE1N
	SPA1..R0, processor backplane YE0F
	SPA2..R0, processor backplane YE1C
	SPA3..R0, processor backplane YE1A
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Net MNTR0.P0, processor backplane XB0H
Machine State	Scratchpad Write Routine (table 4-2) executing

Measurements (All at 1-volt level; all system voltages nominal.)

The leading edges of Scratchpad Address Lines must occur 7 ±7 ns before the trailing edge of SCPM.R0.

Adjustments

Processor card R, wire-wrapped: DL5N G3 (normally G3M to I4F)

Processor card R, etched: JPRN G2 (normally G2C to G2R)

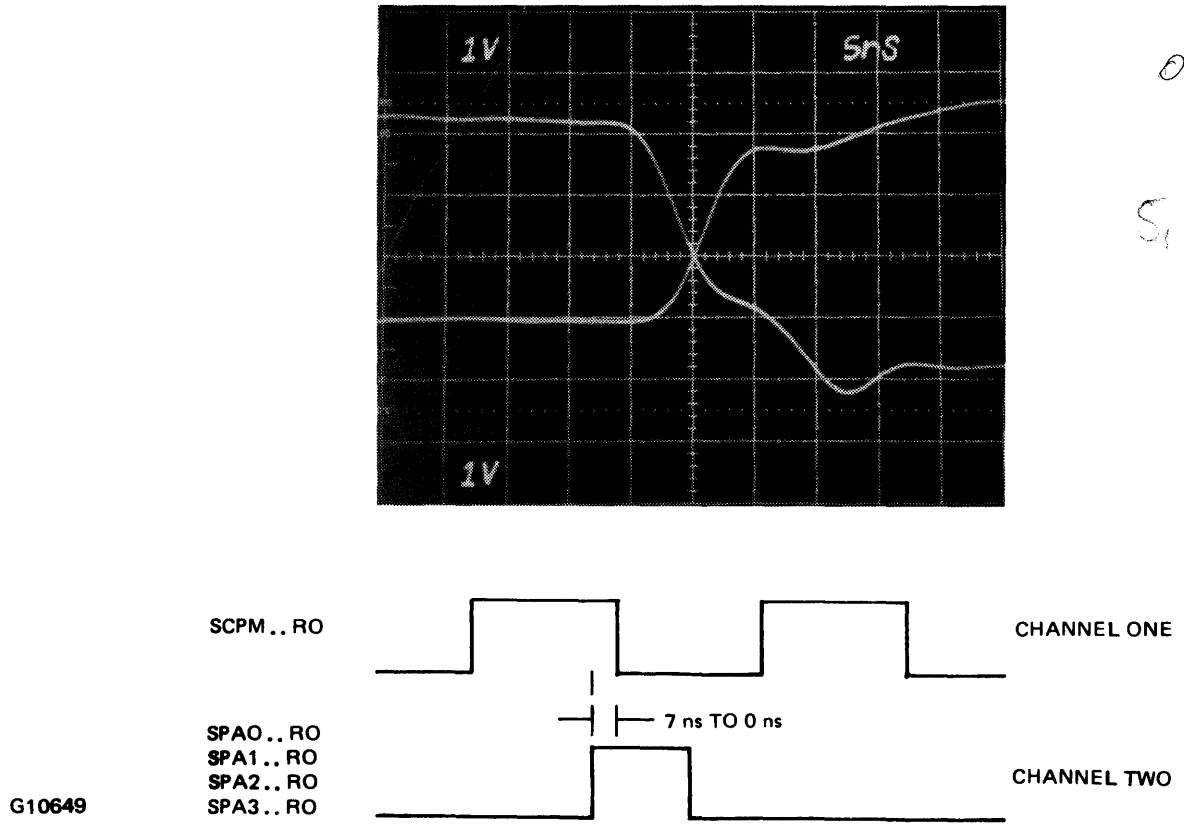


Figure 4-17. B 1720 Scratchpad Address Line Alignment

Step 14: Processor Clock E Delayed Clock

Figure 4-18 shows the oscilloscope set-up and timing requirements for processor card E Delayed Clock (DLYCLKE0) referenced to SCPM..R0. The Delayed Clock is used to clock the FA register and Scratchpad input latches. This is a verification step only; no adjustments are included.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..R0, processor backplane XE0W
Channel Two	DLYCLKE0, processor backplane XB7F
Horizontal	0.05 us/cm, X10 (5 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edge of DLYCLKE0 must occur 30 ±3 ns after the trailing edge of SCPM..R0.

Adjustments

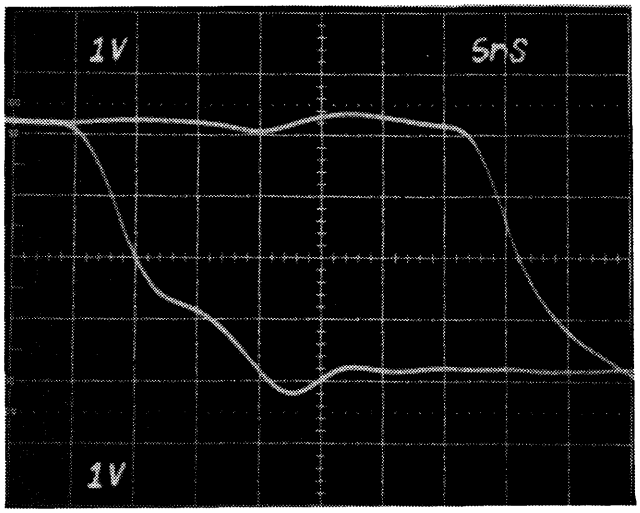
was not ok. 39ns

No adjustments. For reference only.

Was W

Processor card E, wire-wrapped: DL2N D9 (wired D9B to B7S) changed to D9E → B7S

Processor card E, etched: JPRN D9 (wired D9C to D9P)



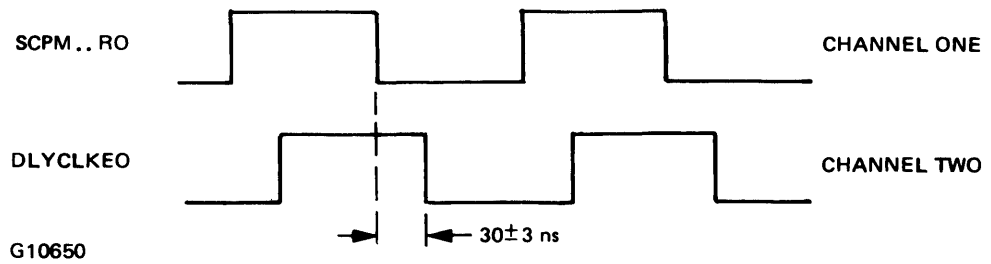


Figure 4-18. B 1720 Processor Card E Delayed Clock Timing

Step 15: A-Stack Write Enable

Figure 4-19 shows the oscilloscope set-up and timing requirements for A-Stack Write Enable (1ASWE/E1 and 2ASWE/E1) referenced to SCPM..R0. 1ASWE/E1 and 2ASWE/E1 are checked for negative-going edge alignment and pulse width. No adjustment is necessary.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	SCPM..R0, processor backplane XE0W
Channel Two	1ASWE/E1, processor card E frontplane $AY
	2ASWE/E1, processor card E frontplane $BY
Horizontal	2 us/cm, X10 (20 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	A-Stack Write Routine (table 4-3) executing

Measurements (All at 1-volt level; all system voltages nominal.)

The write Enable negative-going edge must occur 20 ±3 ns after the trailing edge of SCPM..R0.

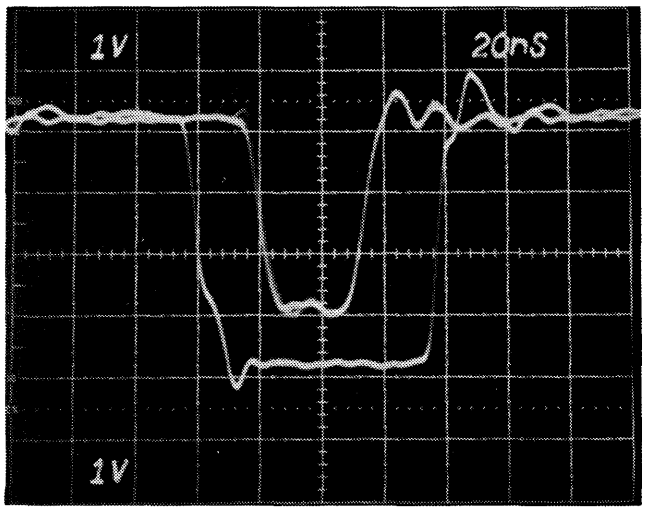
The write Enable negative-going pulse width must be 40 ±8 ns.

Adjustments

No adjustments. For reference only.

Processor card E, wire-wrapped: DL5N C9 (C9F to A8R and B9B give the positive-going edge, C9B to B9A gives the negative-going edge)

Processor card E, etched: JPRN D8 (D8J to D8L gives the positive-going edge; D8N to D8P gives the negative-going edge) Figure 4-20 provides comprehensive timing for the B 1720 Scratchpad and A-Stack.



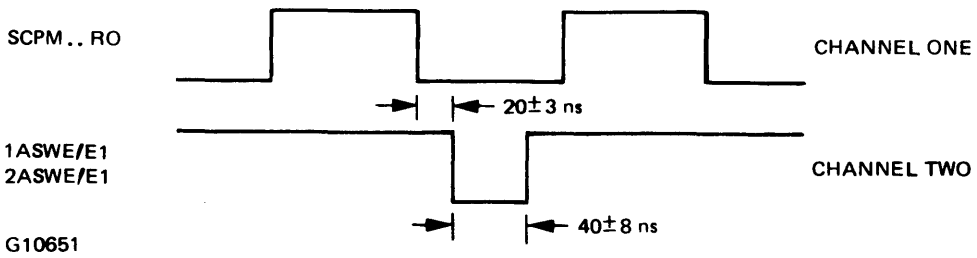


Figure 4-19. B 1720 A-Stack Write Enable Timing

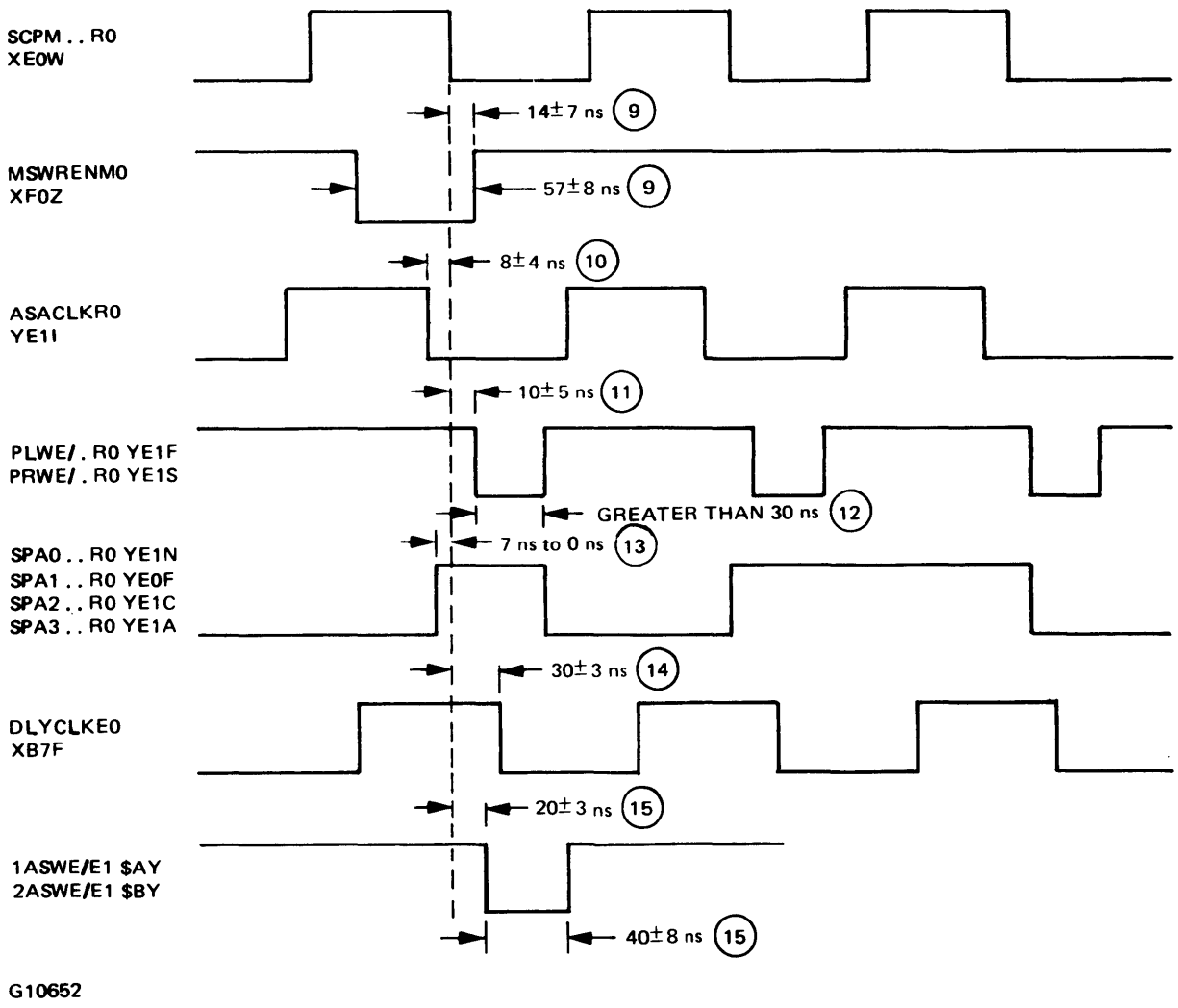


Figure 4-20. Comprehensive Timings, B 1720 Scratchpad and A-Stack

MEMORY BASE ALIGNMENT

Steps 16 to 23, keyed to figures 4-23 to 4-31, provide information for checking and adjusting S-memory in the B 1720 Model 1. These steps must be performed in the sequence listed because some of the adjustments affect later ones. When an adjustment is performed, each remaining step must be checked. Figure 4-32 provides comprehensive memory timing information.

Steps 16 through 20 must be completed for each memory base before steps 21, 22, and 23 are started. All measurements are made at the 1-volt level, with system voltages set to normal.

Figures 4-24 through 4-31 are keyed to steps 16 through 32. Figure 4-32 provides comprehensive memory timings.

Initial Set-up

Jumper chip G2 is wired according to the number of memory base units in the system. (See figure 4-21.)

Jumper chip H1 wiring determines the MBU group number. For memory size less than 256K bytes, H1 is wired pin S to pin D. For a 256K-byte (maximum) memory, H1 is wired pin S to pin H to pin D.

Alignment

This memory timing procedure requires that refresh be disabled and that the processor be looping in a Memory Write Routine (table 4-4). Figure 4-22, which shows the FA register breakdown, provides information needed to select the desired memory base.

Refresh Disable

Refresh is disabled by forcing port interchange backplane pin XA0R true. This assures a clean oscilloscope trace.

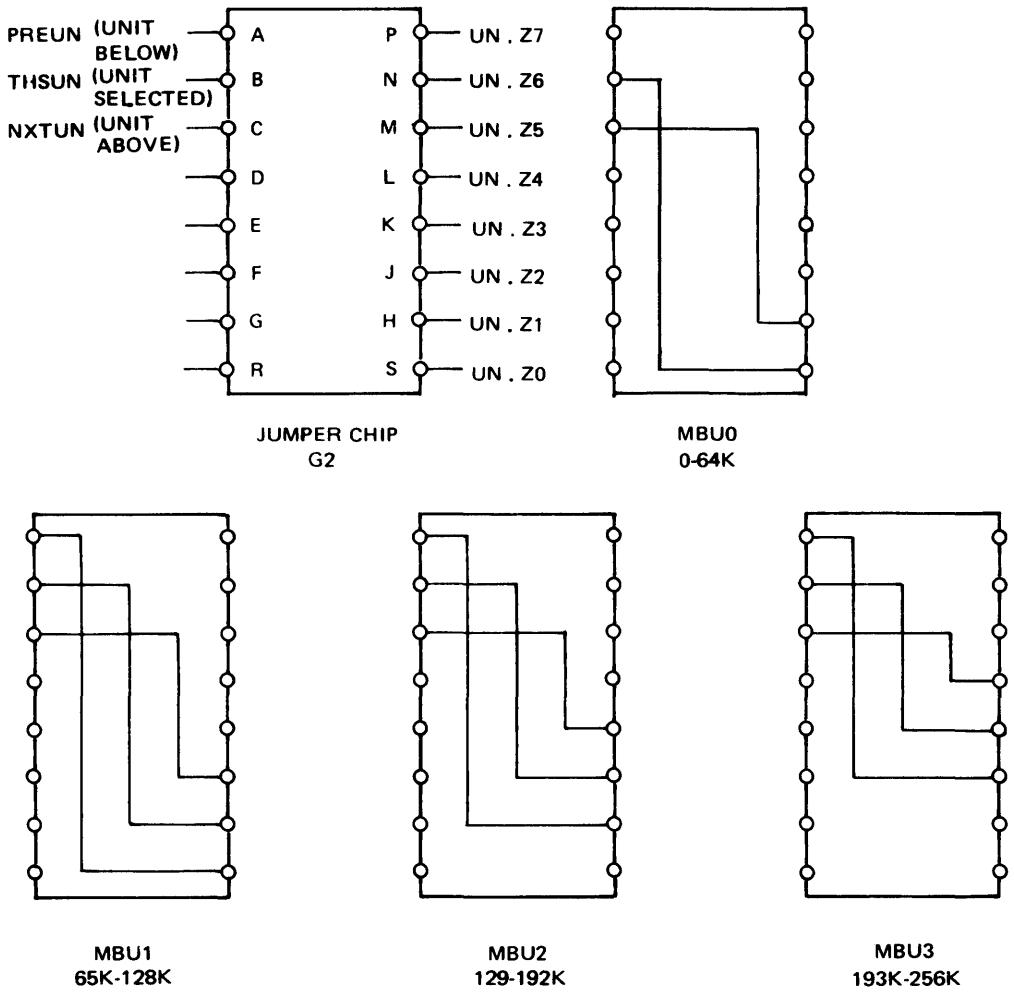


Figure 4-21. Chip G2 Connections for 1 to 4 MBUs

Table 4-4. Memory Write Routine

A Register	MSM Register	Description
000000	7819	Write from X, DFL 25
000001	8400	Lit zeroes to A register

Load the following registers:

TOPM=000008 CP=000018 CD=000007 X=000000 FA=nnnnnn

nnnnnn = any valid address in the desired MBU. (Figure 4-22)

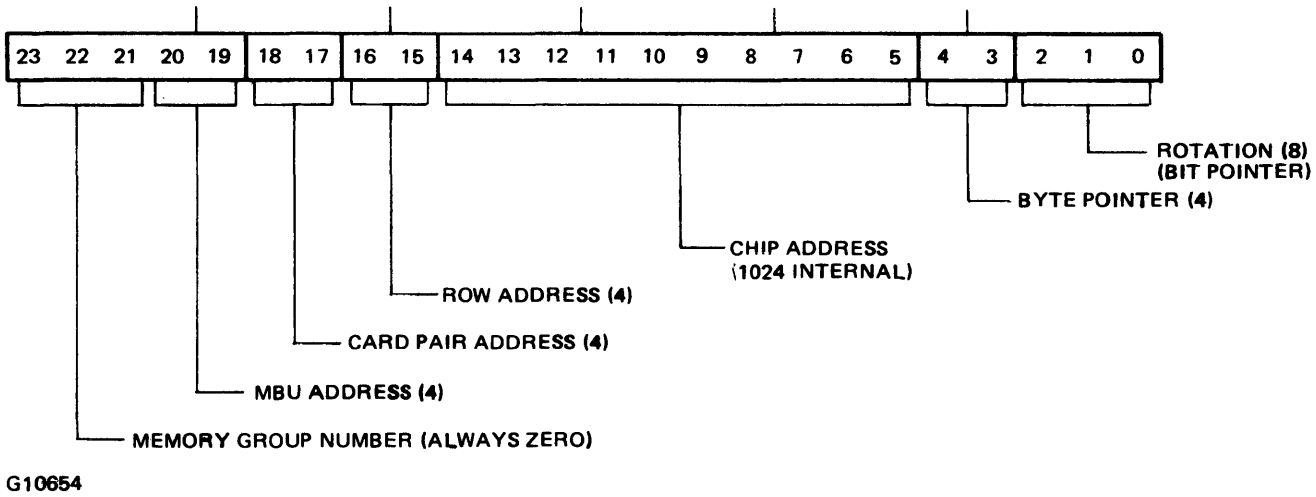


Figure 4-22. FA Register Breakdown

Step 16: Preliminary Precharge Adjustment

Figure 4-23 shows the oscilloscope set-up, adjustment procedures, and timing requirements for Precharge referenced to Memory Address Align (MAALGN). This step ensures that the Precharge signal is at its nominal value before other signals that use Precharge as a reference are set up. This is particularly important when new memory bases are to be installed.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	MAALGN.0, MBU backplane XB6Y
Channel Two	PRE.01.0, MBU backplane YB4S
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Memory Write Routine (table 4-4) executing

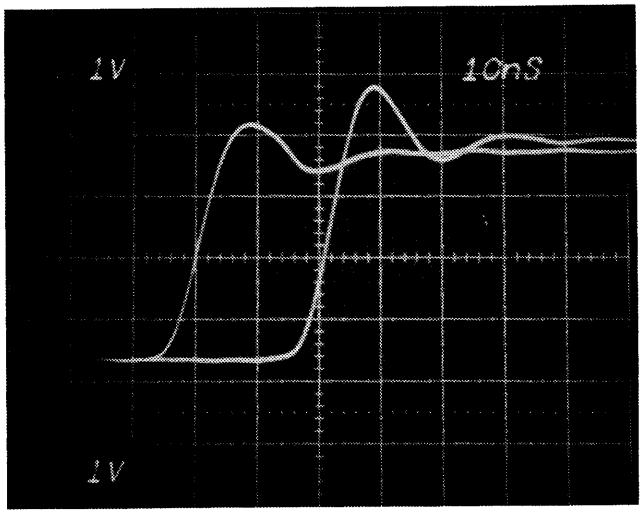
Measurements (All at 1-volt level; all system voltages nominal.)

The leading edge of Precharge must occur 20 ±5 ns after the leading edge of MAALGN.

Adjustments

IC card coarse (>10 ns): DLCN G8 (normally G8F to G7H)

IC card fine (<10 ns): DL2N H7 (normally H7F to K5F)



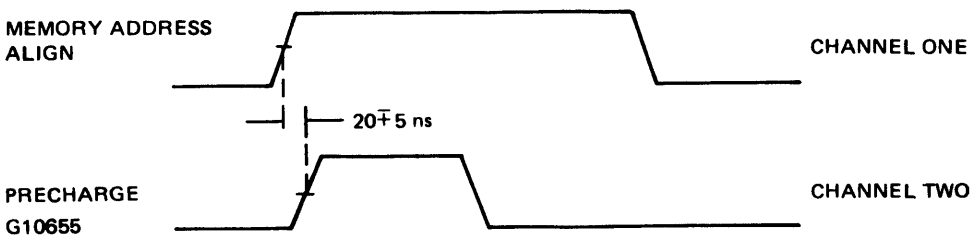


Figure 4-23. B 1720 Preliminary Precharge Timing

Step 17: Chip Enable Leading Edge

Figure 4-24 shows the oscilloscope set-up adjustment procedures, and timing requirements for Chip Enable referenced to Precharge. This adjustment affects both Read and Write Chip Enable.

Oscilloscope Set-up

Trigger	External positive
Channel One	PRE.01.0, MBU backplane YB4S
Channel Two	CN.01..0, MBU backplane YB4T
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	PRE.01.0, MBU backplane YB4S
Machine State	Memory Write Routine (table 4-4) executing

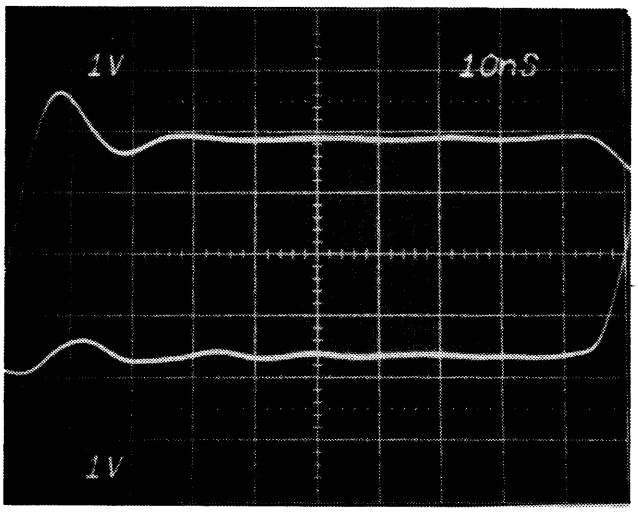
Measurements (All at 1-volt level; all system voltages nominal.)

The leading edge of Chip Enable must occur 101 ±5 ns after the leading edge of Precharge.

Adjustments

IC card coarse (>10 ns): DL2N I4 (normally H3C to I3H)

IC card fine (<10 ns): DL2N I4 (normally I4K to D0R)



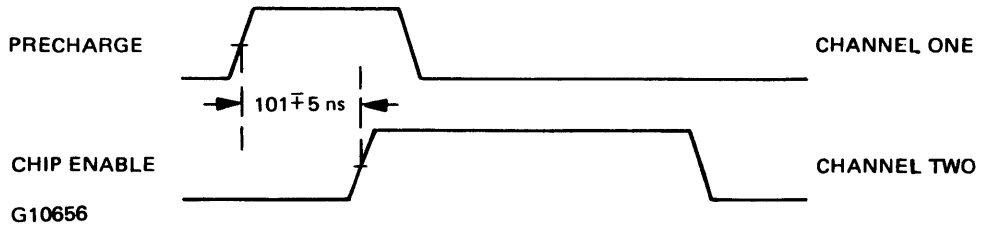


Figure 4-24. B 1720 Chip Enable Leading Edge Timing

Step 18: Chip Enable Trailing Edge, Write

Figure 4-25 shows the oscilloscope set-up, adjustment procedures, and timing requirements for Chip Enable, Write referenced to the Memory Clock. A Chip Enable write pulse is approximately 5 clock periods wide and a read pulse is approximately 3 clock periods wide.

Oscilloscope Set-up

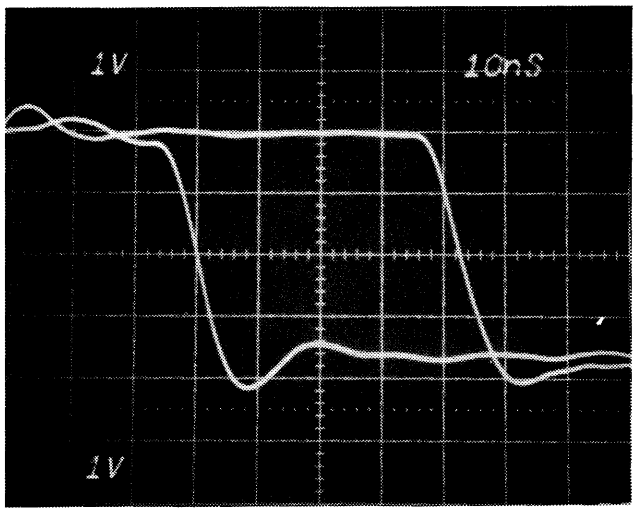
Trigger	External positive
Channel One	MEMCLK.0, MBU backplane XB3W
Channel Two	CN.01..0, MBU backplane YB4T
Horizontal	0.1 us/cm, X10 (1 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	PRE.01.0, MBU backplane YB4S
Machine State	Memory Write Routine (table 4-4) executing

Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edge of Chip Enable Write must occur 47 ±5 ns before the trailing edge of Memory Clock.

Adjustments

IC card: DLCN H6 (normally H6C to J7F)



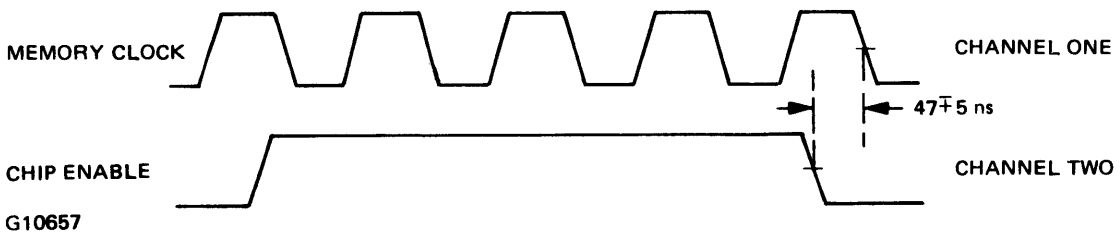


Figure 4-25. B 1720 Chip Enable Trailing Edge Write Timing

Step 19: Write Enable Trailing Edge

Figure 4-26 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the trailing edge of Write Enable referenced to the Memory Clock.

Oscilloscope Set-up

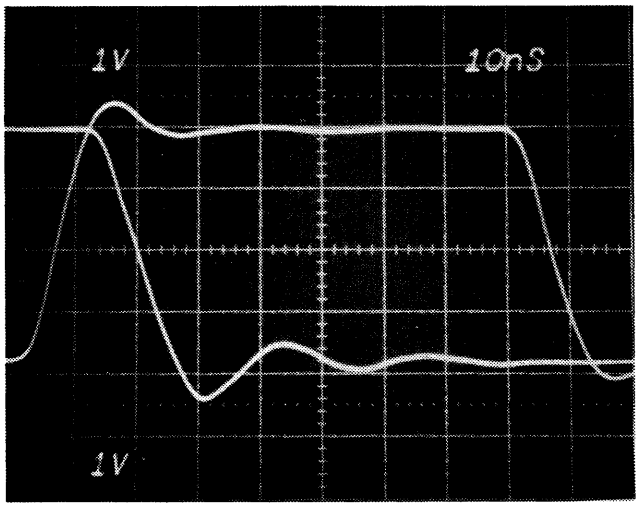
Trigger	External positive
Channel One	MEMCLK.0, MBU backplane XB3W
Channel Two	WE00...0, MBU backplane YB4P
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	PRE.01.0, MBU backplane YB4S
Machine State	Memory Write Routine (table 4-4) executing

Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edge of Write Enable must occur 70 ±5 ns before Memory Clock.

Adjustments

IC card DL5N H6 (normally H6C to K5C)



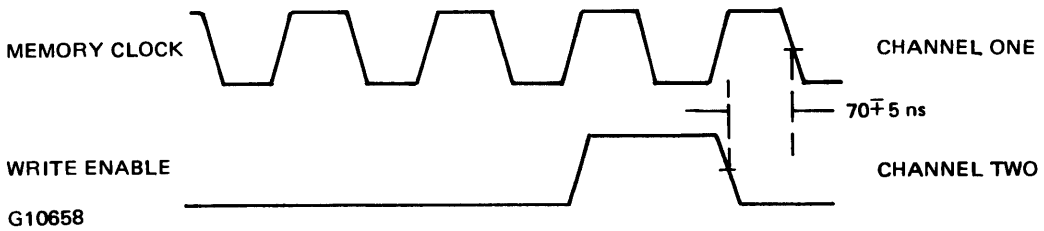


Figure 4-26. B 1720 Write Enable Timing

Step 20A: Read Data Alignment Signal

For an accurate reading, measurement of RDALGN must be done in two steps. Figure 4-27 shows the oscilloscope set-up and timing requirements for the leading edge of Precharge referenced to the trailing edge of Memory Clock. The second step is the measurement of the trailing edge of Memory Clock referenced to the leading edge of RDALGN. Because these signals occur more than one clock period apart, an additional 167 nanoseconds (1 clock period) must be added. This can be seen as A+B+167 in figure 4-32.

Oscilloscope Set-up

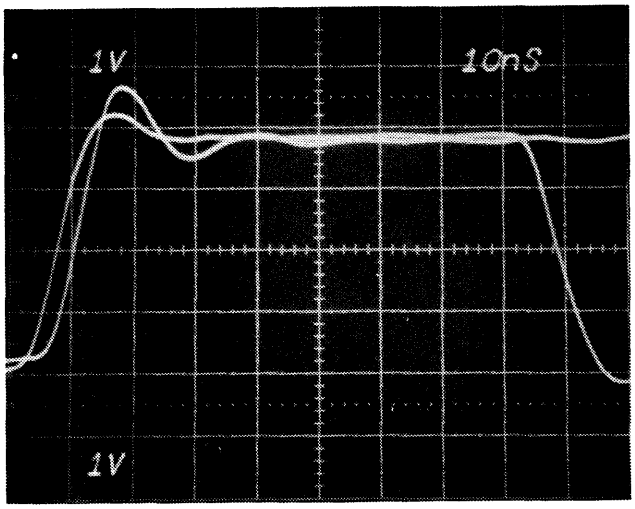
Trigger	External positive
Channel One	MEMCLK.0, MBU backplane XB3W
Channel Two	PRE.01.0, MBU backplane YB4S
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	PRE.01.0, MBU backplane YB4S
Machine State	Memory Write Routine (table 4-4) executing

Measurements (All at 1-volt level; all system voltages nominal.)

Measure and record time between leading edge of Precharge and trailing edge of Memory Clock.

Adjustments

None (go to step 20B).



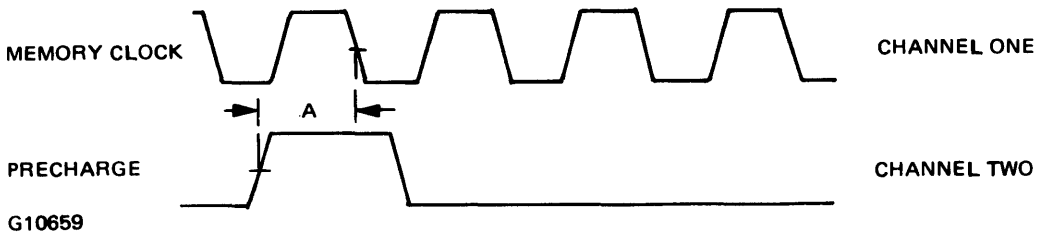


Figure 4-27. B 1720 Read Data Alignment, Part 1

Step 20B: Read Data Alignment Signal

Figure 4-28 shows the oscilloscope set-up, adjustment procedures, and timing requirements for the leading edge of RDALGN referenced to the trailing edge of Memory Clock.

Oscilloscope Set-up

Trigger	External positive
Channel One	MEMCLK.0, MBU backplane XB3W
Channel Two	RDALGN.0, MBU backplane XB3Q
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	PRE.01.0, MBU backplane YB4S
Machine State	Memory Write Routine (table 4-4) executing

Measurements (All at 1-volt level; all system voltages nominal.)

Measure and record time between trailing edge of Memory Clock and leading edge of RDALGN. Add this time to the reading obtained in Step 20A and add 167 ns; the total should be 315 ±5 ns.

Adjustments

IC card coarse (>10 ns): DLCN H5 (normally H5K to I7P)

IC card fine (<10 ns): DL2N I6 (normally I6E to I7H)

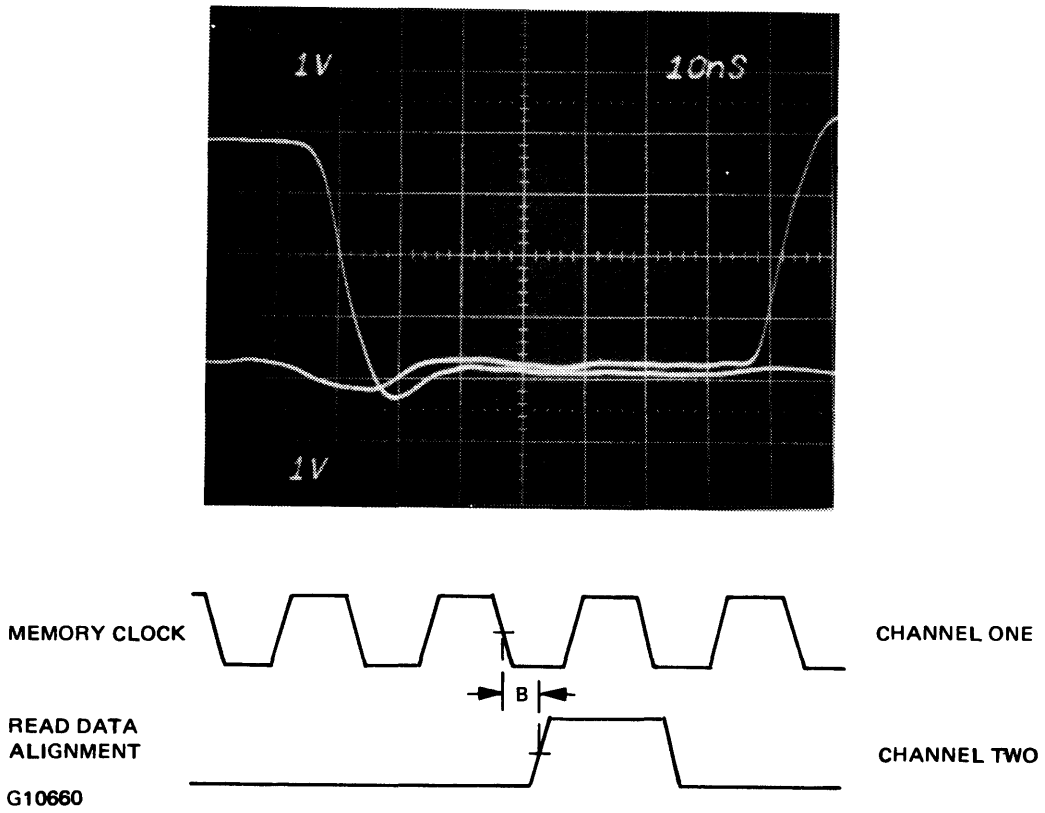


Figure 4-28. B 1720 Read Data Alignment, Part 2

Step 21: Memory Clock Alignment

Figure 4-28 shows the oscilloscope set-up, alignment procedures, and timing requirements for the Memory Clock referenced to the System Reference Clock. Memory Clock is measured at the first memory base (MBU-0). The adjustment is to the clock module and affects the Adjustable Clock output.

Oscilloscope Set-up

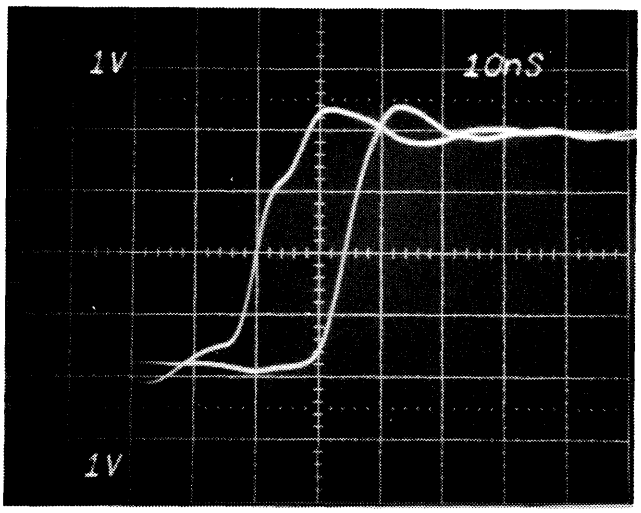
Trigger	Internal positive
Channel One	SCPM..B0, port interchange backplane XA3W
Channel Two	MEMCLK.0, MBU backplane XB3W
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Idle

Measurements (All at 1-volt level; all system voltages nominal.)

The trailing edge of Memory Base Clock must occur 7 ±10 ns after the trailing edge of System Clock.

Adjustments

Clock module assembly: JPRN E3 (normally E3K to E3L).



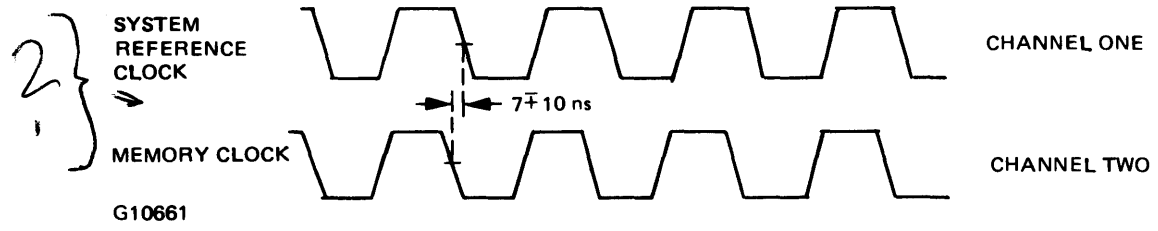


Figure 4-29. B 1720 Memory Clock Timing

Step 22: Precharge Alignment, End MBU

Figure 4-30 shows the oscilloscope set-up, adjustment procedures, and timing requirements for Precharge to Memory Address Align (MAALGN) in the end memory base unit. (For a single-MBU system, perform this step, but omit step 23).

Oscilloscope Set-up

Trigger	Internal positive
Channel One	MAALGN.0, MBU backplane XB6Y (end unit)
Channel Two	PRE.01.0, MBU backplane YB4S (end unit)
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Memory Write Routine (table 4-4) executing

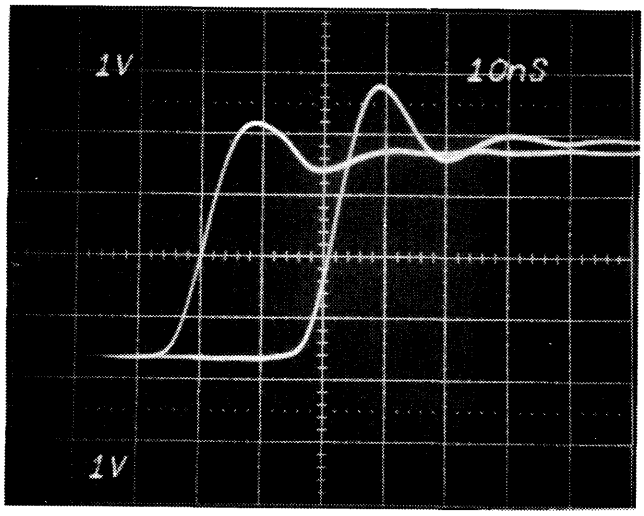
Measurements (All at 1-volt level; all system voltages nominal.)

The leading edge of Precharge must occur 20 ±5 ns after the leading edge of MAALGN.

Adjustments

IC card coarse (>10 ns): DLCN G8 (normally G8F to G7H)

IC card fine (<10 ns): DL2N H7 (normally H7F to K5F)



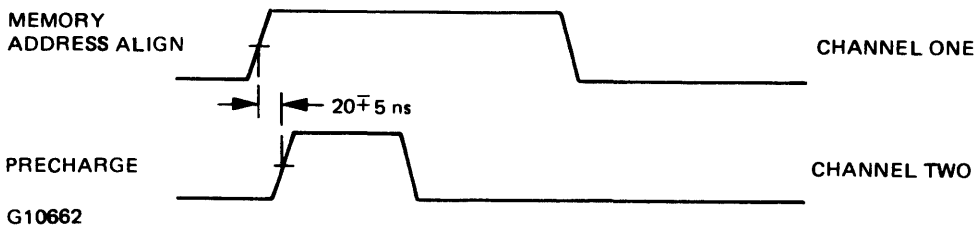


Figure 4-30. B 1720 Precharge Alignment, End MBU

Step 23: Precharge Alignment, Interposed MBU

Step 22 aligns the last MBU; if it is the only MBU, step 23 is unnecessary. If more than one MBU is installed, step 23 is done for every interposed MBU after step 22 has been done for the last one. Figure 4-31 shows the oscilloscope set-up, adjustment procedures, and timing requirements between the Read Strobe (RSTR.3A1) and Read Data Align (RDALN.1) at the port interchange. The adjustment of Precharge allows data from the interposed MBU(s) to be aligned with the end MBU. Figure 4-32 provides a comprehensive memory timing example, Figure 4-33 provides a comprehensive memory timing example with interposed MBU's.

Oscilloscope Set-up

Trigger	Internal positive
Channel One	RSTR.3A1, port interchange card A $NX
Channel Two	RDALGN.1, port interchange card A $MX
Horizontal	0.1 us/cm, X10 (10 ns/cm)
Vertical	0.1 V/cm with X10 probe (1 V/cm)
Display Mode	Alternate
Trigger Source	Channel One
Machine State	Memory Write Routine (table 4-4) executing

Measurements (All at 1-volt level; all system voltages nominal)

Load FA to an address in the end memory unit. Measure and record time between trailing edge of RST.3A1 and trailing edge of RDALGN.1. Change FA address to select an interposed MBU. RDALGN.1 must occur 5 ±10 ns before RDALGN.1 of the end MBU.

Adjustments

IC card coarse (>10 ns): DLCN G8 (normally G8F to G7H)

IC card fine (<10 ns): DL2N H7 (normally H7F to K5F)

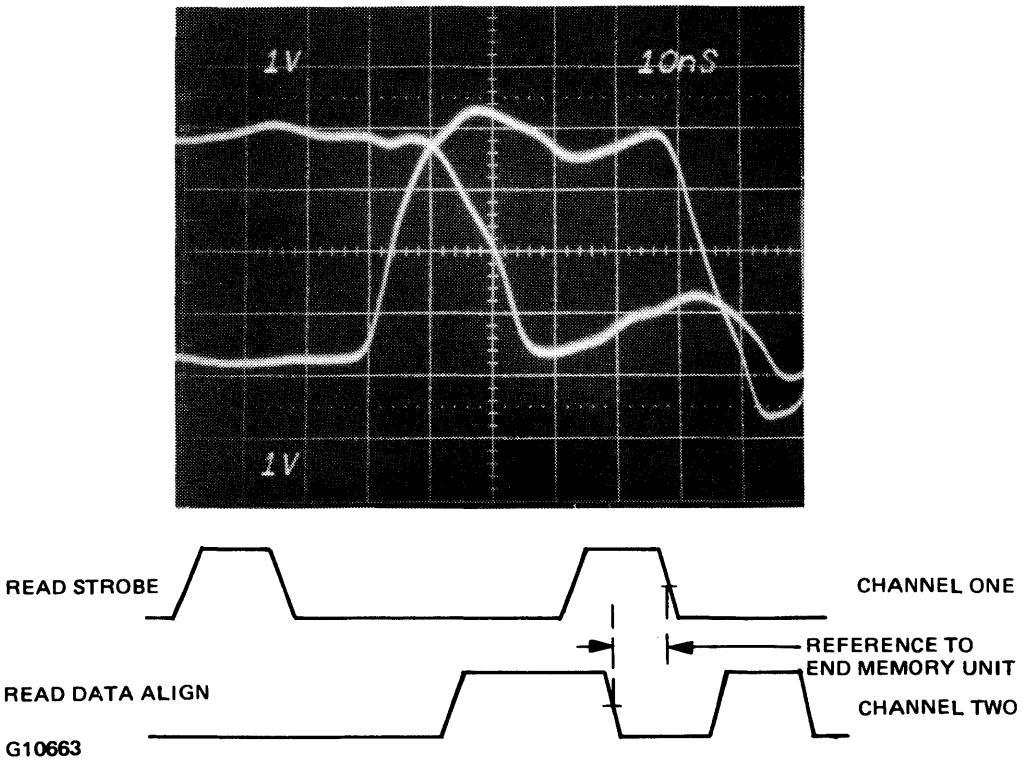


Figure 4-31. B 1720 Precharge Alignment, Interposed MBU(s)

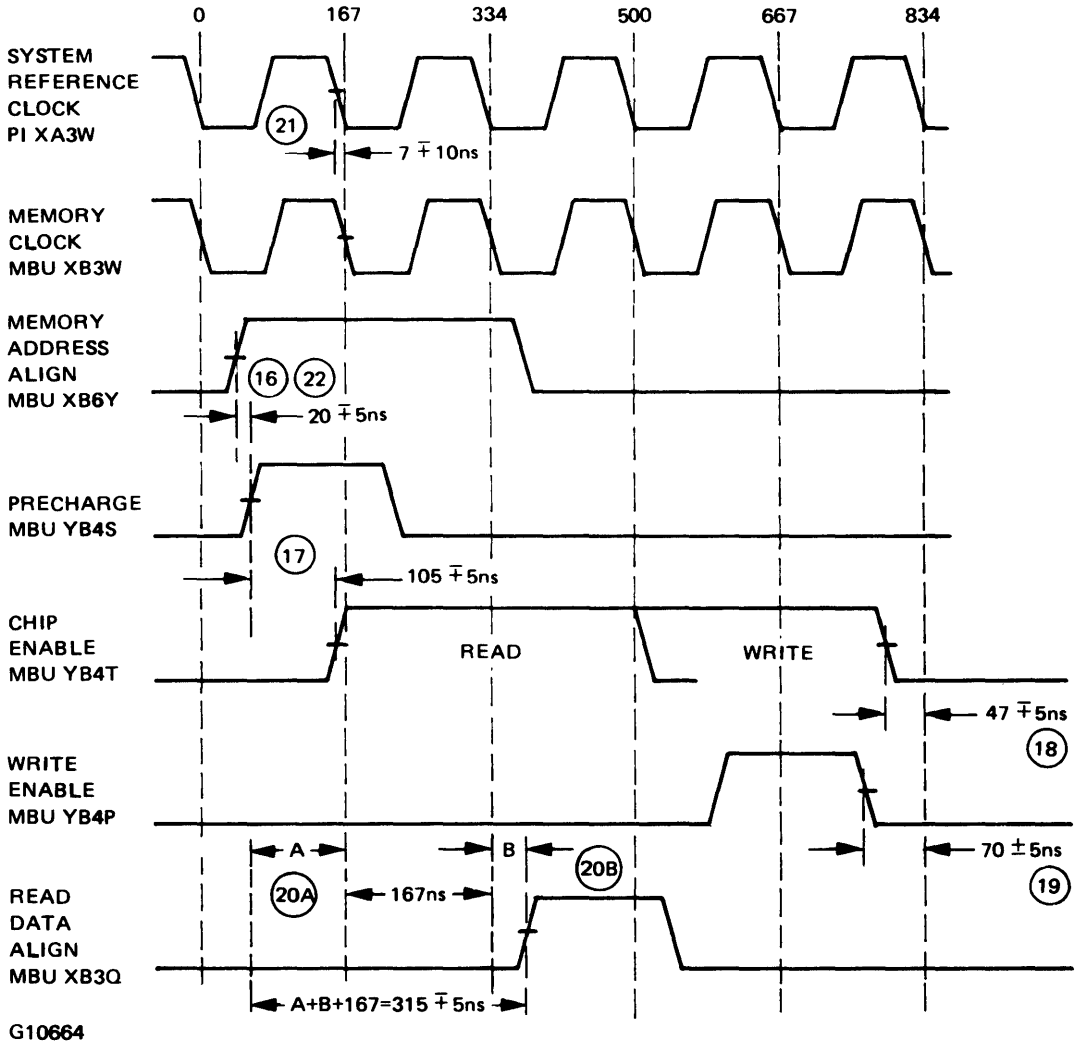


Figure 4-32. B 1720 Comprehensive Memory Timings

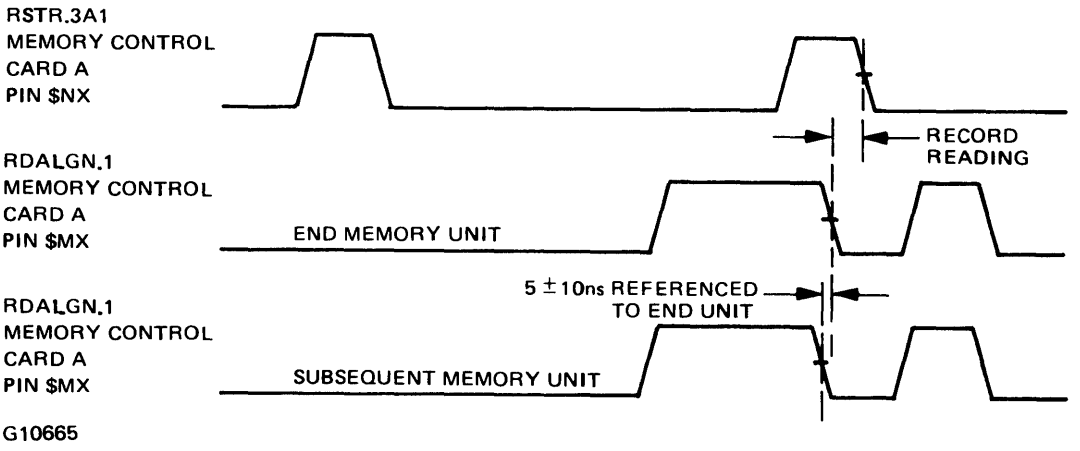


Figure 4-33. Comprehensive Memory Timings, Interposed MBU

MEMORY AND LOGIC POWER SUPPLY ADJUSTMENTS

Refer to Þ 1870/B 1860/B 1700 Power Supplies, FETM 1070281.

page 5-7 in PM PLAN

Although these means may be applied in any combination or sequence which is appropriate to a problem, it is recommended that the following approach be considered:

Regardless of the method used to solve a problem, it is found that troubleshooting proceeds more directly to a solution if a definite set of objectives, or steps are followed. These general steps are described as follows and are shown in figure 5-1.

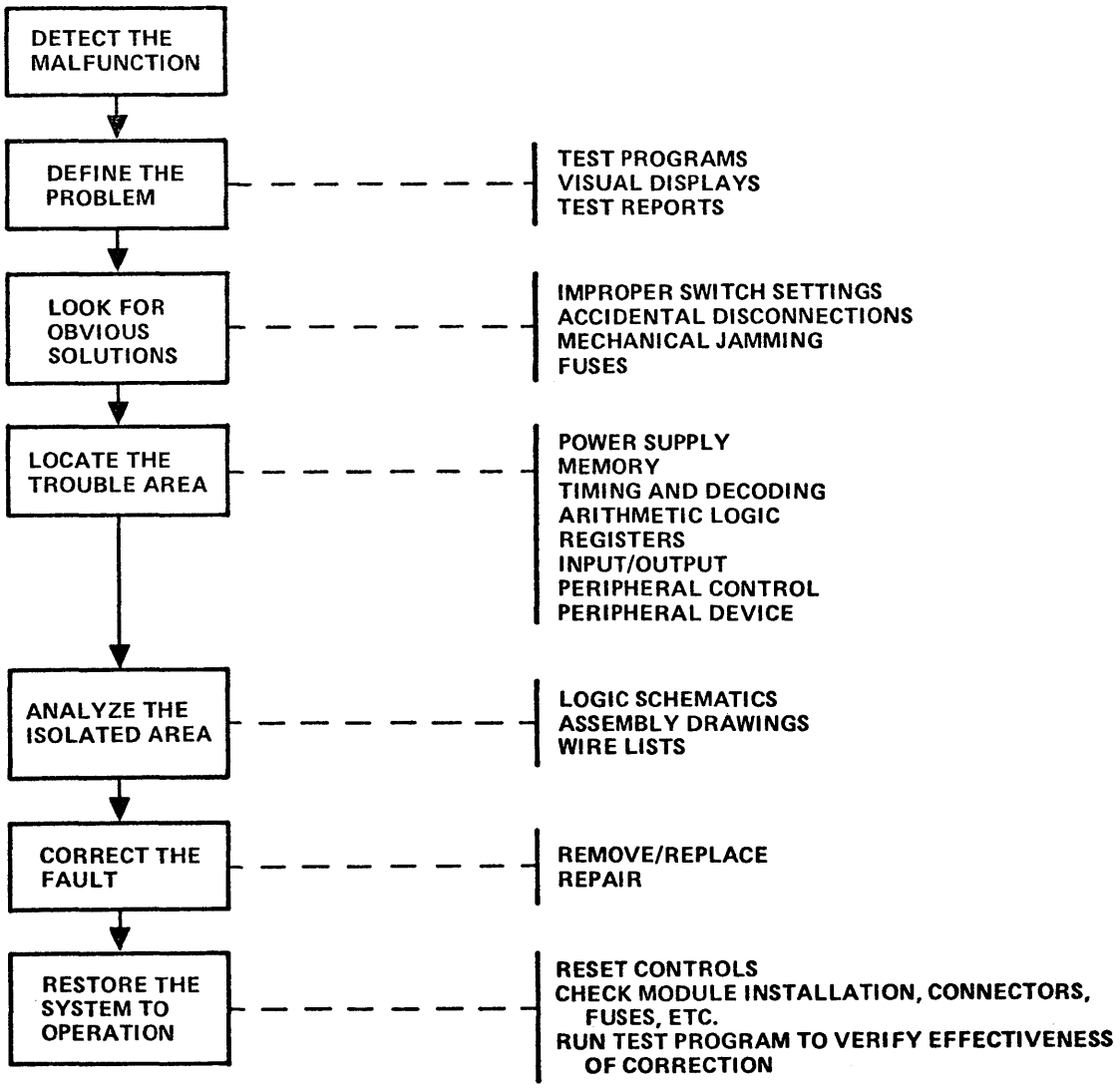


Figure 5-1. Troubleshooting Flow (General)

a. Define the problem. This is the most important single requirement, and is likely to be the most time-consuming. Determine exactly what the system is failing to do, or is doing that it should not. If, for example, the 1C micro (register move) is not performed correctly, check the operation of both the source and sink registers, the intermediate data path, and the logic which decodes and implements execution of the micro. Check also to ensure that the micro is reaching the decoding logic properly.

b. Look for obvious solutions. Make sure that a malfunction has actually occurred. Relate problems to recent events such as cleaning, servicing, or the installation of new or replacement components. Look for improperly set controls, accidental disconnections of cables, and other mechanically caused circuit discontinuities. Consider miscellaneous temporary failures, such as mechanical jamming of peripheral equipment.

c. Locate the trouble area. Isolate the fault to a functional area of the system by a process of elimination.

d. Analyze the isolated area. Use the logic schematics and appropriate test equipment to identify the faulty circuit element.

e. Correct the fault. This is accomplished through repair or replacement. Attempt to determine the cause of the failure and take the necessary steps to prevent recurrence.

f. Restore the system to operation. Return all components, cables, and subassemblies which have been removed to their proper locations. Correctly set all controls, then verify the repair work by running appropriate test programs.

A specialized troubleshooting flow for the memory power supply (only) is shown in figure 5-2.

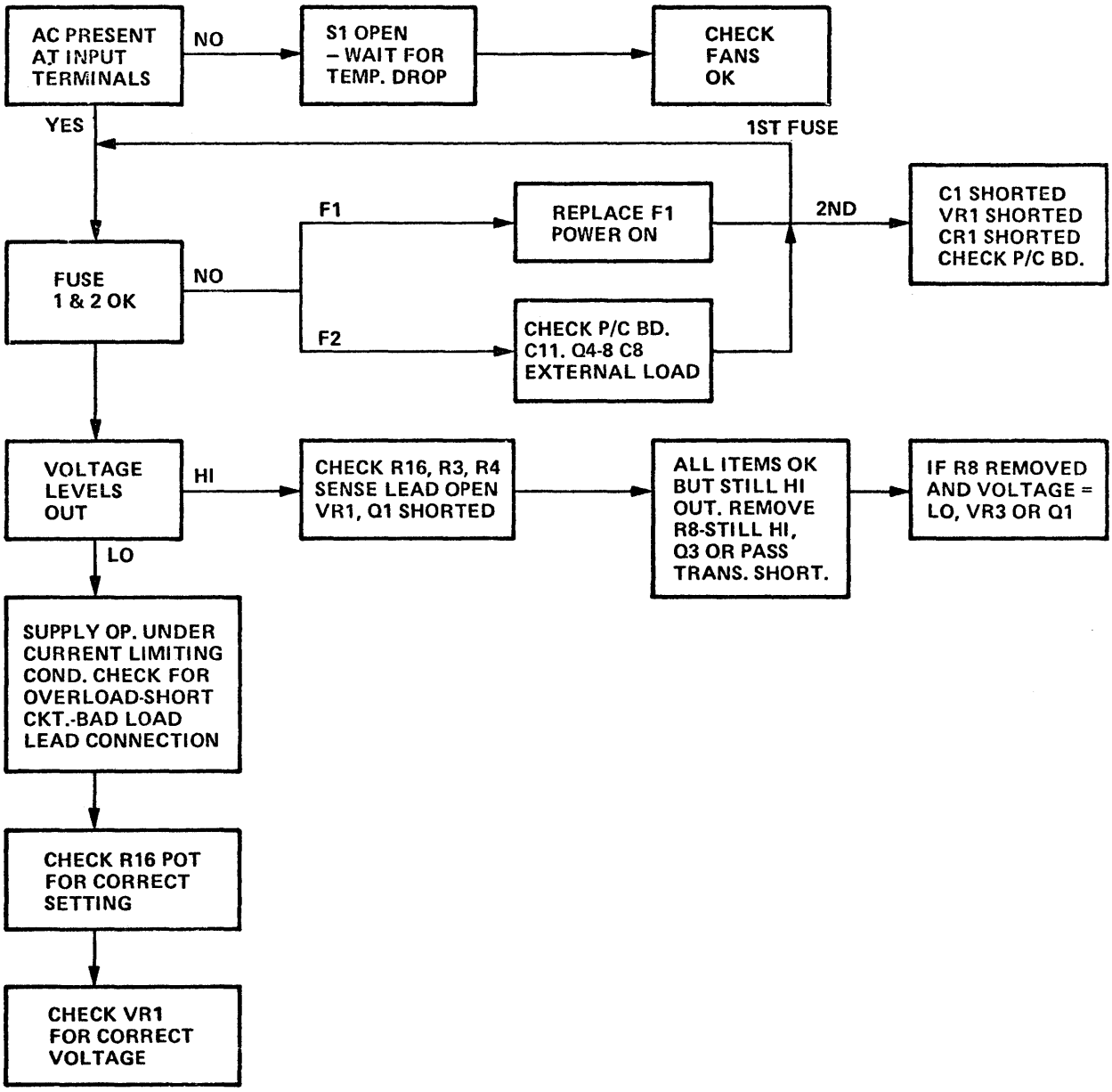


Figure 5-2. Memory Power Supply Troubleshooting Flow

Locating Intermittent Logic Problems

By their very nature, intermittent logic problems are usually very difficult to locate. Field experience has shown that operation of a system at high or low voltage margins is often helpful in locating marginal faults, because such failures tend to occur more frequently under these conditions. The following voltage margins may be used, as appropriate, when intermittent logic failures are suspected. These settings apply to B 1720 Series Systems only.

Voltage	Nominal Setting	Margin High	Margin Low
+4.75	+4.85 ±0.010	+5.05	+4.65
-2.00	-2.08 ±0.010	-2.15	-2.02
+12.00	+12.00 ±0.10	+12.50	+11.50
-12.00	-12.00 ±0.10	-12.50	-11.50
+20.00	+18.75 ±0.05	+19.50	+18.00
+23.00	+3.60 ±0.010*	+3.85*	+3.35*
-5.00	-5.00 ±0.010	-5.25	-4.75

* +3.6 volt negative output is connected to the +18.75 volt terminal.

The system should always be set back to the nominal voltages after maintenance testing is completed.

SUBASSEMBLY REMOVAL AND REPLACEMENT

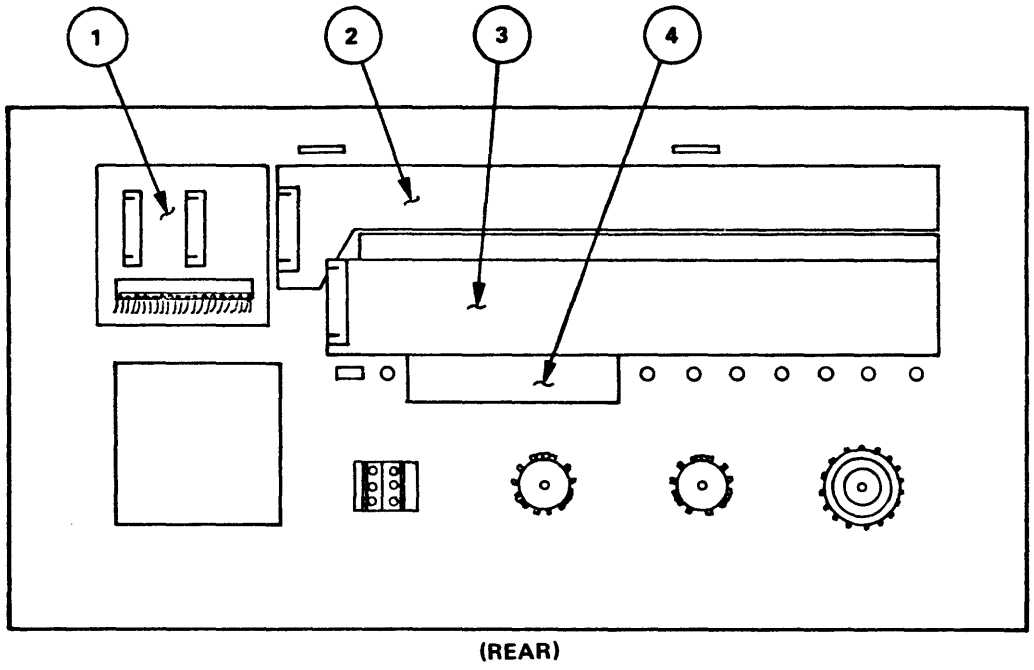
Performing maintenance and/or adjustment work on the various portions of the central system may require the removal of subassemblies to allow access to a desired area. Since such removal in some cases requires special procedures, instructions for the removal and replacement of all system subassemblies is provided in the following paragraphs.

Processor Console

The console contains all the controls necessary for operational control of the system. In hardware terms, these items are switches, lamps, buttons, and a cassette tape reader. The console is mounted to the cabinet frame by means of hinges attached at the right hand side, and may be swung open like a door. To open the console, exert a slight forward pressure (away from the cabinet) on the left side of the panel. This overcomes the force of magnetic latches which hold the console panel closed. With the console open, access may be gained to the rear of the hardware mounted thereon. Note that the console itself should never require removal from the cabinet (except in cases of physical damage), but opening is a necessary step of removing and/or replacing the subassemblies mounted on it. An assembly drawing of the console is shown in figure 5-3.

Console Indicator Lamps

The 24 console indicator lamps (light emitting diodes) are all mounted on a single printed circuit board. When replacement of a lamp becomes necessary, the circuit board must be removed from the console to allow access. The board itself is not attached to the console, but rather is held in position by the lamps, each of which is press-fitted into its respective mounting hole. Consequently, to remove the board a small amount of rearward pressure should be applied at a number of points around its perimeter. Once the board has been removed, the defective lamp may be unsoldered and replaced. Be sure to observe correct polarity when installing the replacement lamp. This may be accomplished by orienting the flat side of the lamp case toward the series resistor on the reverse side of the board.



1 CONNECTOR BOARD
2 LAMP BOARD
3 SWITCH BOARD
4 SMALL LAMP BOARD
5 24 CONSOLE LAMPS/SWITCHES
6 CONTROL SWITCHES AND INDICATORS
7 REGISTER GROUP SWITCH
8 REGISTER SELECT SWITCH
9 MODE SWITCH
10 POWER SWITCH
11 CASSETTE SWITCHES AND INDICATORS
12 CASSETTE DRIVE UNIT

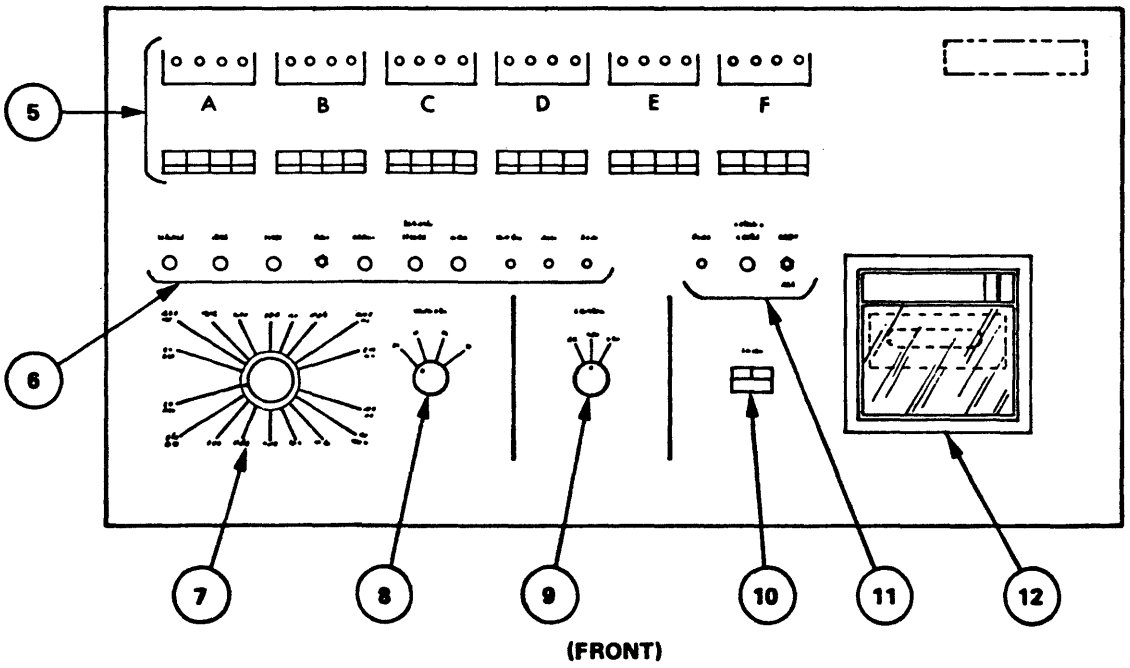


Figure 5-3. Control Panel Assembly

Console Switches

The 24 console switches corresponding to the 24 indicator lamps are mounted on a separate printed circuit board. Removal of the board is necessary if switch replacement is required. To replace a switch, perform the following steps:

a. Open the processor console.

b. Remove the console lamp board cable.

c. Remove the console switch board cable.

d. Remove the eight (8) retaining nuts which attach the switch board to the front panel.

e. Remove the metal retaining bars.

f. Remove the printed circuit board.

g. Unsolder the defective switch.

h. Install a new switch in the vacant position and solder in place.

i. Replace the console components in reverse order from which they were removed.

Cassette Tape Reader

The cassette tape reader is a device separate from the processor logic, having somewhat the same status as a peripheral. The major differences between the cassette and peripherals are that it is an input device only, and has direct access to the processor's main exchange (by way of the U-register), rather than going through the I/O subsystem. The cassette is described in detail in its own separate manual.

To remove the cassette tape reader from the console, proceed as follows:

a. Remove power from the cassette by placing the CASSETTE ON/OFF switch in the OFF position.

b. Make sure that the cassette tray (on the rear of the access door) is empty.

c. Disconnect the power cable attached to the top rear portion of the cassette reader chassis. Access to this area may be gained from the right side of the system cabinet.

d. Locate the two mounting latches which hold the cassette reader chassis in place. These are at the center of the left and right sides of the chassis, and may be adjusted from the front panel with a hex wrench (earlier models require a screwdriver). See figure 5-4.

e. Loosen the mounting latches and rotate them (about 1/4-turn) until the cassette chassis is free.

f. Remove the cassette reader by pulling directly outward from the processor console.

g. Replacement of the cassette reader may be accomplished by performing the above steps in reverse order.

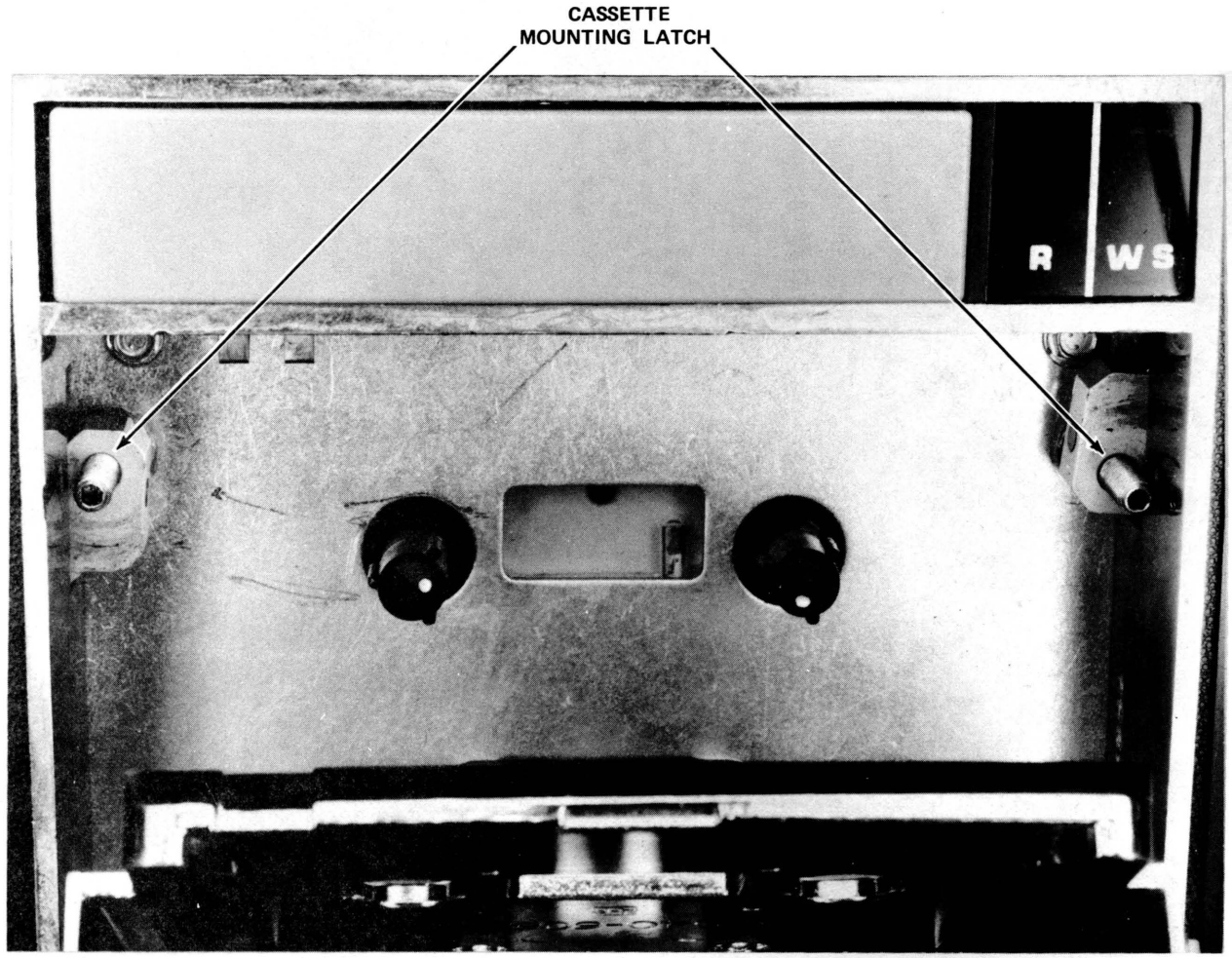


Figure 5-4. Cassette Mounting Latch Locations

Logic Cards and Cables

Each logic card and memory card is equipped with etched contacts which mate with matching receptacles mounted on the backplane. In addition, many cards are also equipped with front-plane receptacles into which logic cables (having matching plugs) are inserted. These connections are fragile, and must be handled carefully to avoid damage. Insertion and removal instructions for the cards and cables are as follows.

Removal of Logic Cards and Cables

Cables may be removed by moving the plastic connector handle up and down slightly while exerting a gentle outward pull. Do not twist the connectors or move them from side to side. Remove coaxial cables by pulling straight out.

Logic cards should be removed with the aid of an extraction tool, P/N 2207 6228. To use the tool, locate the hole near the lower front corner of the desired card and insert the pry pin into it (pointed end of tool downwards). Pull outwards on the handle until release of the backplane connectors is felt. The card may now be readily removed from its slot. Refer to figure 5-5.

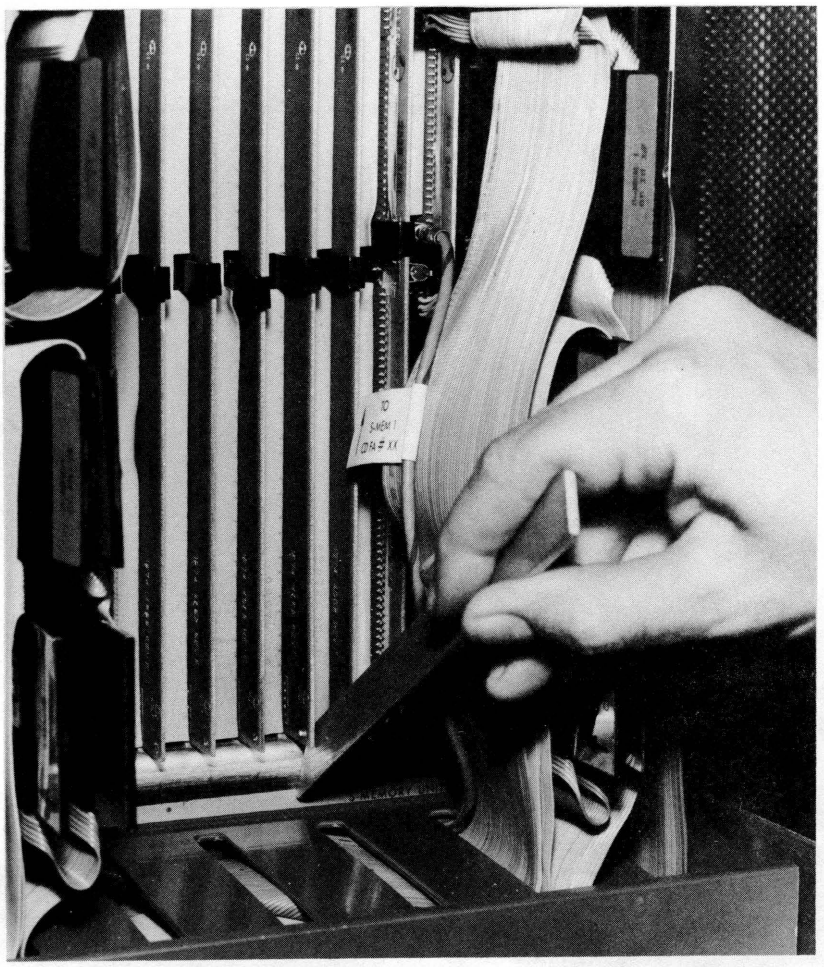


Figure 5-5. Logic Card Removal

Replacement of Logic Cards and Cables

When preparing to replace a logic card, first ascertain that it is oriented properly. The component side must be to the right when facing the frontplane. Slide the card carefully into the slot, making sure that the top and bottom edges are inside the guide tracks. When the limit of travel is reached, seat the backplane connectors by giving a firm push (using both hands) on the front edge of the card. Check to ensure that the card's front edge is even with those on either side of it.

Replace logic cables by visually aligning the socket and plug, then push straight together. Replace coaxial cables in the same way.

Memory Power Supply

Memory power supplies are used to provide the operating voltages for S-memory, with one such supply being utilized for each installed memory unit. The memory units each consist of a separate backplane with ten card slots, and can contain up to 64 kilobytes of storage. All memory power supplies are installed at the rear of the central system cabinet, as shown in figure 5-6. To remove a memory power supply, proceed as follows:

a. Remove the central system cabinet rear outer cover, and the side cover nearest the memory supply to be removed.

b. Making certain that system power is off, disconnect the ac and output leads from the supply by separating the in-line connectors on each. See figure 5-7.

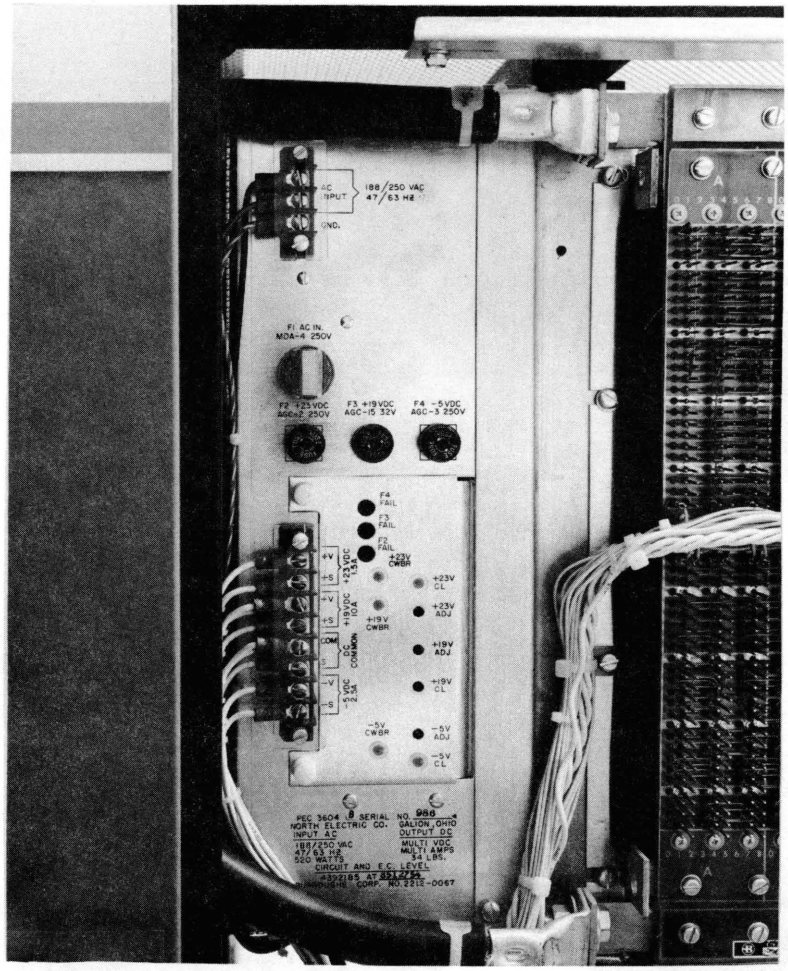


Figure 5-6. Memory Power Supply Location (Rear View of System Cabinet)

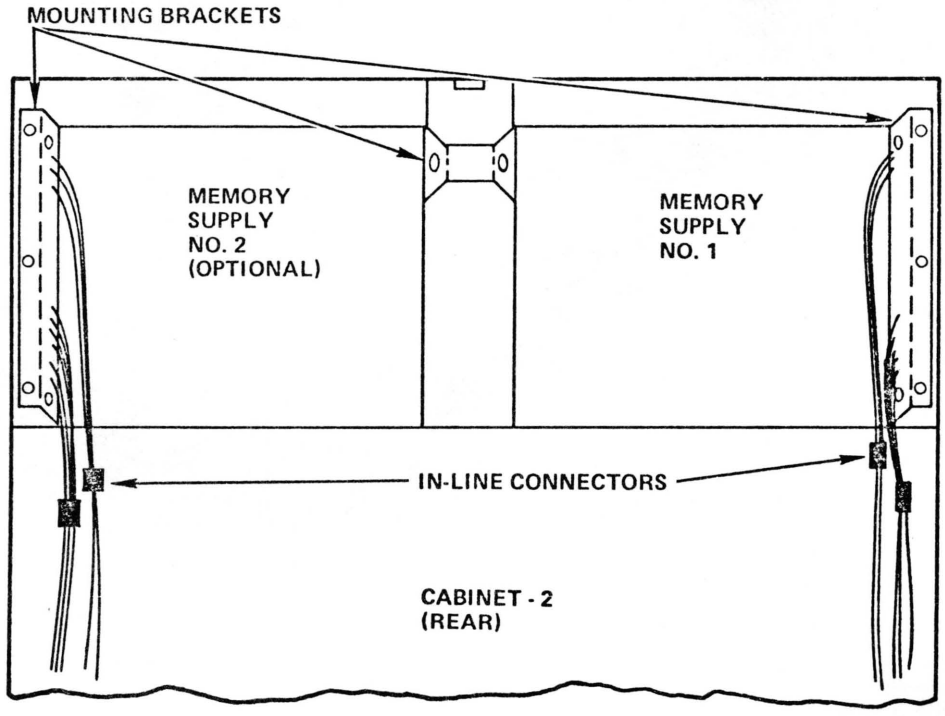


Figure 5-7. Memory Power Supply Removal

c. Remove the two retaining screws from the supply mounting bracket (attached to the rear inboard vertical frame member) and the single screw from the support bracket (center of top rear horizontal frame member), and lift the supply from the cabinet.

Logic Power Supplies

The logic power supplies are mounted in the base of the system cabinet, with each forming the lowest member of a ventilation stack. Each supply may be extended from the cabinet on tracks, and tilted for access to the underside. See figure 5-8. To gain access to the interior of a logic power supply, proceed as shown below. Refer to the Logic Power Supply Field Engineering Technical Manual, form number 1070281, for further information if it is necessary to remove the supply from the system.

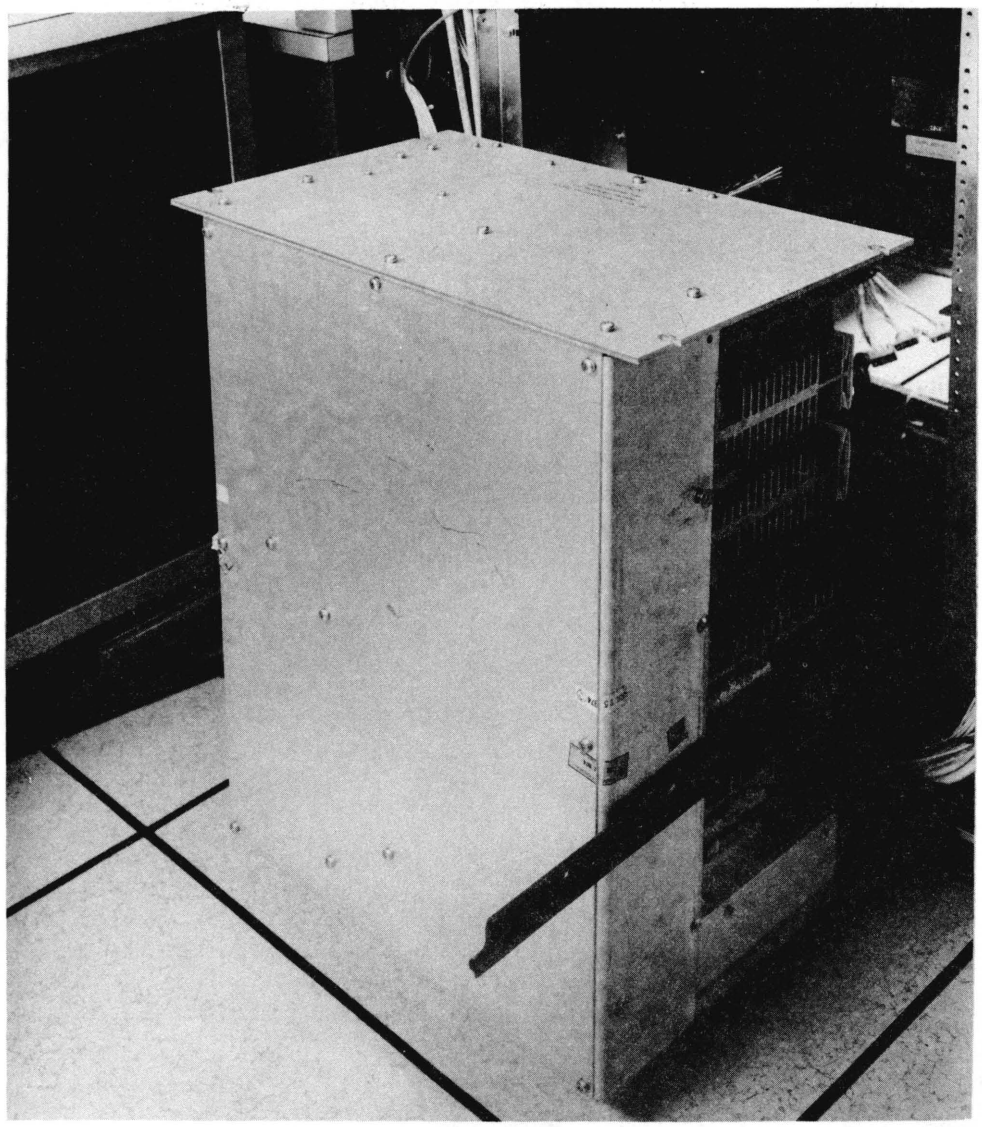


Figure 5-8. Logic Power Supply Extended and Tilted Upward

a. Make sure that system power is off.

b. Disconnect the three bus bar terminals at the rear of the supply (+4.75V, GND, and -2.0V outputs). Also disconnect plugs JLG, J12A, and J12B. The ac power cable need not be disconnected.

c. Remove the four retaining screws which attach the front panel of the supply to the cabinet frame. The supply may now be extended from the cabinet.

d. Remove the top and/or bottom covers of the supply as required.

COMPONENT REPLACEMENT

Replacement of individual circuit components in the B 1700 Central System should be performed in a manner consistent with that employed for other devices utilizing printed circuit technology. However, since the B 1700 uses integrated circuit chips and wire wrap terminals extensively, the special precautions to be observed when working with these parts are included below.

CAUTION

Never permit a hot soldering iron to come in contact with the insulated bus bars which cross the card.

Chip Replacement

To replace a faulty IC chip, proceed as follows:

CAUTION

Use of excessive heat damages the card etching.

a. Using a clean low-heat soldering iron and a solder sucker (part number 1622 2887), remove the solder from each leg of the chip. This should be done from the solder side of the board.

b. Using a soldering aid or small screwdriver, press each leg of the chip toward the chip body (on component side). This should break loose any residue of solder which may remain.

c. Lift out chip with a chip removal tool (part number 1622 4206). Do not pry with a screwdriver or other tool, as this can break the etching. Refer to figure 5-9.

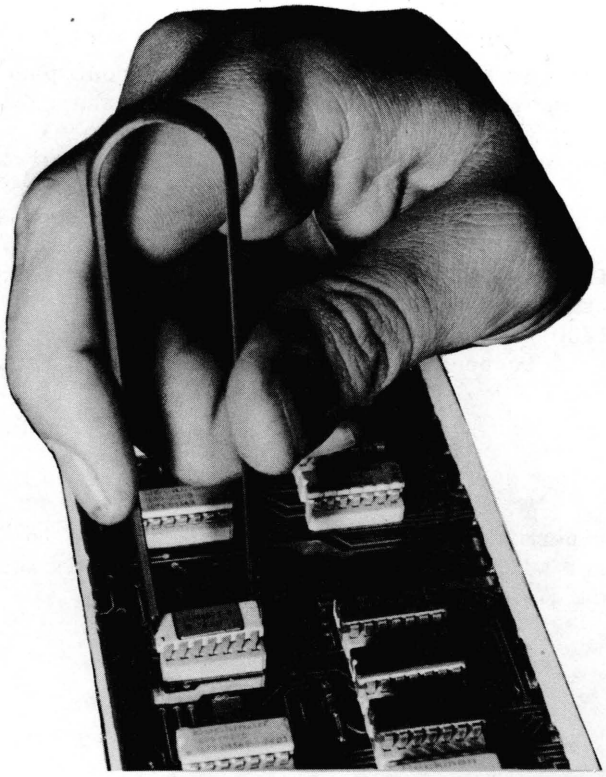


Figure 5-9. Integrated Circuit Chip Removal

Wire-Wrap Connections

Wire wrap connections provide a rapid, high quality joint, and are adaptable to automatic manufacturing processes. However, the tolerances involved are quite small, and require careful attention to detail when such connections are made by hand.

The following specifications should be observed when making wiring changes in the field:

Number of Turns: The minimum number of turns (per connection) of bare wire is five, and maximum number is seven. The maximum number of turns of insulation preceding the bare wire is three for any connection.

Insufficient Insulation: Wire insulation shall end no further than 1/32 inch from wire wrapped connections.

Wire and Terminal Contact: The bare wire and terminal must make contact on all corners following the point at which the origin of the number of turns are counted.

Separation of Turns: Turns may have a maximum separation of 1/2 the thickness of wire being used to make the wrapped connection.

Excessive Tail Wire: The wire tail shall be construed as being "that end of bare wire which follows the last wrap." The wire tail shall be parallel to the terminal surface.

Overlapping of Turns: This condition is caused when succeeding wraps overlap the ones previously made. If this condition exists, it is necessary to make a new connection.

Clearance: There shall be at least 1/32-inch clearance between grid pattern connections, terminals, bare wire, or components.

Height: The maximum clearance between the connector block and the first turn of the first connection shall be 1/16-inch.

Height for Single Wrap: The maximum height for a single wrap shall be 1/4-inch.

Height for Two Wraps: The maximum height for two wrapped connections shall be 1/2-inch.

Unwrapping: The connection shall be capable of being unwrapped from the wire-wrapped terminal without breaking. The unwrapping operation shall be done with a standard unwrapping tool only, so as to insure the life of the wire-wrapped terminal.

Wire Reuse: If a wire was previously wrapped, the portion of the wire which was wrapped cannot be used again. If the old wire is not long enough to strip off enough insulation to permit another wrap, a new wire must be routed in its place. Soldering a wrapped connection directly at the terminal is not recommended.

Terminal Reuse: Prior to re-wrapping, the terminal should be inspected for damage. When there is plating loss, corrosion, or other damage which would cause a poor connection, the terminal must be replaced.

Approved Wire Wrap Installation Procedure

To make an acceptable wire-wrap connection, proceed as shown below (refer to figure 5-10). Note that a separate wire-wrap tool is available for each gauge of wire which is used. Be sure that the correct tool is on hand before beginning work.

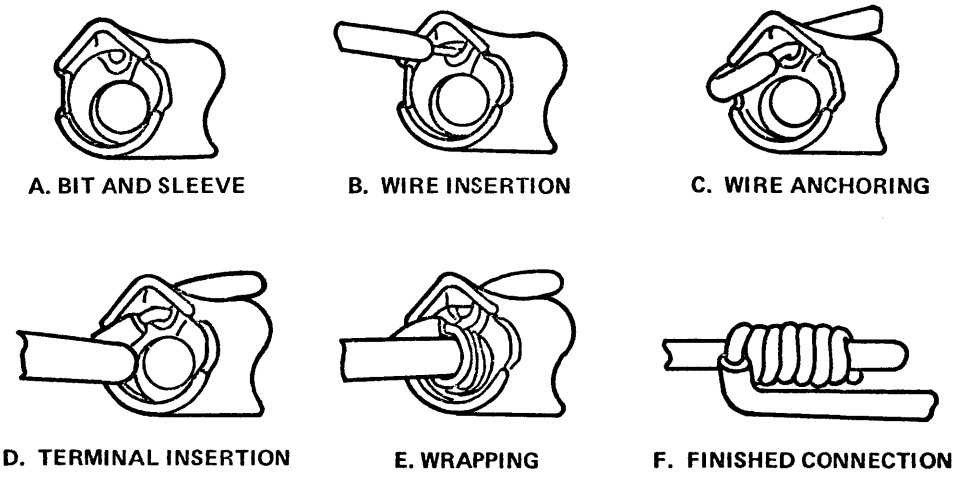


Figure 5-10. Wire Wrap Installation

a. Remove the insulation from the end of the wire. Approximately 1-1/4 inches of wire is required for a six-turn connection of wire.

b. Select the wire-wrap tool applicable for the specific gauge wire used.

c. Place the tool over the wire as shown in figure 5-10(b).

d. Anchor the wire as shown in figure 5-10(c) and insert the tool over the pin as shown in figure 5-10(d).

e. Rotate the tool in a clockwire direction. The wire wraps around the pin as shown in figure 5-10(e) and (f). Too much pressure causes the wire to bunch.

MAINTENANCE AIDS

To assist the field engineer in performing maintenance work, a number of publications and diagnostic programs have been produced. These include documentation which is part of the system and other data which is presented in subsequent portions of this section.

SYSTEM DOCUMENTATION

Each system which is assembled is furnished with a basic issue of test and field documentation. Included are schematics and logic diagrams for all portions of the system, assembly drawings, a backplane circuit list, diagnostic program listings, card test data, and a hardware rules book. Since these documents are used extensively during maintenance operations, a guide to the use of each is provided.

Logic Schematics

The B 1700 Logic Schematics are used to present a graphic representation of the circuits which make up the system. For reasons like those which require physical division of the system hardware into backplanes, cards and groups of cards, it was necessary to sectionalize the schematics. The sections thus produced follow the physical rather than electrical divisions of the system, with each individual schematic being identified with one of the logic cards. Where necessary to avoid crowding, several sheets are used to show the circuits on a single card.

Schematic Rules

To present a uniform appearance and avoid confusion, all schematics conform to a predetermined set of rules. Briefly, the rules for schematic layout are as follows:

a. Schematics deal with the logic contained on a single card, and may consist of one or more pages.

b. Logic flow is left to right.

c. Signals may be either uni-directional or bi-directional as circuit requirements dictate, but must be identified as such.

d. Uni-directional signals entering a card by way of frontplane connectors enter the schematic at the top of the page.

e. Uni-directional signals leaving a card by way of frontplane connectors leave the schematic at the bottom of the page.

f. Bi-directional signals using frontplane connectors may appear at either the top or bottom of the page.

g. Backplane and inter-page input signals enter the page from the left.

h. Backplane and inter-page output signals exit the page to the right.

Signal Names (Mnemonics)

To aid in understanding circuit functions and the tracing of signals through the logic, the output of each active device has been assigned a mnemonic name which comprises an abbreviated description of that signal's purpose. Mnemonics consist of eight characters, and are composed of letters, digits, special symbols, and/or spaces. The following rules apply to the composition of mnemonics.

a. All mnemonics must be eight characters in length; unused spaces are to be filled with periods.

b. The first character must be either a letter or a digit.

c. The seventh character is reserved for the letter corresponding to the logic card upon which the named signal originates.

d. The eighth character is reserved for indication of the logic card side on which the named signal appears or is most accessible. This is shown as a zero (0) to indicate component side, a one (1) for solder side, or a period (.) where not applicable.

e. The letters A through Z may be used.

f. Digits 0 through 9 may be used.

g. The following special symbols may be used:

= (equal)

≠ (not equal)

≥ (equal to or greater than)

≤ (equal to or less than

+ (plus)

- (minus)

* (asterisk)

← (left arrow)

→ (right arrow)

/ (not)

To illustrate the above rules, several examples are presented and explained:

Example a. PL←FA.R0

Move the contents of the FA register to left scratchpad. This control signal originates on the component (0) side of logic card R.

Example b. ASA00.E1

A-stack address bit 0 originates on the solder (1) side of logic card E.

Example c. BSWM0PH1

Basic swap micro operator. This control signal from the micro decoding logic originates on the solder side of logic card H.

Pseudo Connection Symbols

Since each B 1700 logic card may contain up to 120 integrated circuit chips, it is not practical to draw a complete card schematic on one page. In addition, the complexity of many circuits precludes the illustration of all interconnections between chips, in order to preserve the legibility of the drawings. To alleviate these problems, a system of pseudo connection symbols was created. These symbols are generally used whenever the illustration of a complete circuit is not practical. The pseudo connection symbols are divided into five categories, as follows:

a. Internal page symbols (figure 5-11)

b. Backplane pin symbols (figure 5-12)

c. Frontplane pin symbols (figure 5-13)

d. Inter-page connection symbols (figure 5-14)

e. Special symbols (figure 5-15)

Each type is fully described in the referenced illustrations. Note that a number beside such symbols indicates the number of places to which that signal is connected.

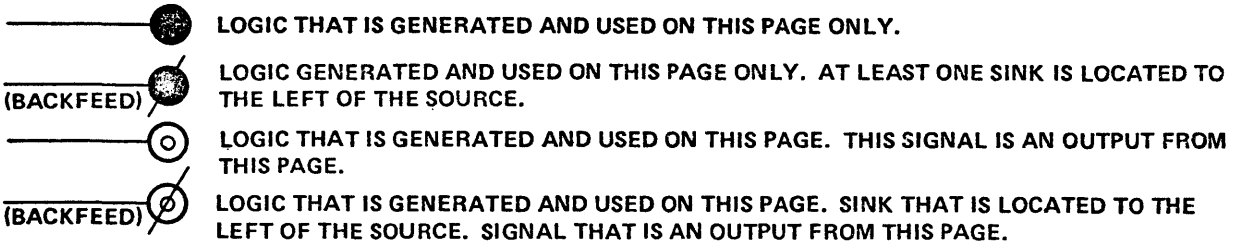


Figure 5-11. Internal Page Symbols

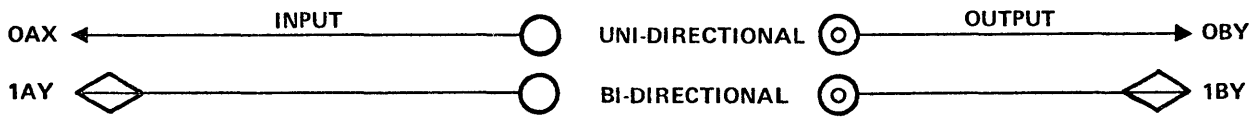
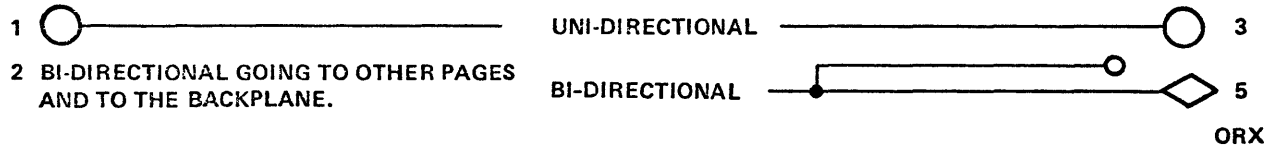


Figure 5-12. Backplane Pin Symbols

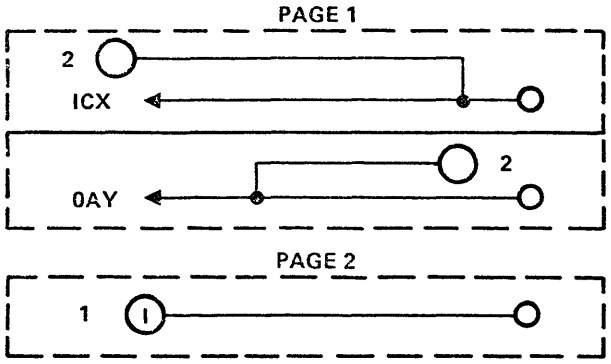
FRONTPLANE PINS ARE LABELED A THROUGH Z WITH O BEING OMITTED.

Figure 5-13. Frontplane Pin Symbols



THE NUMBER TO THE LEFT OF AN INPUT SIGNAL DESIGNATES THE PAGE WHERE THE SIGNAL ORIGINATED.

THE NUMBER TO THE RIGHT OF AN OUTPUT SIGNAL REFERENCES THE PAGE OR PAGES WHERE THAT SIGNAL IS USED.



LOGIC GENERATED ON PAGE 2 IS OR-ED WITH BACKPLANE INPUTS ON PAGE 1.

AN INPUT FROM THE BACKPLANE ON PAGE 1 WHERE THE LOGIC IS USED. IT IS ALSO USED ON PAGE 2 WHERE THE INPUT SIGNAL WILL CONTAIN AN (I).

THIS SYMBOL SIGNIFIES THAT THE INPUT IS A BACK-PLANE PIN INPUT TO PAGE 1.

Figure 5-14. Inter-Page Connection Symbols

A TRIANGLE PLACED PERPENDICULAR TO AN OUTPUT SIGNAL LINE DESIGNATES THAT A PULL DOWN OR LOAD RESISTOR IS CONNECTED TO THIS CIRCUIT. THE ACCOMPANYING ALPHA NUMERIC COMBINATION TELLS THE CHIP LOCATION AND PIN WHERE THIS RESISTOR IS LOCATED.

AN INVERTED TRIANGLE PLACED ON AN OUTPUT LINE DESIGNATES A 150 OHM LOAD RESISTOR TO GROUND ON THE SOLDER SIDE OF THE BOARD. THE ALPHA NUMERIC COMBINATION NEXT TO THE TRIANGLE INDICATES THE PIN LOCATION THE RESISTOR IS TIED TO.

Figure 5-15. Special Symbols

Hardware Rules Book

As with other kinds of components which are available in a variety of types to fit individual needs (e.g., transistors, diodes, vacuum tubes, etc.), the integrated circuit elements employed in the B 1700 are identified by a special designator code. The code is unique to B 1700 series systems, and serves as an abbreviated specification for commonly used devices. Since the system, for the most part, consists of a limited number of individual IC types, it is practical to compile detailed information concerning them in a single reference volume. This publication is known as the B 1700 Hardware Rules Book, form number A2209 6150A, and is an essential part of the system documentation.

The hardware rules book is arranged in alphabetical order by a code designator, with the designators having been assigned in a manner roughly approximating abbreviated device output functions. Device designator codes consist of four (4) characters, the first two of which are always alphabetical. The third character may be a letter or a number, depending upon whether one or more devices with the same general function (but with minor individual differences between them where there are two or more) exist. The last character of the code always appears as an N (meaning "number") in the references. Often such devices are packaged two or more to a chip, requiring that numbers be assigned to distinguish between them in actual applications. Such numbers appear only in the logic schematics and discussions which refer to them. Several examples of circuit element designator codes follow:

Designator Code	Chip Function
AFAN	Adder/subtractor
CFAN	Comparator
FFAN	Flip-flop
LFAN	Latch
RFAN	Three-bit register

Backplane Circuit Lists

For each portion of the system which uses a wire wrap backplane for interconnection of the plug-in logic cards, a backplane circuit list is provided. This is a complete listing, by signal name and pin numbers, of all point-to-point connections on the backplane itself. The listing serves to complement the schematics, providing a means of tracing wiring circuits external to the logic cards.

Card Test Data

One of the several means available for troubleshooting the B 1700 system is use of the logic card tester. This device provides an external means of exercising the circuit elements of any desired logic card. Cards so tested are subjected to an individualized routine which is designed to produce a predictable set of responses. The results of the tests, known as node counts, are compared with published specifications, thus serving as a means for detecting malfunctioning circuitry. The card test data serves as a guide to the use of the card tester, providing both set-up instructions and expected results. A separate section is provided for each logic card included in the system.

Diagnostic Program Listings

Diagnostic programs provide for system fault analysis through special exercising of suspected portions of the logic. Included with each such program is a listing which contains operating instructions and result analysis data. Refer to the diagnostic programs discussion for further information.

DIAGNOSTIC PROGRAMS

Because hardware failures often manifest themselves in ways not directly indicative of the actual fault and, additionally, due to the frequent difficulty in separating hardware problems from faulty software, a number of diagnostic programs have been produced. These programs exercise the processor with sequences of micro operators, and are designed to produce specific identifiable results. Included with each program is a listing of proper responses and those which are indicative of certain faults. All such test programs are stored on tape cassettes, and produce their responses by way of the 24 console lamps or the console printer (SPO). Diagnostic programs are provided in two types:

a. Maintenance Test Routines (MTR). These tests are loaded and executed directly from the cassette tape. Only one MTR program is currently available, this being the Processor Maintenance Test Routine.

b. Dynamic Tests. These tests are loaded from the cassette tape into M-string or S-memory, then executed from there. Three Dynamic tests are currently available:

   1. M-memory dynamic routine

   2. S-memory dynamic routine

   3. Processor dynamic routine

These should be employed as appropriate whenever actual or suspected hardware faults occur.

CENTRAL SYSTEM FUNCTIONAL GUIDE

The following data is intended to be used for general reference purposes during maintenance work. Included are directions for locating and/or identifying components, connections, pins, and other portions of the hardware.

INTEGRATED CHIPS

Integrated circuit chips are employed which are of the 14-, 16-, and 18-pin dual in-line package type. The individual pins are assigned identifying letters as shown in figures 5-16, 5-17, and 5-18. Certain pins in all dual in-line chips are reserved (dedicated) for specific purposes. These are as follows:

Pin E = ground

Pin M = +4.75 volts

Pin L = -2 volts

14 PIN CHIP

SEVEN PINS ARE ON EACH SIDE OF THE CHIP. ONE SIDE IS LETTERED A THROUGH G, THE OTHER SIDE IS LETTERED H THROUGH P WITH THE LETTERS I AND O BEING OMITTED.

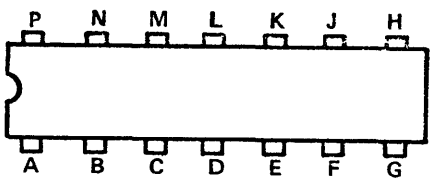


Figure 5-16. 14 Pin Chip

16 PIN CHIP

THE 16 PIN CHIP IS SIMILAR TO THE 14 PIN CHIP EXCEPT FOR THE ADDITIONS OF PIN R AND PIN S.

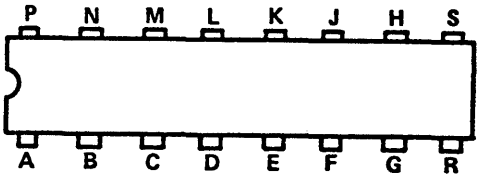


Figure 5-17. 16 Pin Chip

18 PIN CHIP

THE 18 PIN CHIP IS SIMILAR TO THE 16 PIN CHIP EXCEPT FOR THE ADDITION OF PIN T AND PIN U.

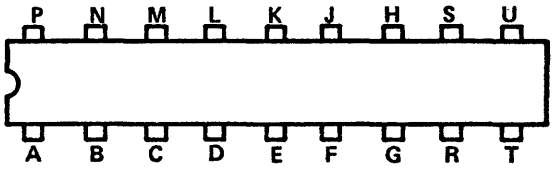


Figure 5-18. 18 Pin Chip

Otherwise, the individual types of chips vary greatly as to pin connections. For further information on chip types and functions, refer to the B 1700 Hardware Rules Book, form number A2209 6150A.

B 1700 LOGIC CARDS

Circuit logic for the B 1700, including the processor, S- and M-memories, the port interchange, the I/O base, and individual I/O controls is physically packaged on pluggable, etched circuit cards. The logic cards are uniform in overall configuration, and have the following characteristics:

a. 14-1/2" long by 12-1/2" high, with etching on both sides.

b. Maximum capacity is 120 integrated circuit chips.

c. All chips are located on the same side of the card, which is known as side 0.

d. Solder connections and some discrete components are located on the opposite side (side 1).

e. Signal connections between individual chips are made primarily by wires which attach to wire-wrap terminals.

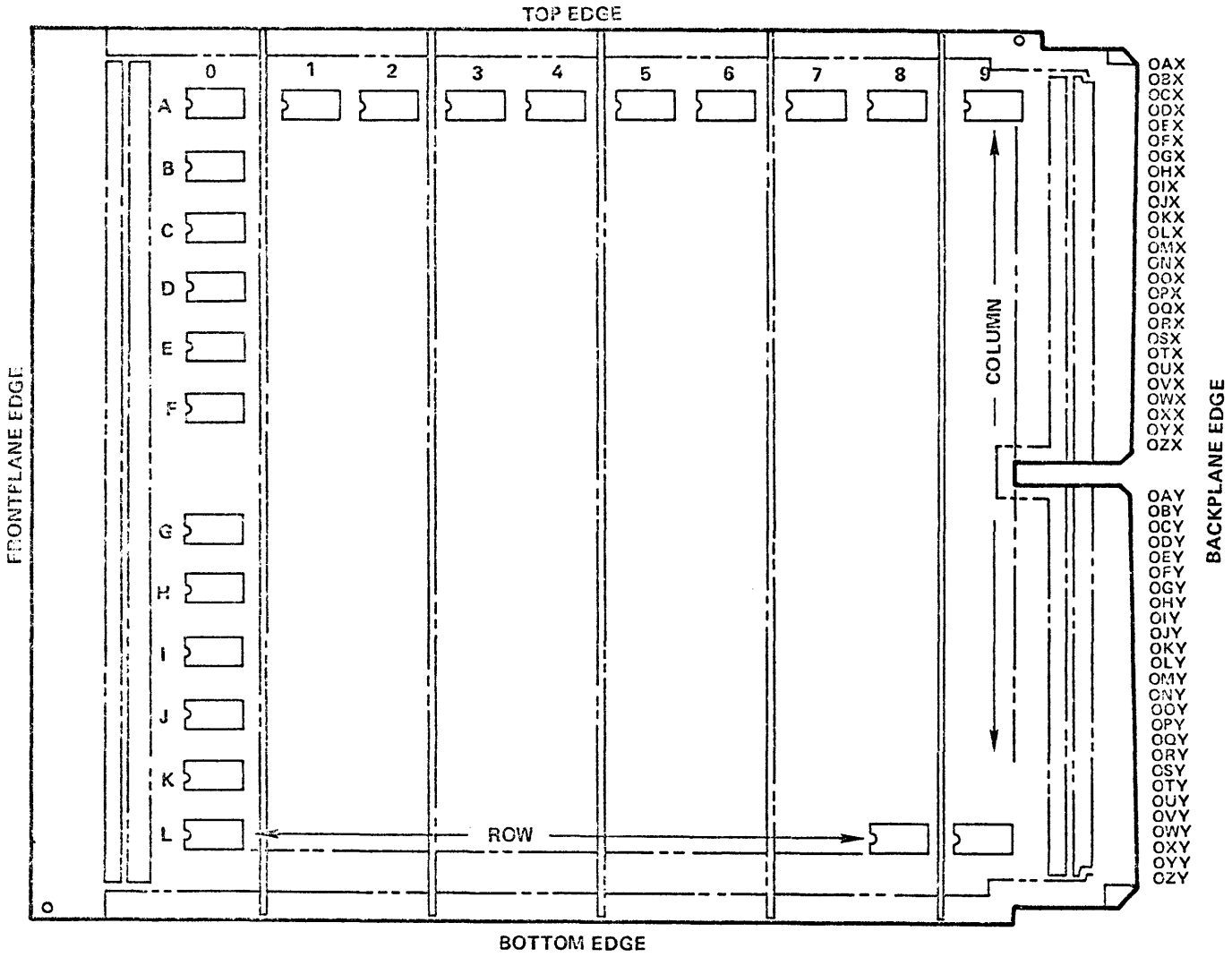
f. Operating voltages and the system clock signal are distributed by way of bus bars which are mounted perpendicular to the surface of the card, and cross it lengthwise.

g. Signal connections into and out of the cards are by way of frontplane and backplane connectors. The maximum number of connections to one card is 200, divided evenly between frontplane and backplane.

h. Clock signals are distributed to the cards by way of coaxial cables.

Chip Locations

To provide a rapid, easily understood method of identifying chip positions on logic cards, a system of coordinates is employed. In this system, vertical columns of chips are assigned numbers (0 through 9) and horizontal rows letters (A through L). Thus a chip's identity is determined by the column and row coordinator which it occupies. These coordinates are used exclusively in the card schematics. Chip location coordinates are illustrated in figure 5-19.



NOTE:
THE ROW, COLUMN COORDINATE LOCATION OF A CHIP IS USED AS IT'S REFERENCE DESIGNATION ON THE CARD LOGIC DIAGRAM; EG.,A CHIP LOCATED IN ROW C, COLUMN 5 HAS THE REF. DESIGNATION C5.

Figure 5-19. Coordinates for Chip Locations on Component Side of Board

Pin and Connector Designations

Input/Output connections are separated into several groups which correspond to physical sections of the card. To provide a distinctive method of identifying the location of a particular connection, certain conventions have been adopted. These differ for frontplane and backplane connections, but follow the overall scheme of identifying those located on the upper-half of the card with an X, and those on the lower-half with a Y.

Backplane pins are identified with a three-character code which describes the pin's location. Essentially, the pins are divided into four groups of 25 each, with the divisions being the upper and lower card halves, and the component (0) and solder (1) sides of the card. The location code is explained as follows (refer to figure 5-20):

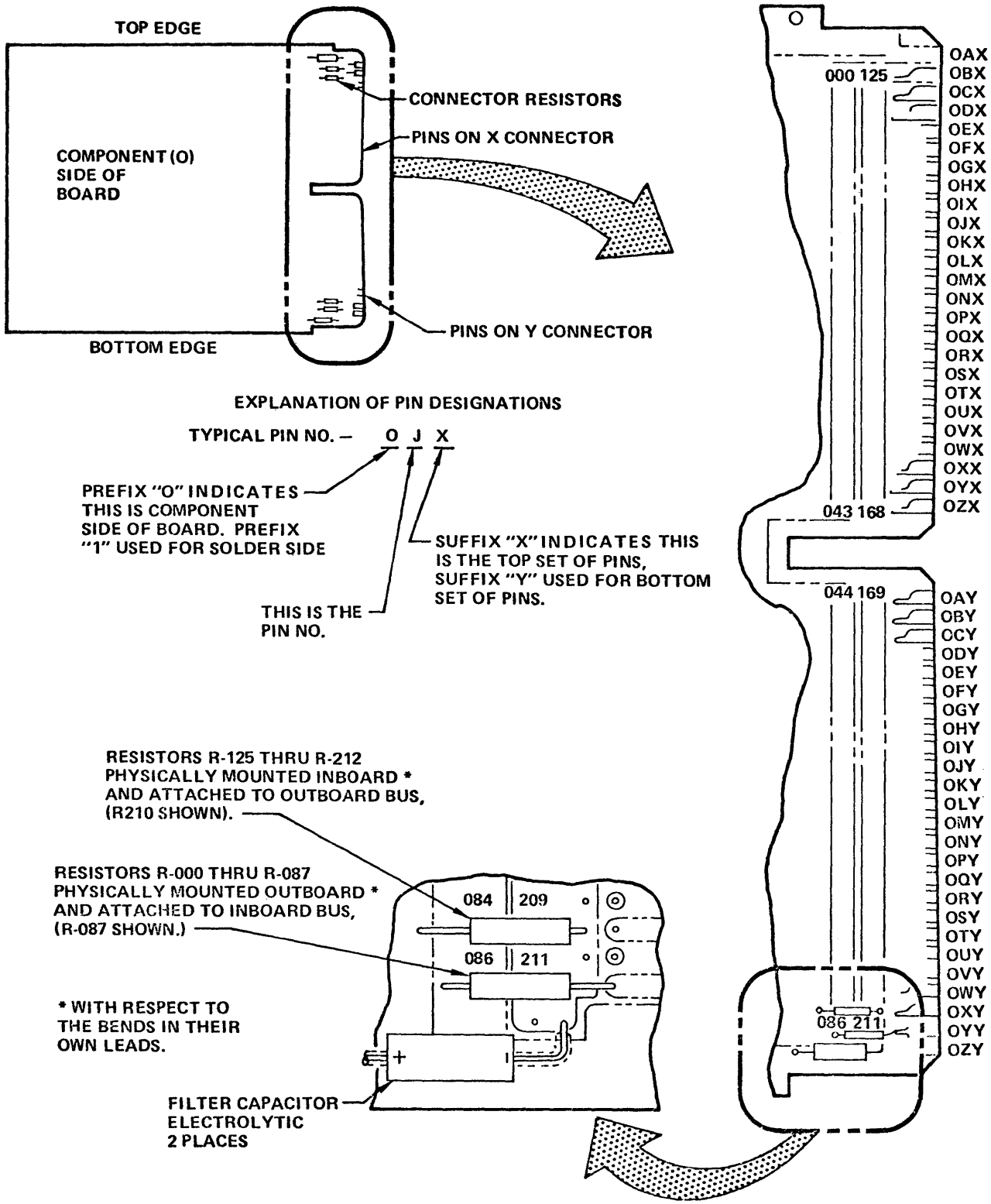
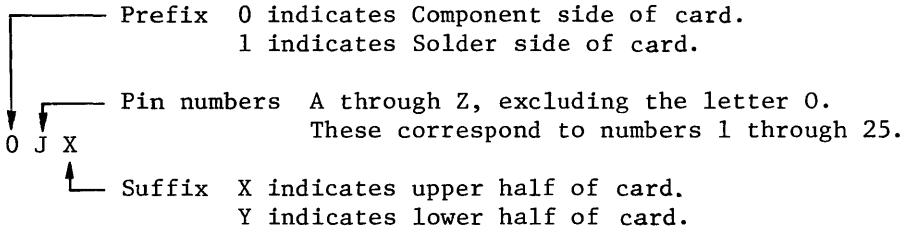
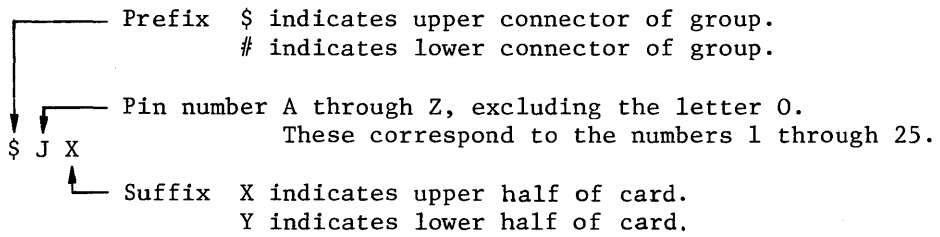


Figure 5-20. Backplane Pin and Resistor Designations



Frontplane pins are also identified with three-character codes. However, since the frontplane connectors are all located on the same side of the card and are four in number, a slightly different identity system is employed. In this system the X and Y designations are retained, and an additional symbol is added to indicate the board half concerned. The symbols $ and # (dollar sign and pound) have been adopted for this purpose. Dollar sign indicates the upper connector of the X or Y group, and pound the lower. Thus the frontplane pin designations appear as follows: (refer to figure 5-21):



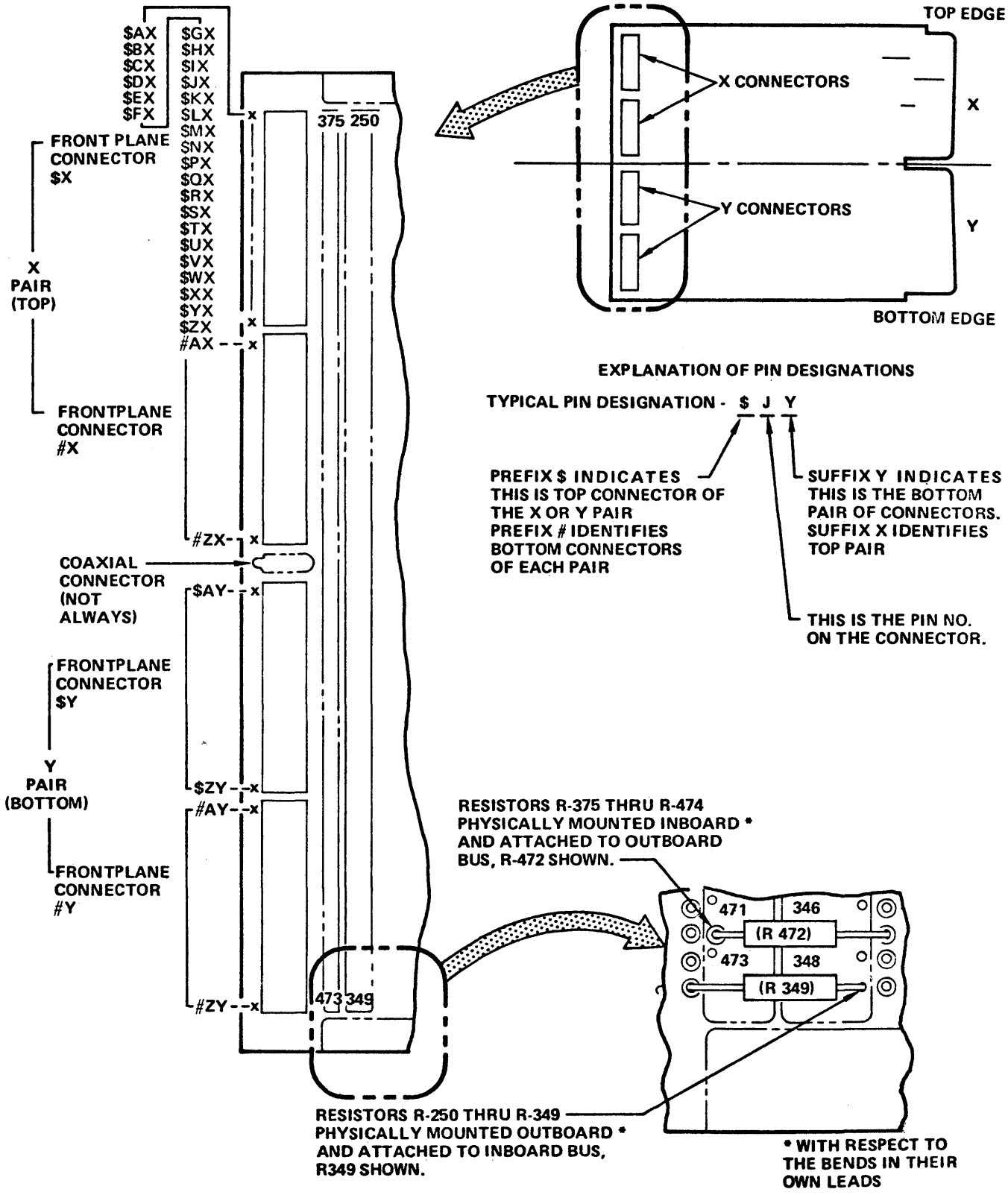


Figure 5-21. Frontplane Pin and Resistor Designations

Discrete Component Locations

In addition to integrated circuit chips, discrete resistors and capacitors are used on the logic cards. These components are installed in the following ways:

Capacitors:

a. Located in a vacant chip position.

b. Located on a jumper chip.

c. Located along etched voltage distribution busses.

Resistors:

a. Located in a vacant chip position.

b. Installed adjacent to frontplane or backplane connector etching. See resistor designations (below).

c. Located on solder side of card between a chip pin and ground.

Resistor Designations

Many of the circuit elements which are connected to other places outside the confines of their own particular logic card employ pull up or pull down resistors to ensure that reliable control levels are maintained. Because such use is frequent, a special area has been provided for the installation of these resistors adjacent to the frontplane and backplane connectors. For location purposes, resistors mounted in the locations mentioned are assigned numbers which also appear in the card schematics. The numbering scheme for pull up/pull down resistors is illustrated in figures 5-20 and 5-21.

S-MEMORY STORAGE CARDS

S-memory storage cards are similar in size and design to the logic cards, but differ in the chip mounting configuration employed. Essentially, the card consists of 8 columns (numbered 0-7) and 20 rows (lettered A through U, excluding O). Chip locations are identified in the same manner as with logic cards. Note that RAM storage chips are located only in columns 0 through 3. Fully-populated cards have all four columns filled with chips, whereas half-populated cards have two columns. Refer to figure 5-22.

M-STRING MEMORY CARDS

M-string memory cards are very similar to the logic cards, with the same number of chip positions and the same coordinate system for location. Chip identity within the 1024-word storage capacity is shown in figure 5-23.

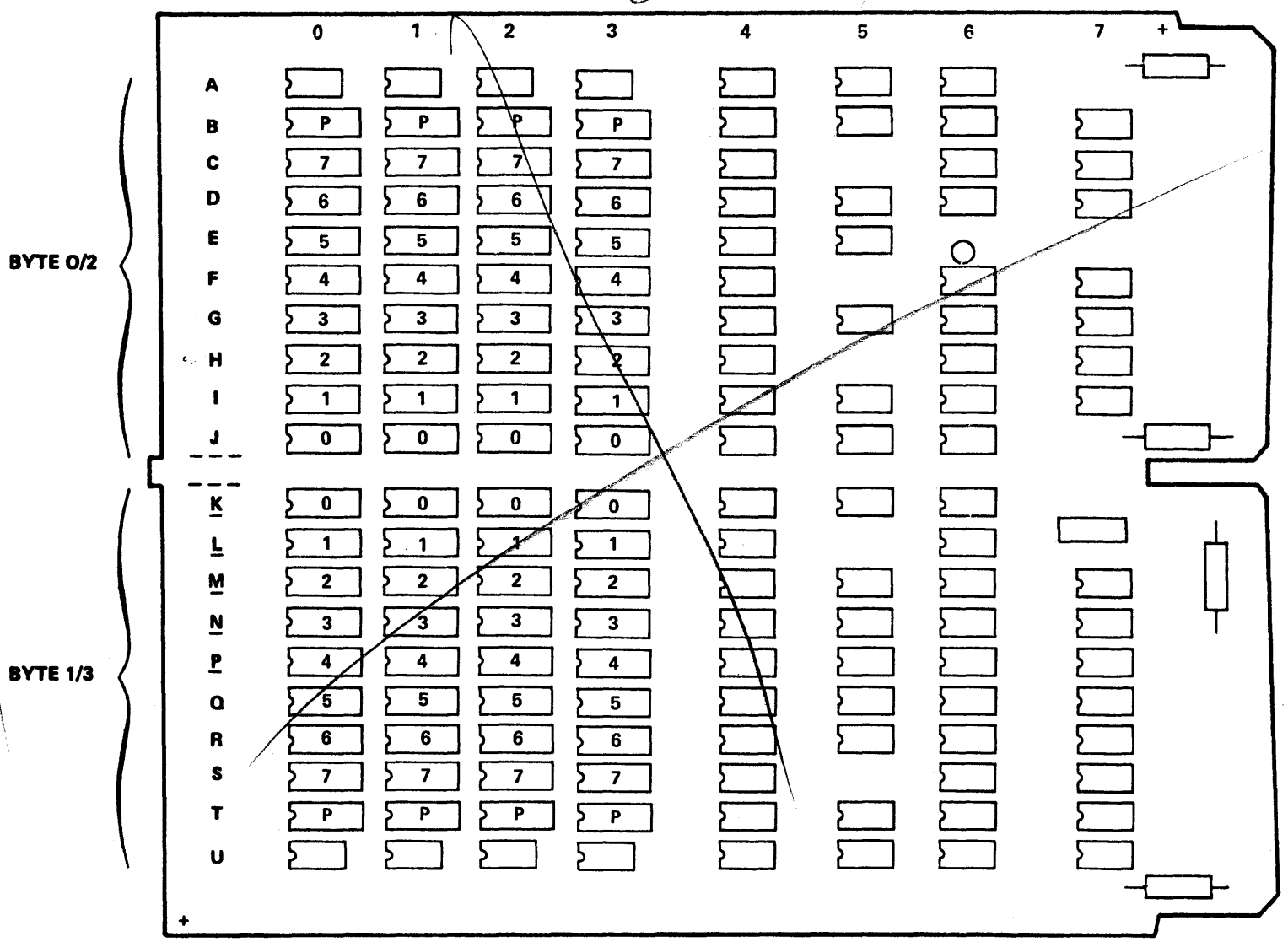


Figure 5-22. The Main Memory Storage Card

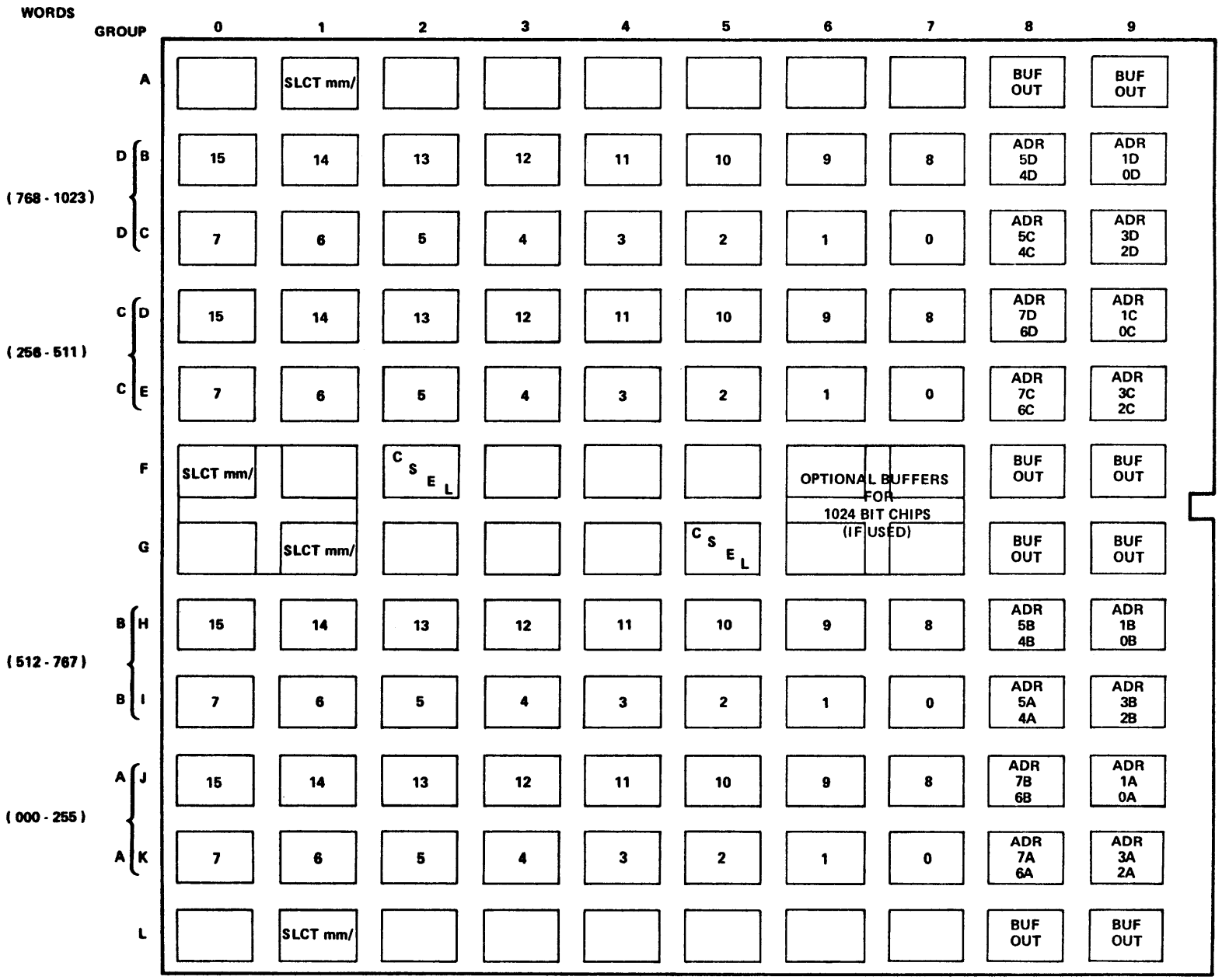


Figure 5-23. M-String Memory Storage Card

COMPONENT AND CIRCUIT LOCATION DATA

The following pages are for reference use in locating subassemblies, logic cards, circuit elements, and signals. Included are the following:

a. Central system subassembly locator (figure 5-24).

b. Central system card and cable locator (figure 5-25).

c. Logic card circuit chart (figure 5-26).

d. Rapid circuit locator.

e. Logic card functions and inputs/outputs (figures 5-27 to 5-46).

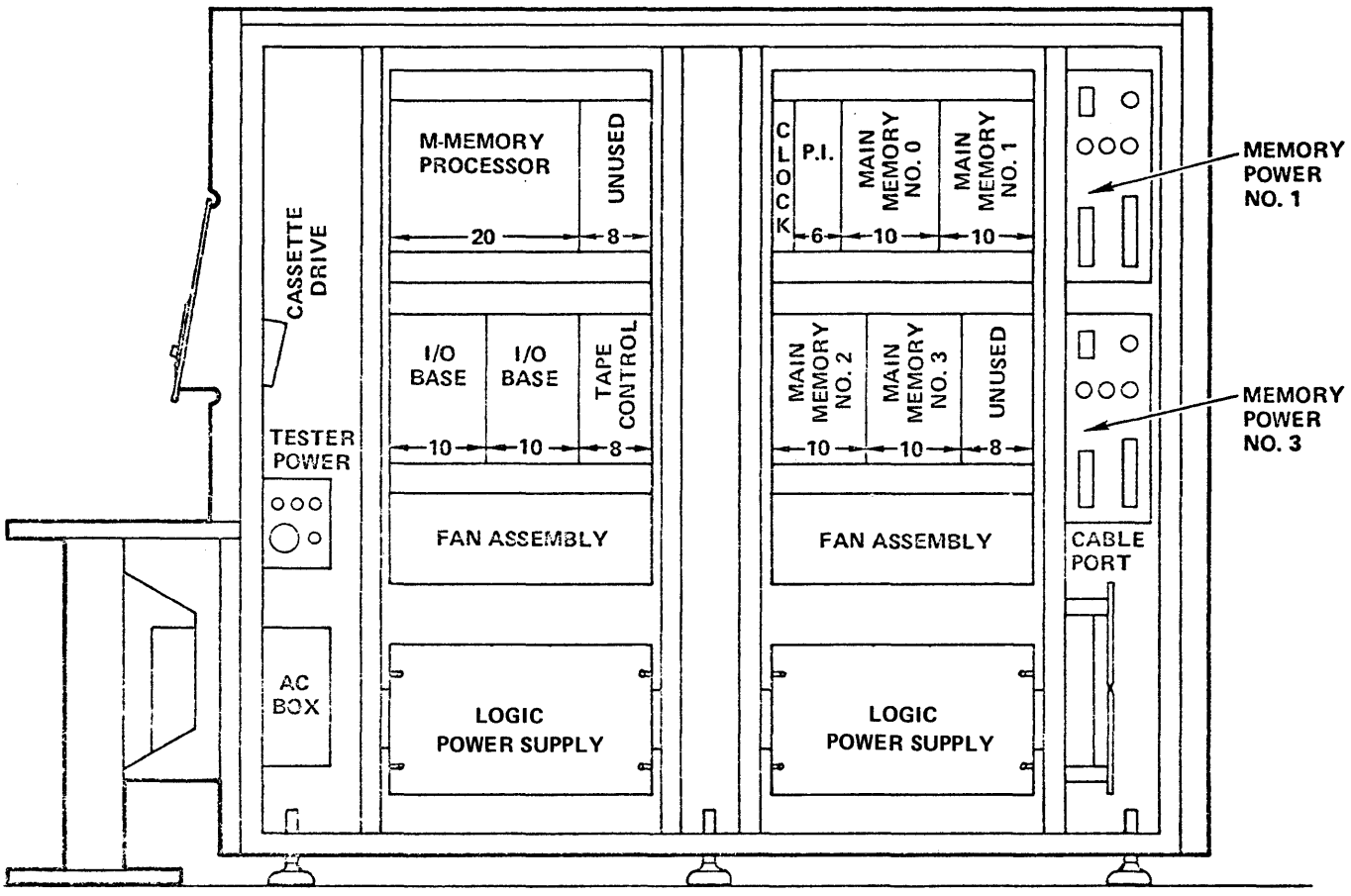


Figure 5-24. Central System Subassembly Locator

M-MEMORY PROCESSOR

M N	C	R	D	L	J K	A B	E	F	P	Q	H G
A REGISTER M REGISTER MSM READ DATA MOP LINES EXPOSE	FA, FB REGISTERS. LEFT SCRATCHPAD. RIGHT SCRATCHPAD.	CONTROL SIGNALS FOR OTHER CARDS.		CP REG. DECODING FOR J AND K. CASSETTE LOGIC.	X AND Y REGISTERS. 26-BIT FUNCTIONS.	T REGISTER. L. REGISTER CA, CB, CC, CD. ROTATOR. MASK. CONSOLE LAMP DRIVE REG.	SFL. SFU. A STACK	4-BIT FUNCTIONS SLOW SOURCES.	BASIC MICRO AND ADDRESS CONTROL. FINISH LOGIC. SOURCE AND SINK CONTROL LOGIC.	BR. LR. CMND. DATA. CA. RC.	PA:PD INTERFACE MBR. TOPM. AOB.

PORT# INTERCHANGE

A	B	C	PA
MIR MASK-REG. MERGER PARITY: NO# (CHECK/GEN./ FORCE)	ROTATOR AND CONTROLS MASK-GENERATOR WRITE-DATA-DRIVERS	HOLD-REG. MAR ADDRESS-BOUNDARY REFRESH PRIORITY MEM. TM/CNT REQ. PFG. DISPATCH REG REAL-TIME CLOCK	DATA-INTERFACE CONTROL - INTERFACE PORT. ADPT ENABLE-LOGIC

Figure 5-26. Logic Card Circuit Chart

Rapid Circuit Locator

	Name	Card	Sheet
A	A+MBR Adder	M	5
	A+MBR Register	G	7
	A-Register	M	1
	A-Register Branching Logic	M	2
	A-Register Increment Counter	M	3
	A-Register Move Decoding	M	3
	A-Register Output Buffers	M	5
	A-Stack and Input Latches	E	5/6
	A-Stack Input Control	E	1
	A-Stack Pointer and Read/Write Control	E	7
	Auxiliary 4-Bit Bus - Gating XYCN, BICN, XYST, and FLCN	F	6
B	Bad FA Address Comparator	Q	5
	4-Bit BCD Correction Logic	K/J	5
	Bias Logic	E	3/4
	BR Register		
	BR Input From MEX	Q	1/2
	BR Gating To MEX	Q	7
C	C-Register (CA, CB, CC and CD)	A	4
	C-Register Input From MEX, 4-RSLT DR XDSP	A	1
	C-Register Gating To 4-Bit Exchange	A	6
	CA and RC Signal Generation	Q	7
	Cassette Logic	L	6/7
	Cassette Sequence Counter	L	5
	Concurrency Control	P	3
	Concurrency Set Check Logic	N	3
	Console Count FA/FL Logic	P	5

	Name	Card	Sheet
	Console Lamp Register	A	7
	Console Load/Display Logic	N	4/5
	Count FA Adder	C	5
	Count FL Adder	D	5
	CP Register	L	1
	CP Register Controls	E	3/4
	CP Register Output Control	J	6/7
	CP and CPU Decoding	L	7
	Current State Generation (Processor)	P	1
	Current State Generation (PAPDIC)	H	6
D	Data Gating Enables and Controls	P	6
	Decoding For 4-Bit Source Addresses	N	7
	Dispatch 7 Bits - Gating To T- and L-Registers	A	1
	Dispatch 7 Bits Source and Control	H	3
	Dispatch 7 Bits Source Logic	A	2
	Done and In Process Control Level Generation	H	9
	Drive Data Select (Write Data Or Address)	G	2/3
E	Expose Flip-Flops	N	4
F	FA Adder	C	5
	FA Address Comparator	Q	5
	FA Register	C	3/4
	FB Register (including FL, FU, FT)	D	3/4
	FB Gating To 4-Bit Exchange	D	6
	FB Input Control	D	1/2
	FB Source Controls	D	6
	Finish Logic	P	2
	FL Count Up/Down Logic	D	5

	Name	Card	Sheet
	FLCN Logic	E	2
	FLCN Generation	D	5
	4-Bit Function Box	F	2/3
	4-Bit Function Box Branch and Skip Generation	F	3
	24-Bit Function Box, Bits 0 through 11	K	1
	24-Bit Function Box, Bits 12 through 23	K	2
	24-Bit Function Box, Adder-Subtractor	J/K	3/4
	24-Bit Function Box, Output Control	K	6/7
H	Halt Logic	P	1
I	Increment A Counter	M	3
	Interrupt Condition Storage (From Port Interchange)	H	2
	I/O Base/MEX Gating (Bi-Directional)	Q	3
	I/O Bus DATA and CMND sync Logic	P	7
	I/O Bus Gating To MEX	Q	6
	I/O Service Functions	Q	7
L	L-Register and 4-Bit Exchange Input	A	3
	L-Register - Gating To MEX (24 Bits)	A	5
	L-Register - Gating To 4-Bit Exchange	A	6
	L-Register - MEX, 4-RSLT and XDSP Inputs	A	1
	Load/Display Micro Generation (From CSW)	N	5
	LR Register	Q	1/2
	LR Register - Gating To MEX	Q	6
M	Mask Control (T-Register)	B	4
	MAXM and MAXS Jumpers	K	5
	Memory Base Register (MBR)	G	5/6
	Memory Cycle Startup Logic	H	8

	Name	Card	Sheet
	MEX Inputs From A-Register, Console Switches, and Eight Least-Significant Bits of MOP Lines	M	4
	MEX/I/O Base Gating (Bi-Directional)	Q	3
	MEX/MSM Gating (Write Data)	Q	4
	Micro and Address Control	P	1
	Micro Decoding - Control Signals	R	2/3/4
	Micro Decoding - 4-Bit	N	6/7/8
	Micro Decoding For Cards J, K, and L	L	2
	Micro Decoding 4-Bit Sink Of FB, L, T, and TOPM	R	1
	Micro Decoding 24-Bit Function Box Outputs	L	4
	Micro Decoding General Function and Register Manipulation	F	4
	Micro Decoding General Register Source and Sink	F	5
	Micro Decoding PAPDIC Controls	H	1
	Micro Decoding - 7C and 2D Register Control	H	8
	Micro Decoding - 4D and 5D Shift/Rotate X/Y	F	6
	Micro Decoding For X and Y Registers	L	3
	M (Micro) Register	N	6
	M-Register Inputs From MSM and MEX	N	1/2
	MOP Inputs From XPOS and Load/Display Logic	N	5
P	PAPDIC Control Logic	G	1
	PAPDIC Control Logic	H	7
	PAPDIC Current State Generation	H	6
	PAPDIC Done and In Process Controls To Other Logic	H	9
	PAPDIC Interface Control Drive and Receive Logic	H	5
	PAPDIC Line Drivers and Receivers	G	4
	PAPDIC Request Signal Generation	H	4
	Processor Current State Generation	P	1

	Name	Card	Sheet
R	Register Move Generation - X-, Y-, T-, and L-Registers	H	8
	Request Signal Generation (PAPDIC)	H	4
	4-Bit Result Exchange Gating To T- and L-Registers	A	1
	Rotation Control (T-Register)	B	1
	Rotator (T-Register)	B	2/3
	Run Indicator Operational Logic	P	8
S	Scratchpad Address Control Logic	R	5
	Scratchpad (Left) and Input Control	C	1/2
	Scratchpad (Right) and Input Control	D	1/2
	Scratchpad Relate Adder/Subtractor	C	6
	SF Register (SFL and SFU)	E	2
	SF (SFL, SFU, and A-Stack) Input Control	E	1
	Shift Control (T-Register)	B	1
	Sink Shield (Protects 4-Bit Registers)	A	6
	S-Memory Drive Data Select	G	2/3
	S-Memory M-Fetch Address Adder	G	7
	Source and Sink Control Logic	P	6/7
	Stop On CSW Address Comparator	M	6
	Suppress Finish Generate	P	4
T	T-Register - and 4-Bit Exchange Input	A	2
	T-Register - MEX, 4-RSLT and XDSP Inputs	A	1
	T-Register - Gating To MEX	A	5
	T-Register - Gating To 4-Bit Exchange	A	6
	T-Register Mask Control	B	4
	T-Register Shift/Rotate/Extract Controls	B	1
	TOPM Register	G	6
	Transfer Vector (Width and Sign) Generation	H	3

	Name	Card	Sheet
U	U-Register and Control	L	
W	Write Enable Control Logic	H	4/6/7
X	X-Register (Bits 0 through 11)	J	1
	X-Register (Bits 12 through 23)	K	1
	X-Y Adder/Subtractor (Bits 0 through 11)	J	3/4
	X-Y Adder/Subtractor (Bits 12 through 23)	K	3/4
	X-Y Conditions Generation	J	2
	X-Y 4-Bit Correction Logic	K/J	5
Y	Y-Register (Bits 0 through 11)	J	1
	Y-Register (Bits 12 through 23)	K	1
	X-Y Adder/Subtractor (Bits 0 through 11)	J	3/4
	X-Y Adder/Subtractor (Bits 12 through 23)	K	3/4
	X-Y 4-Bit Correction Logic	K/J	5

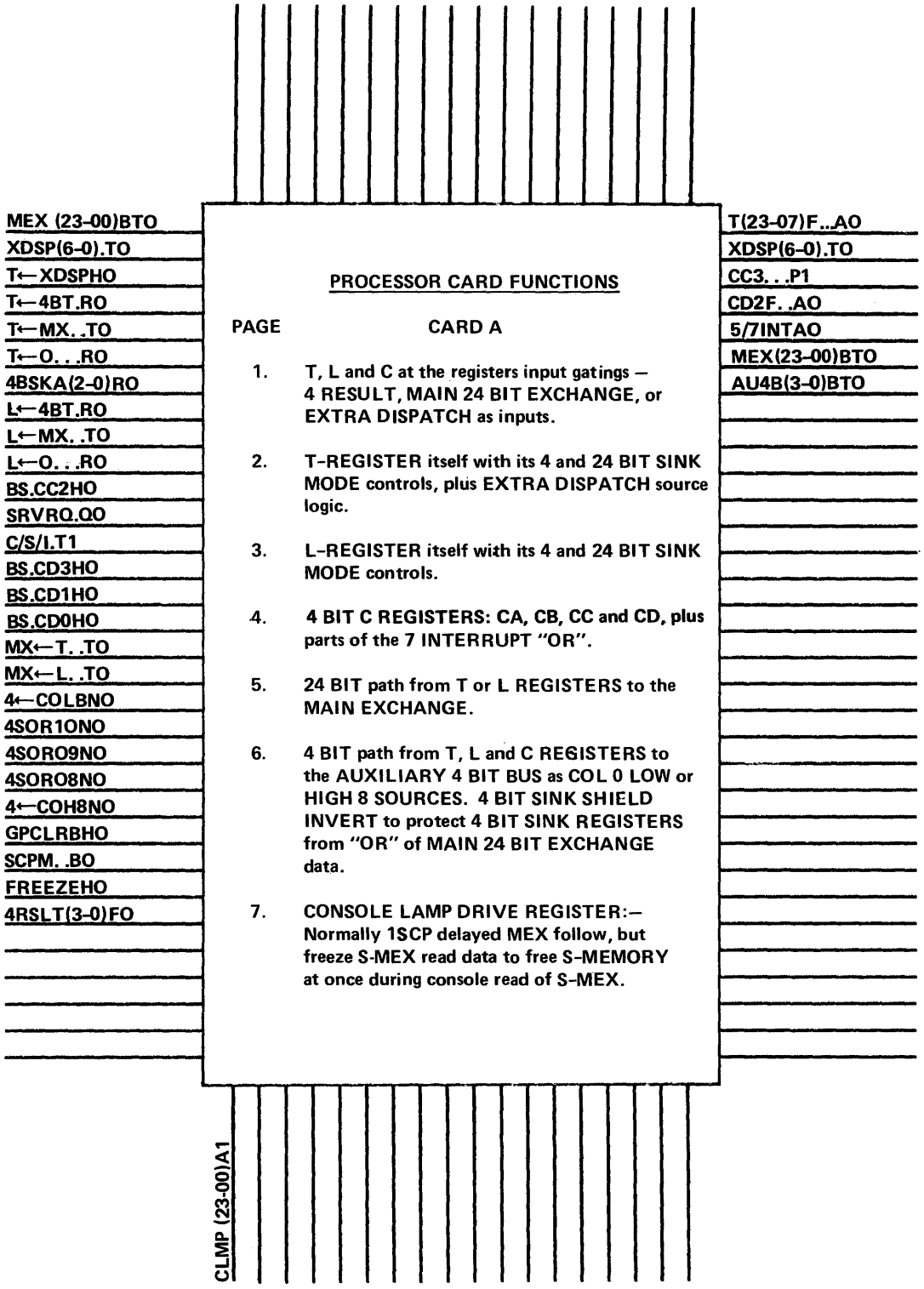


Figure 5-27. Processor Card A — Functions and Inputs/Outputs

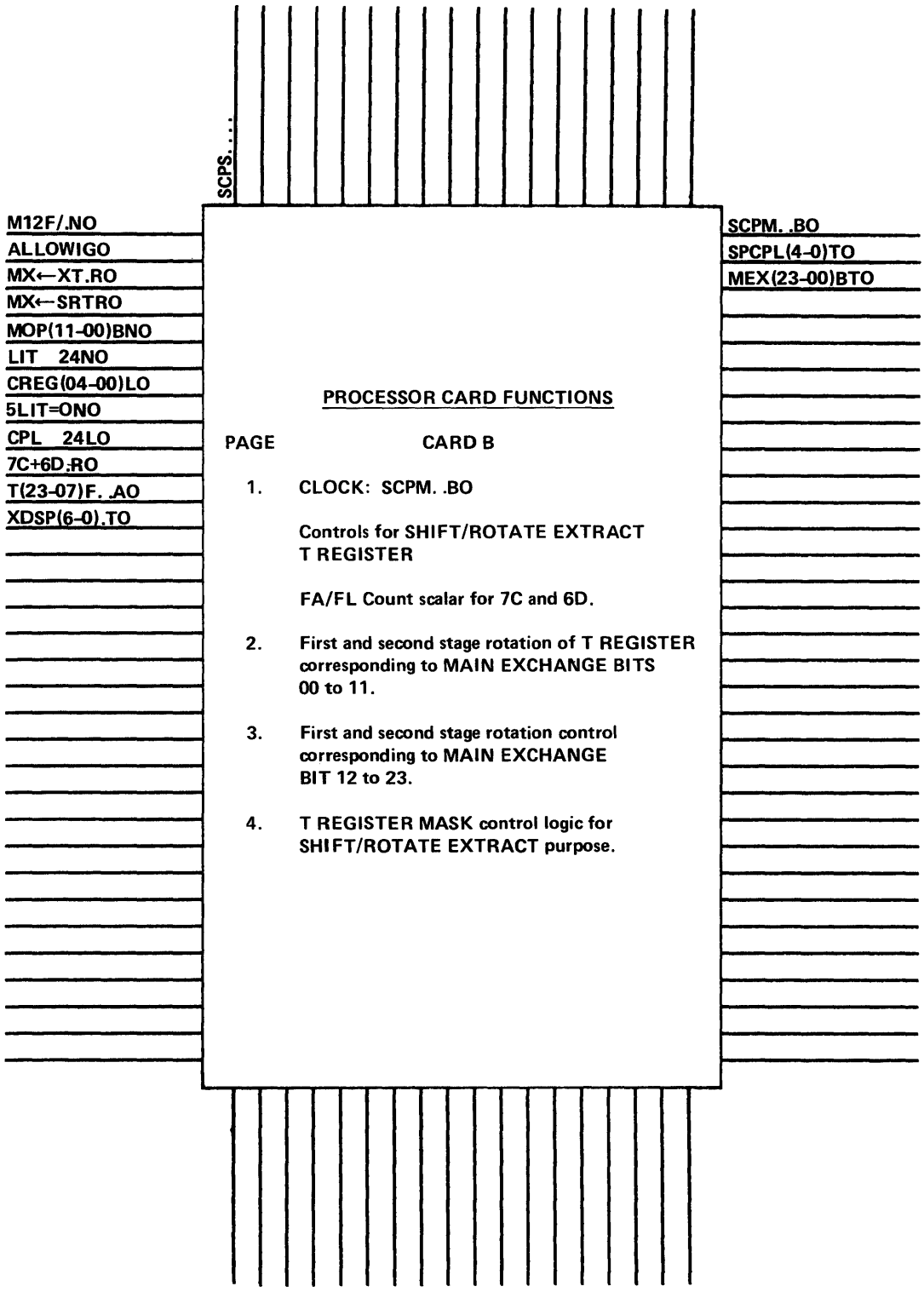


Figure 5-28. Processor Card B — Functions and Inputs/Outputs

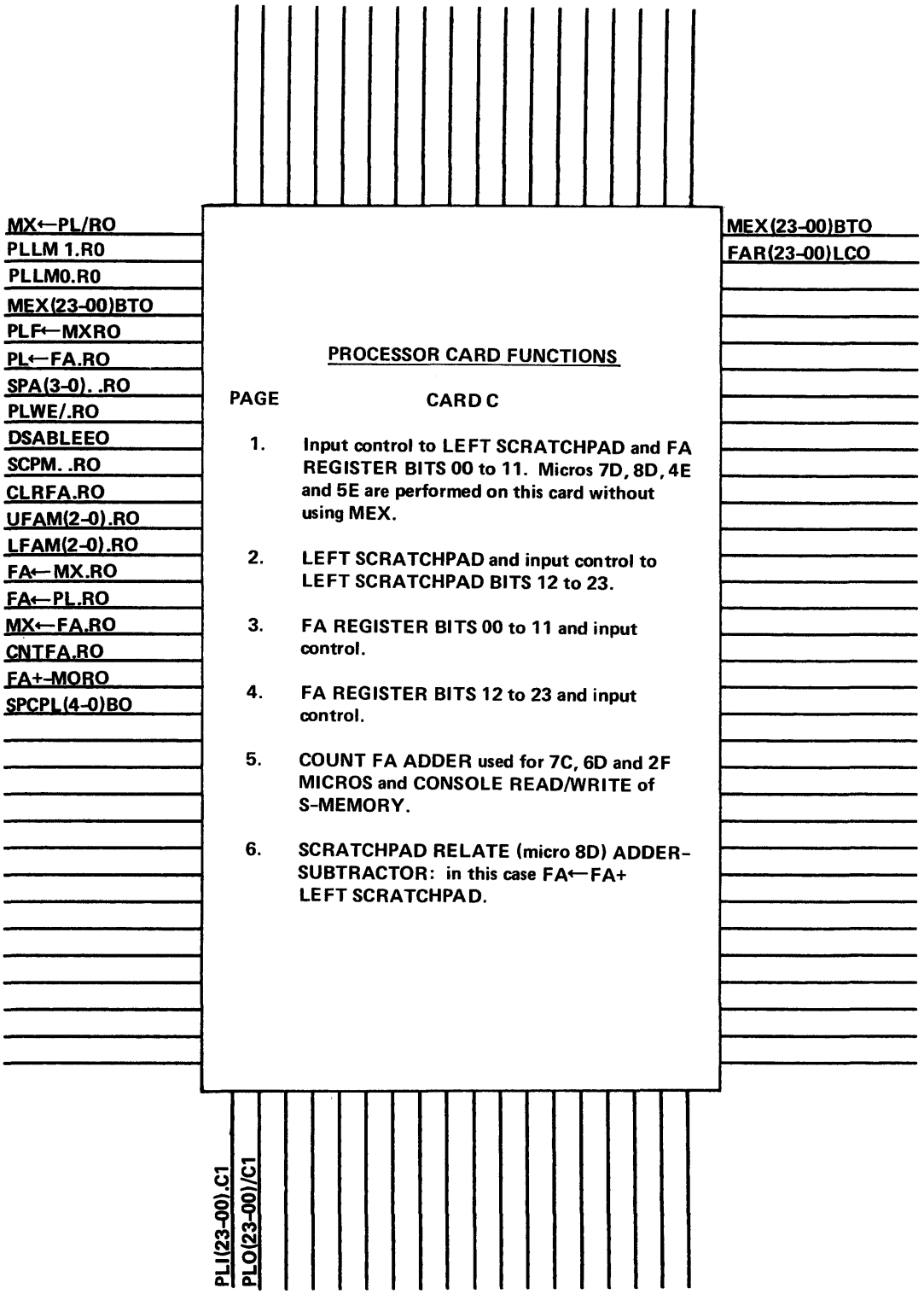


Figure 5-29. Processor Card C — Functions and Inputs/Outputs

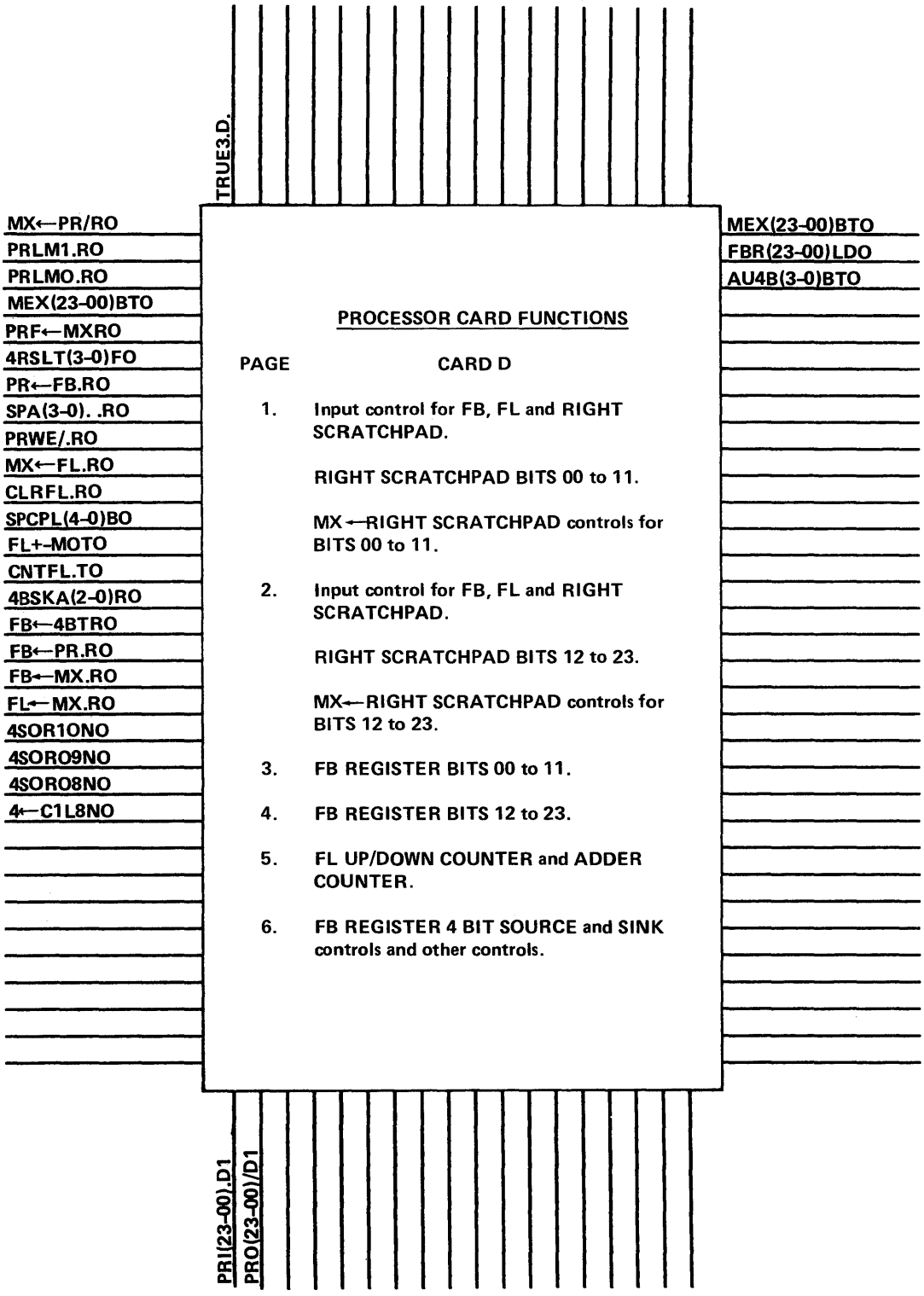


Figure 5-30. Processor Card D — Functions and Inputs/Outputs

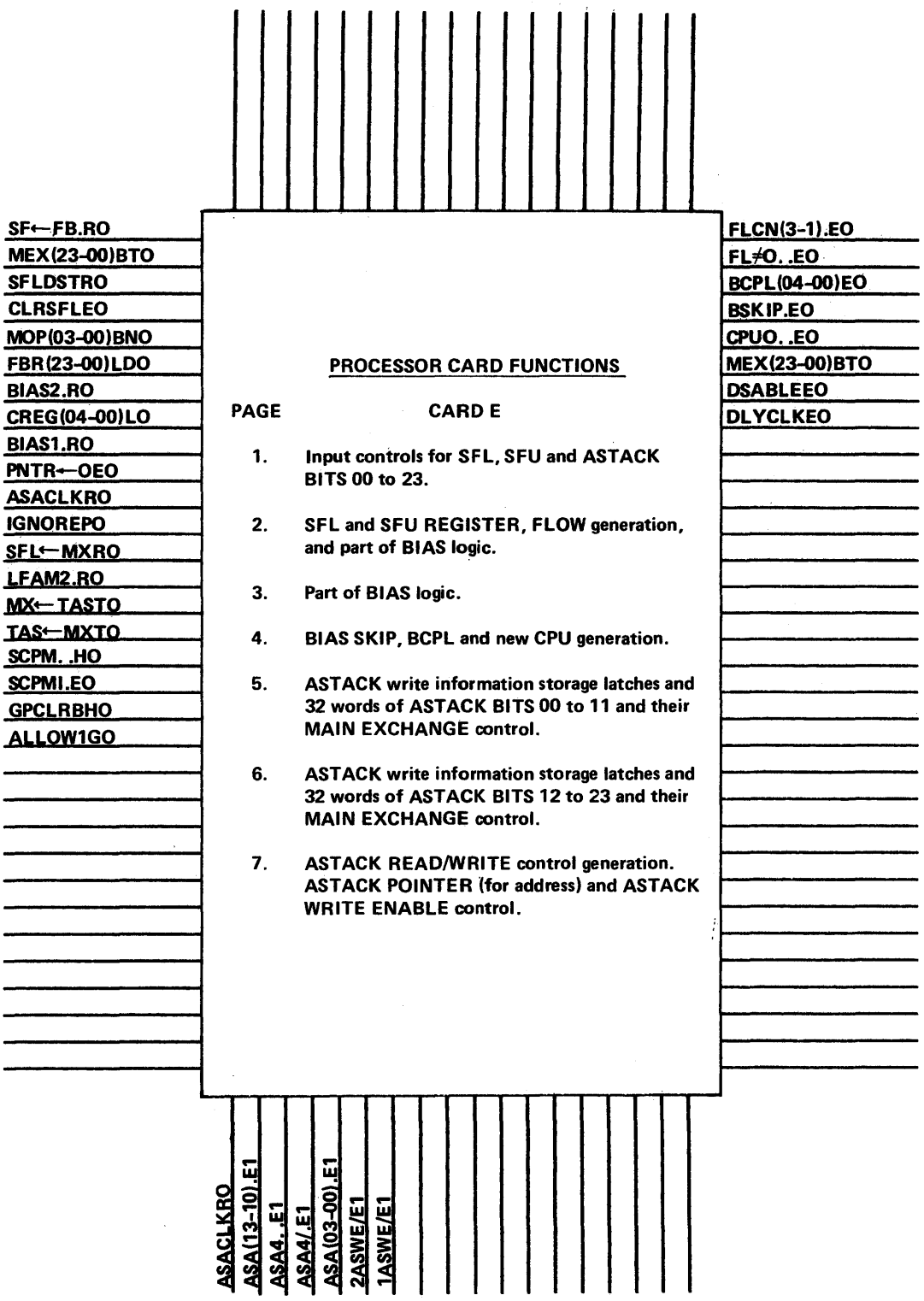


Figure 5-31. Processor Card E — Functions and Inputs/Outputs

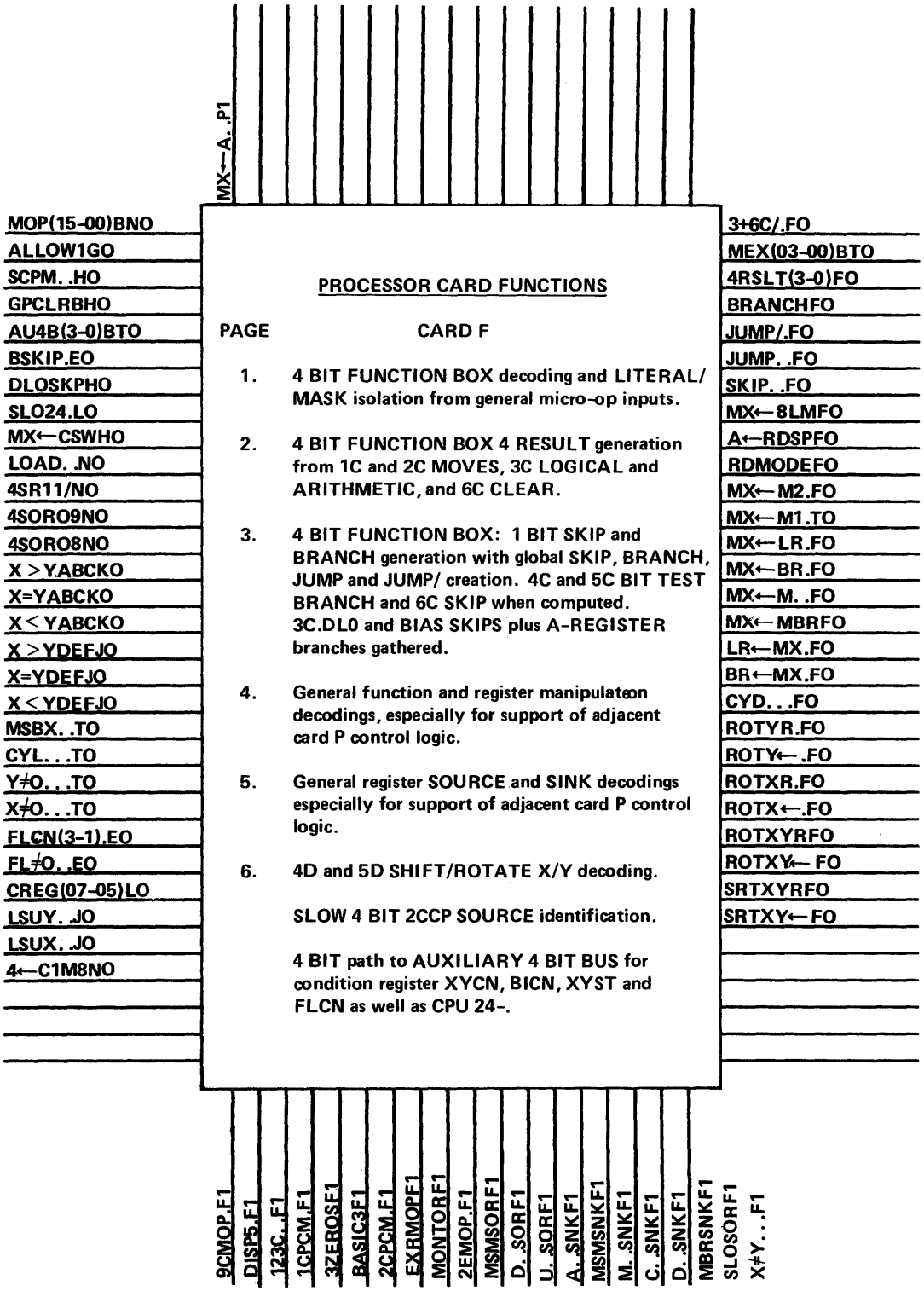


Figure 5-32. Processor Card F — Functions and Inputs/Outputs

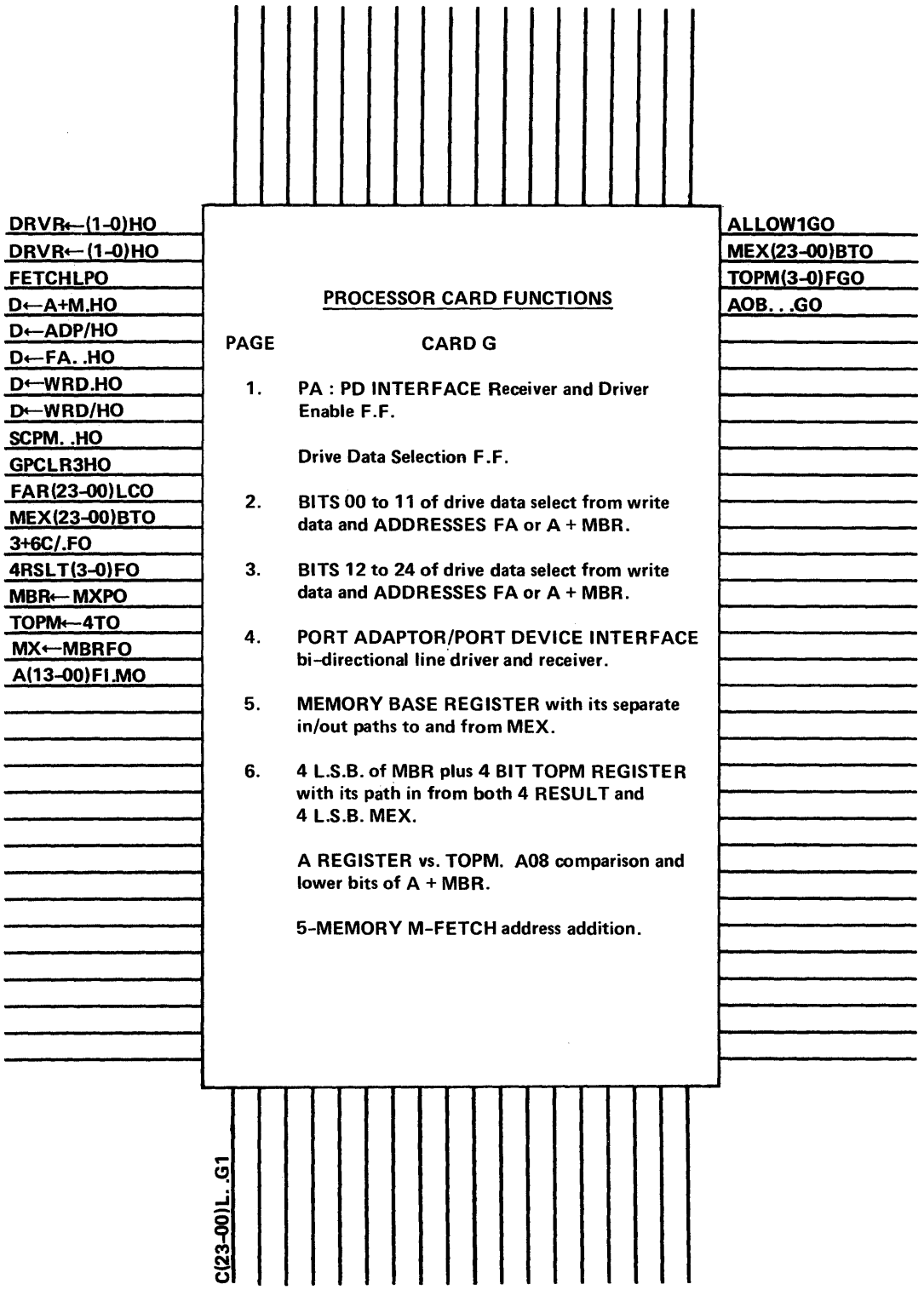
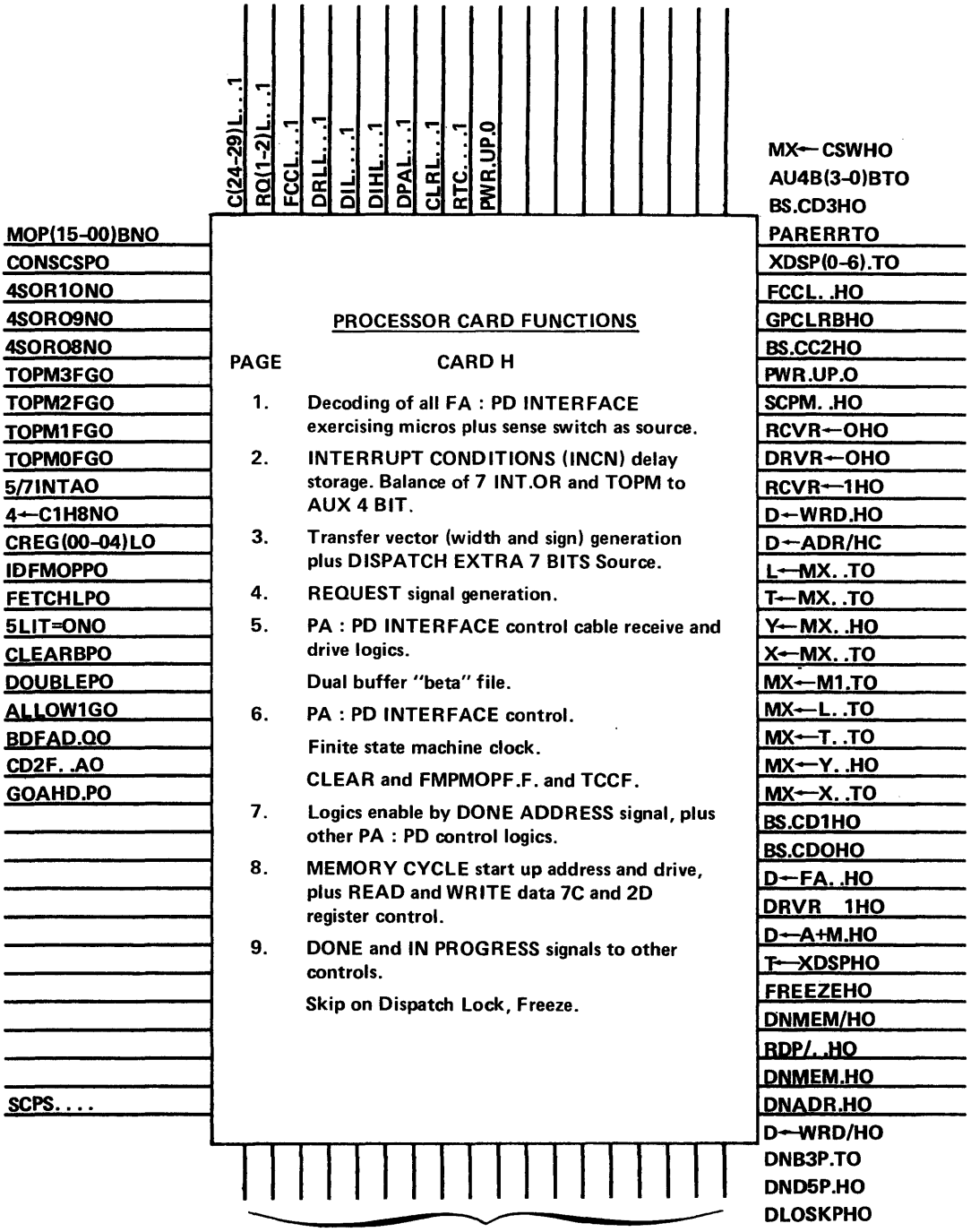


Figure 5-33. Processor Card G — Functions and Inputs/Outputs



See Figure 5-34 (Sheet 2 of 2)

Figure 5-34. Processor Card H — Functions and Inputs/Outputs (Sheet 1 of 2)

NS4L. .H1
NS2L. .H1
NS1L. .H1
TCCF. .H1
FMPMOPH1
CS(4-2-1)F. .H1
IDLECSH1
ADDRCSH1
READCSH1
SWAPCSH1
WRITCSH1
WAITCSH1
HOLDCSH1
NSSKIPH1
SKIPCSH1

NSHOLDH1
SSREADH1
SSSWAPH1
SSWRITH1
SSWAITH1
TCCF←1H1
TCCF←OH1
ENSCHGH1
NSIDLEH1
NSADDRH1
NSREADH1
NSWAITH1
CHG.CSH1

DATALOH1
DATAL1H1
NORSNKH1
DATAFOH1
DATAF1H1

STOPB3H1
DRLL. .H1
FCCL. .HO
MSCLR. .1
PDPL. . .1
TCCL. . .1
RQ(1-5)L. . .1
C(24-29)L. . .1
BSIC3.H1
BSWMOPH1
OVLMOPH1
CRDMOPH1
CWRMOPH1
DISP6.H1
BRDMOPH1
BWRMOPH1
BSIC3/H1
DS.INCH1
7INTOH1
TVRQ←OH1
TDF←1.H1
C←TD. .H1
C←BS.WH1
C←B3/WH1
DRQ5F.H1
CRO2F.H1

DSFAILH1
FAIL←OH1
FAILF.H1
B2NOGOH1
DNB3/PH1

Figure 5-34. Processor Card H — Functions and Inputs/Outputs (Sheet 2 of 2)

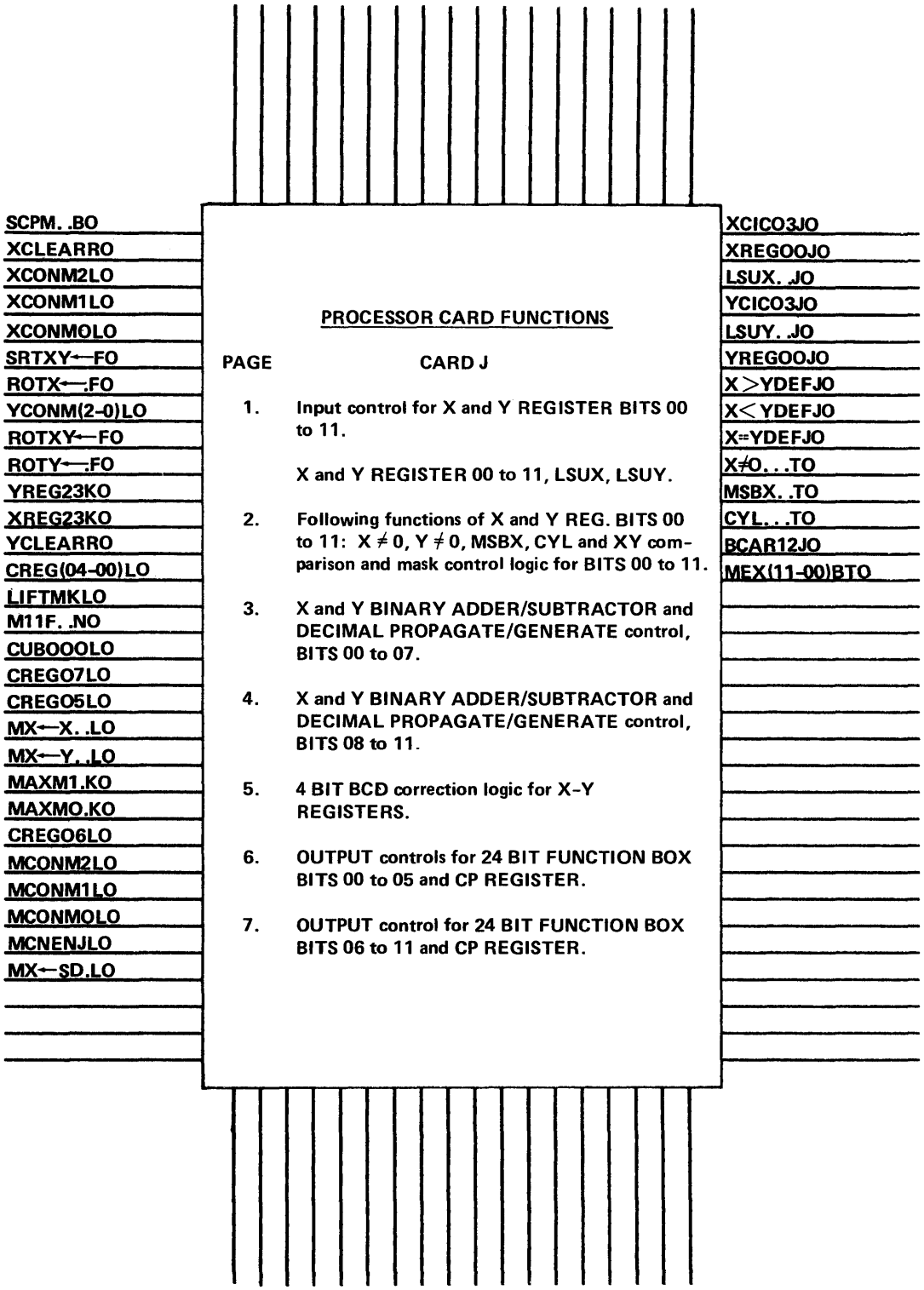


Figure 5-35. Processor Card J — Functions and Inputs/Outputs

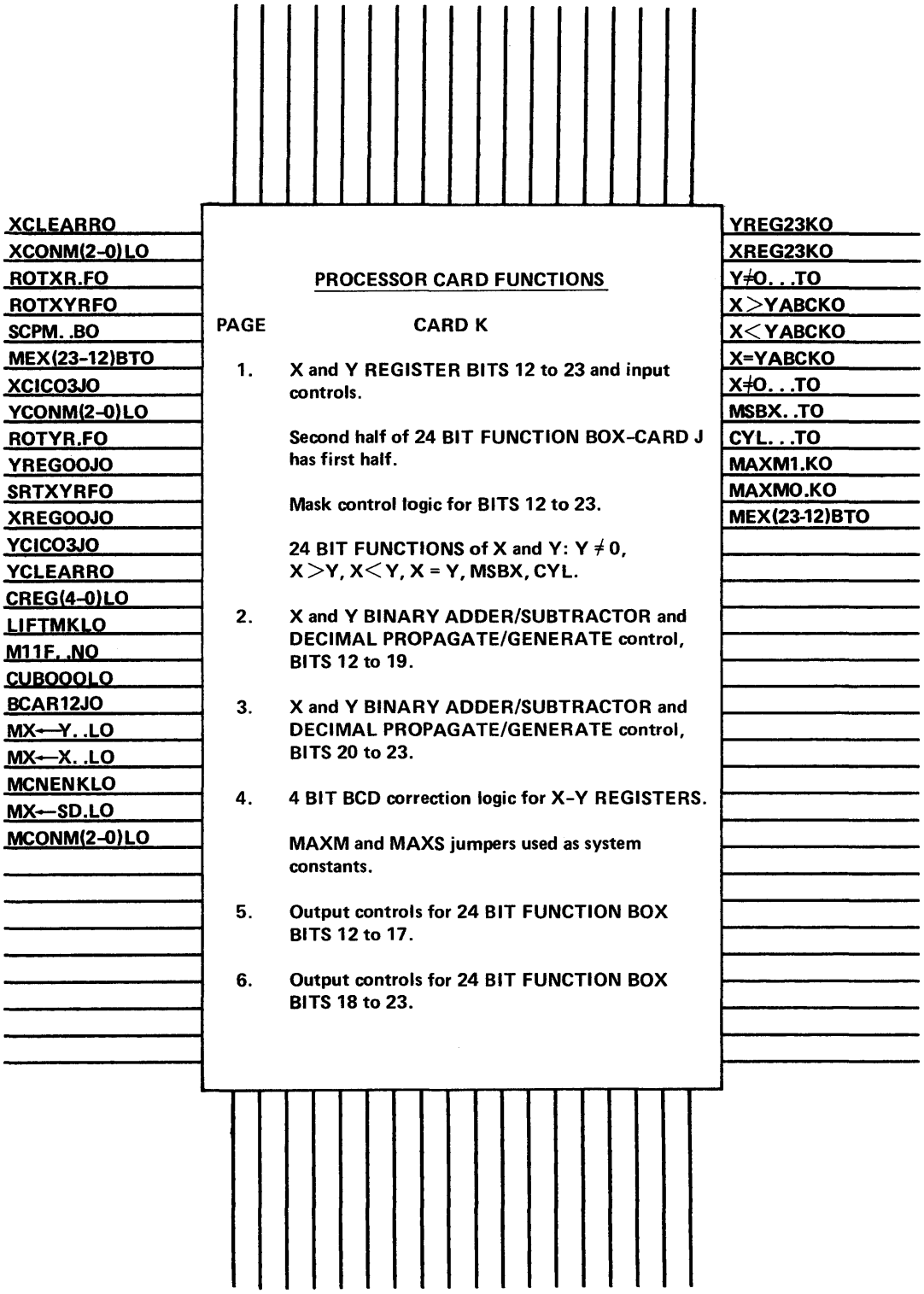


Figure 5-36. Processor Card K — Functions and Inputs/Outputs

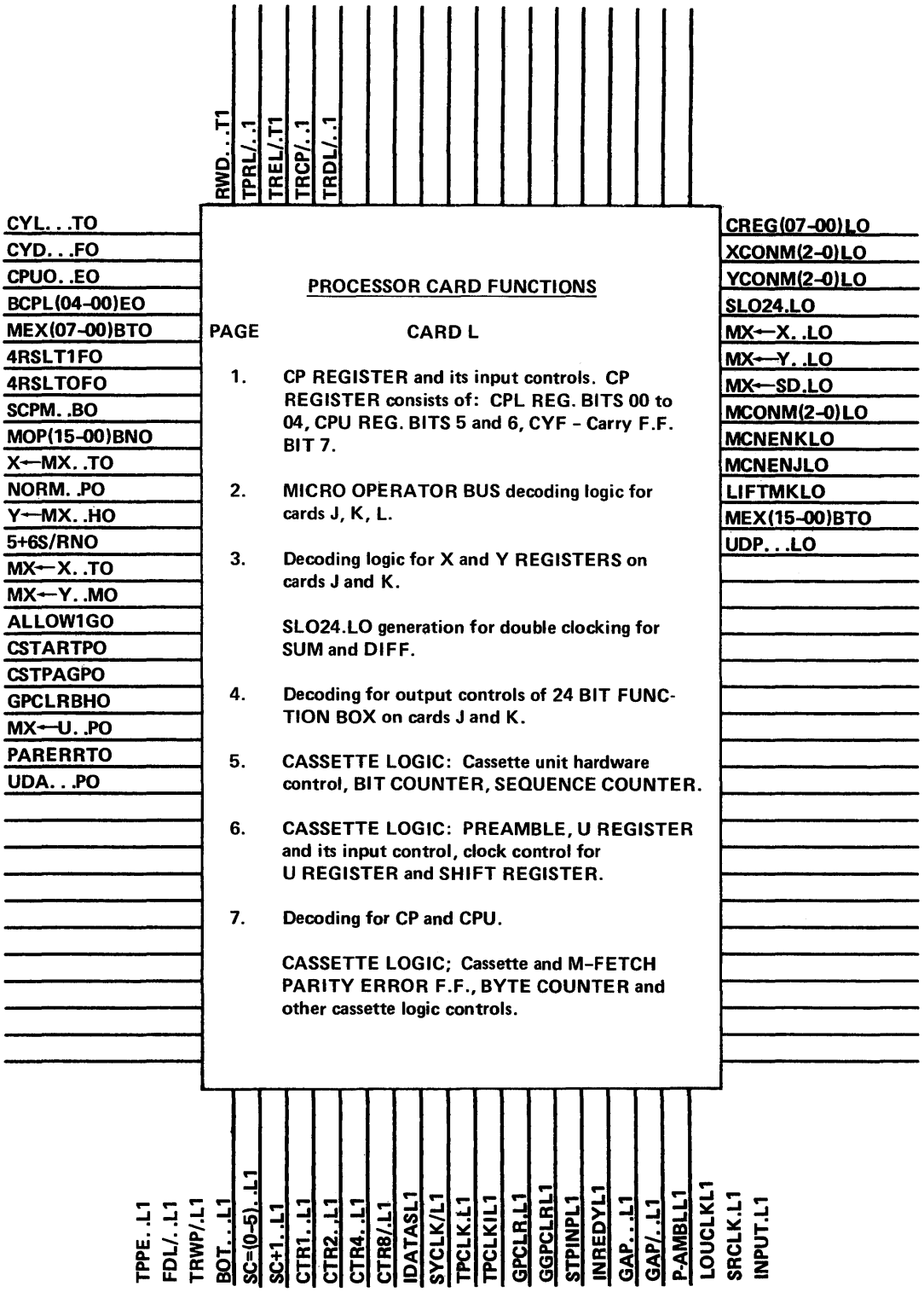


Figure 5-37. Processor Card L — Functions and Inputs/Outputs

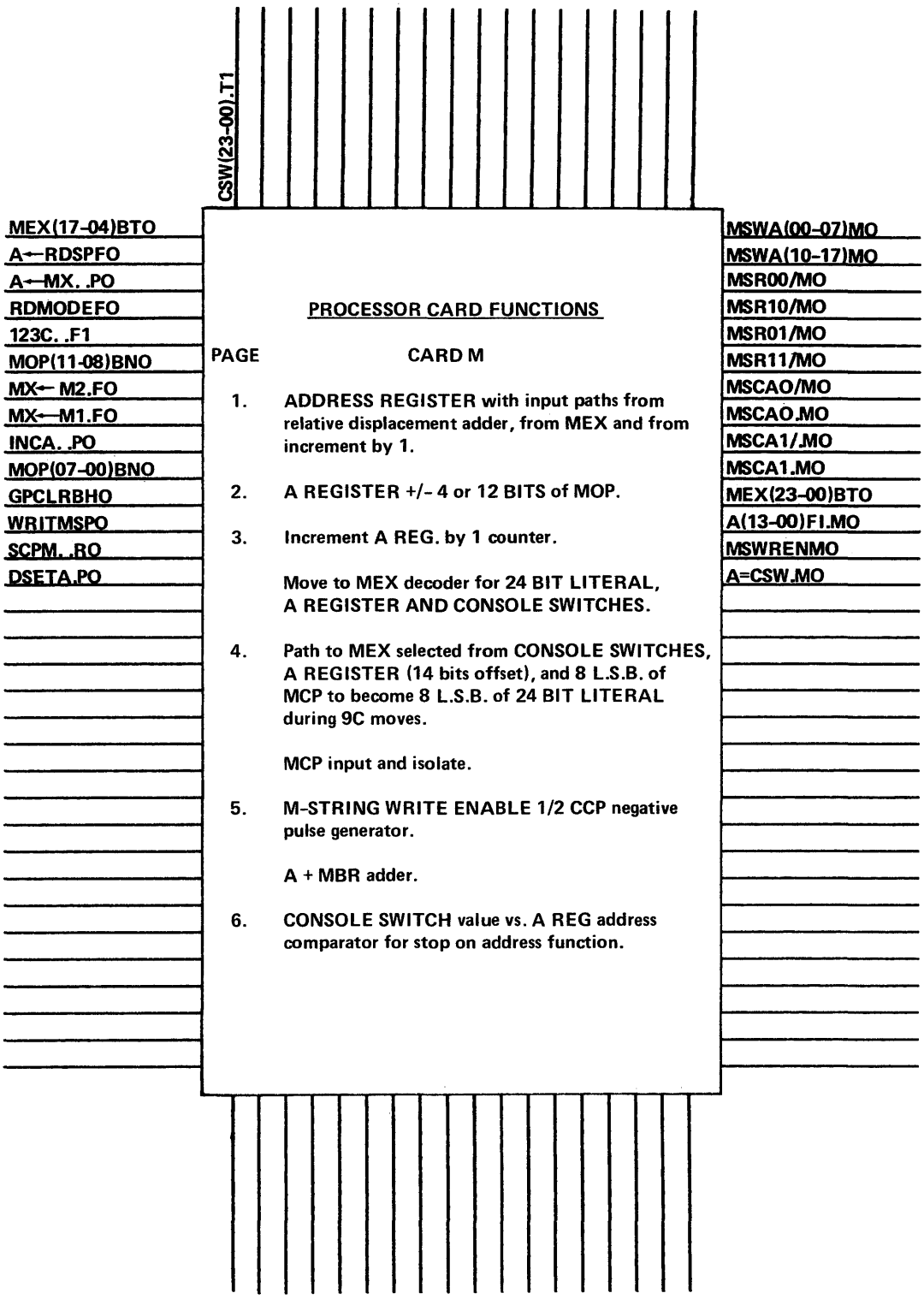


Figure 5-38. Processor Card M — Functions and Inputs/Outputs

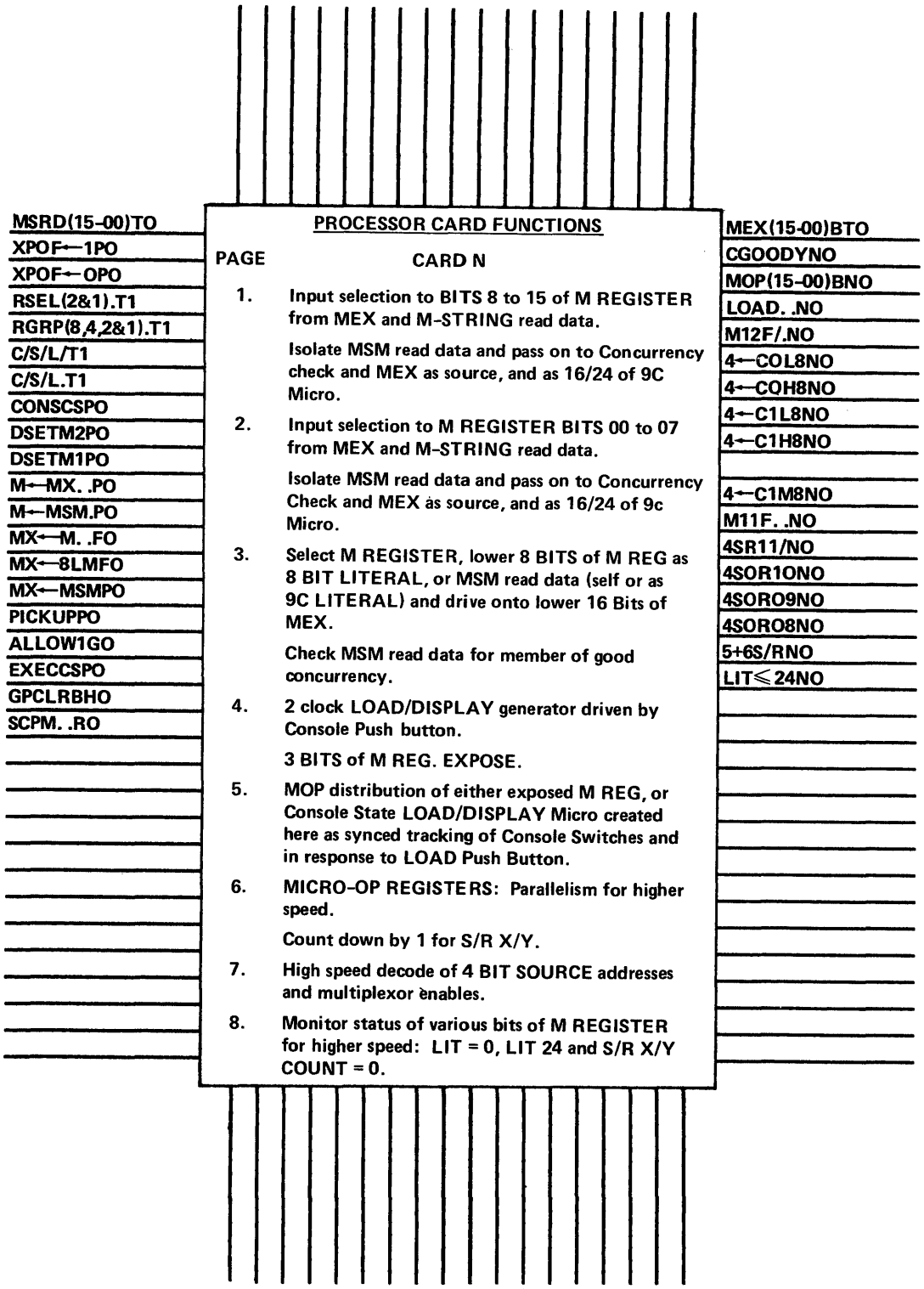


Figure 5-39. Processor Card N — Functions and Inputs/Outputs

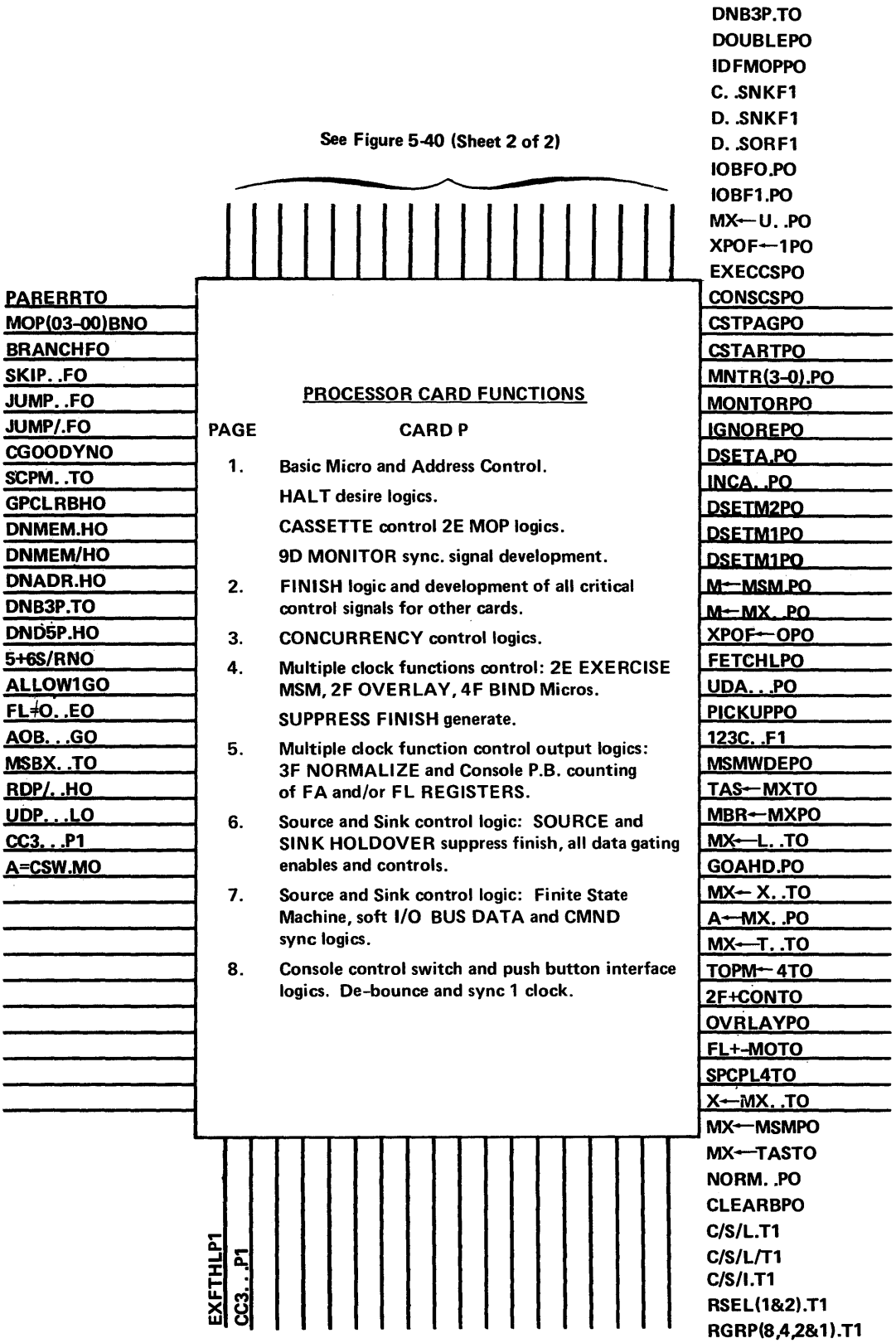


Figure 5-40. Processor Card P — Functions and Inputs/Outputs (Sheet 1 of 2)

A. .SNKF1
C. .SNKF1
EXRMOPF1
DISP5.F1
2EMOP.F1
USOR. .F1
MSMSNKF1
9CMOP.F1
D. .SNKF1
D. .SORF1
SLOSORF1
MSMSORF1
MBRSNKF1
M. .SNKF1
BASIC3F1
1CPCM.F1
2CPCM.F1
123C. .F1
3ZEROSF1
MONTORF1

X≠Y. . .F1
C/S/G.T1
C/S/G/T1
C/S/C.T1
C/S/CT1
C/S/L.T1
C/S/L/T1
C/S/I.T1
C/S/I/T1
C/S/H.T1
C/S/H/T1
C/S/S.T1
C/S/S/T1
RGRP(1,2,3,8).T1
SNGMODT1
RUNMODT1
RSEL(1,2).T1
INCA/.T1
INCA. .T1

Figure 5-40. Processor Card P — Functions and Inputs/Outputs (Sheet 2 of 2)

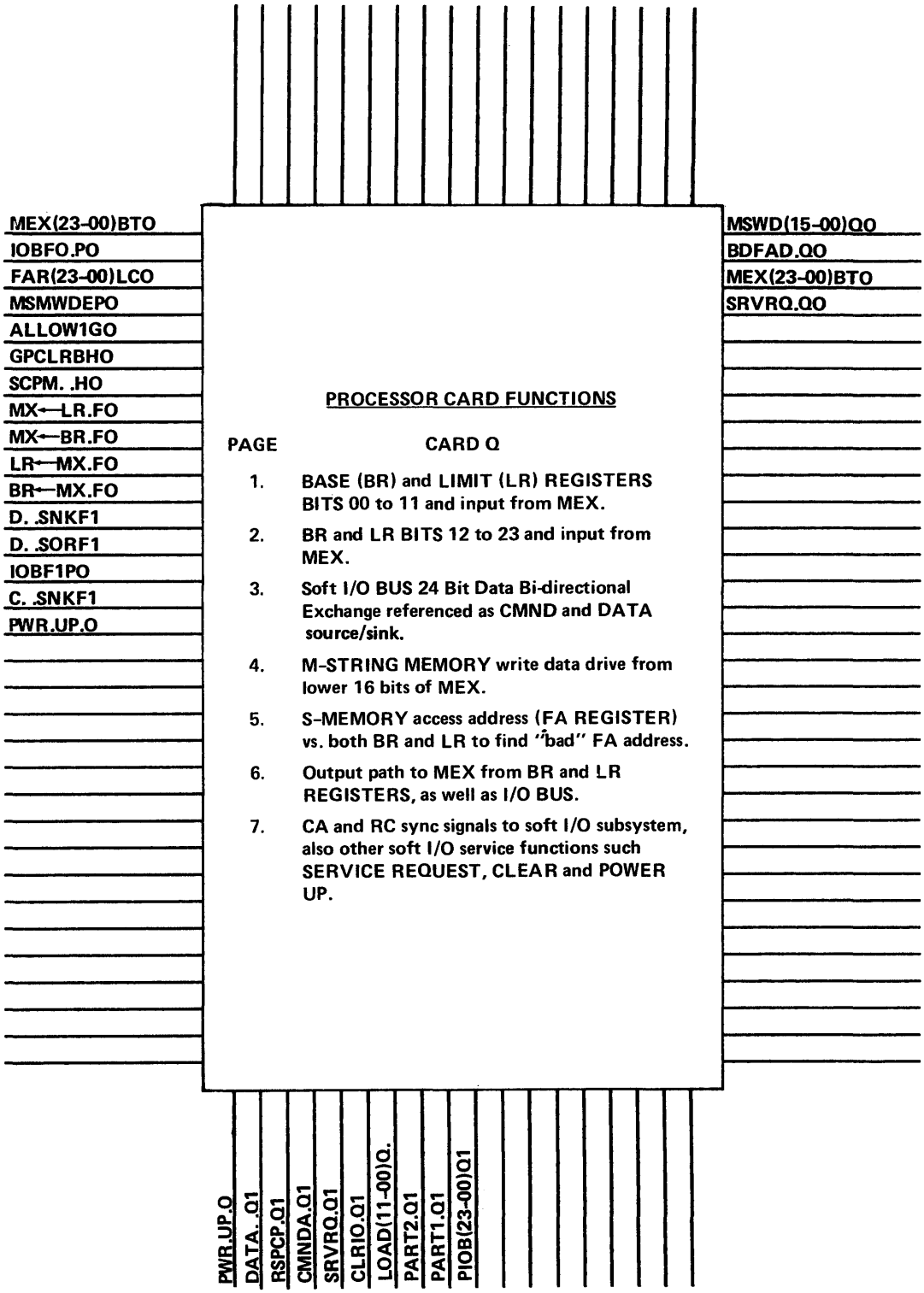


Figure 5-41. Processor Card Q — Functions and Inputs/Outputs

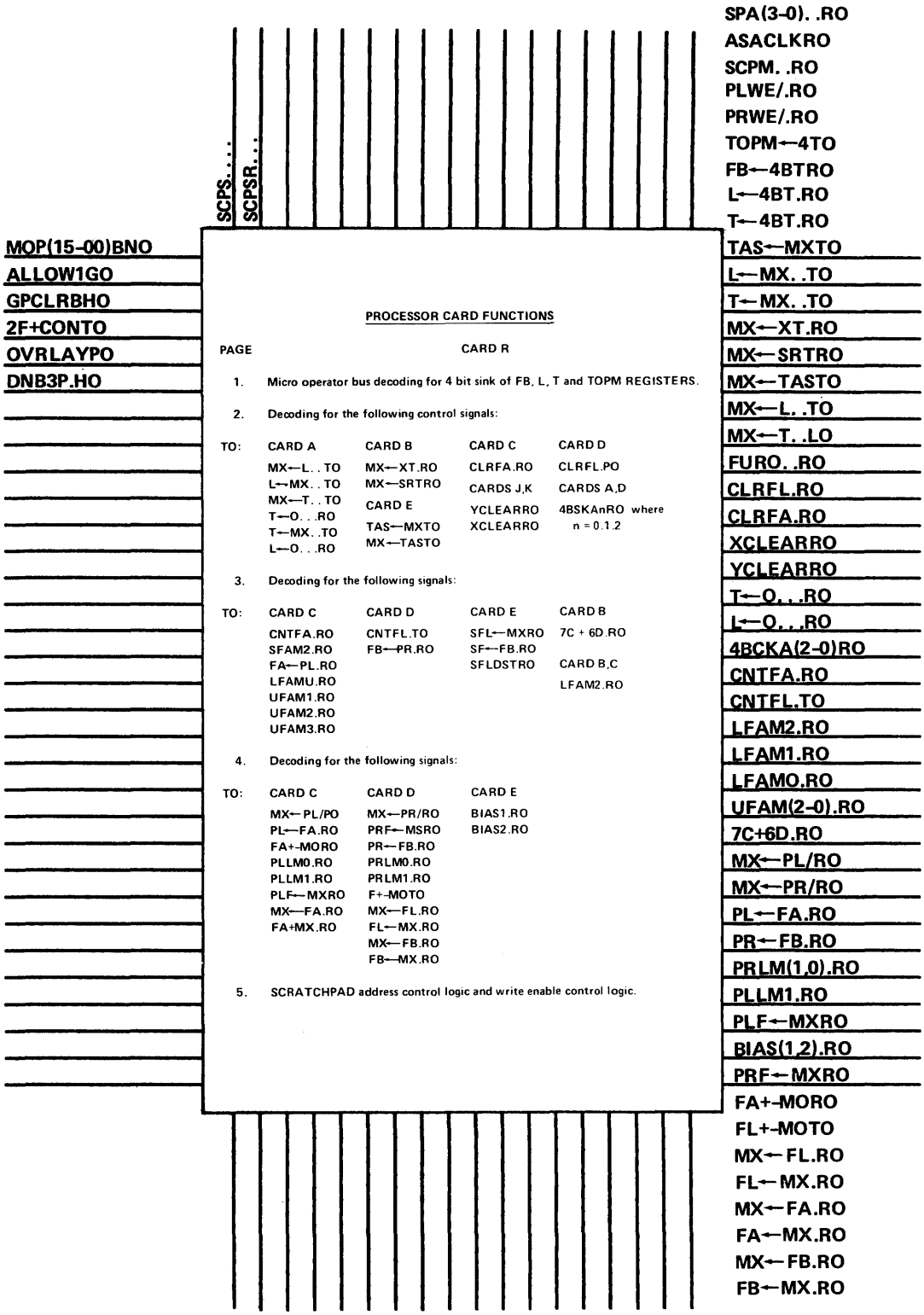


Figure 5-42. Processor Card R — Functions and Inputs/Outputs

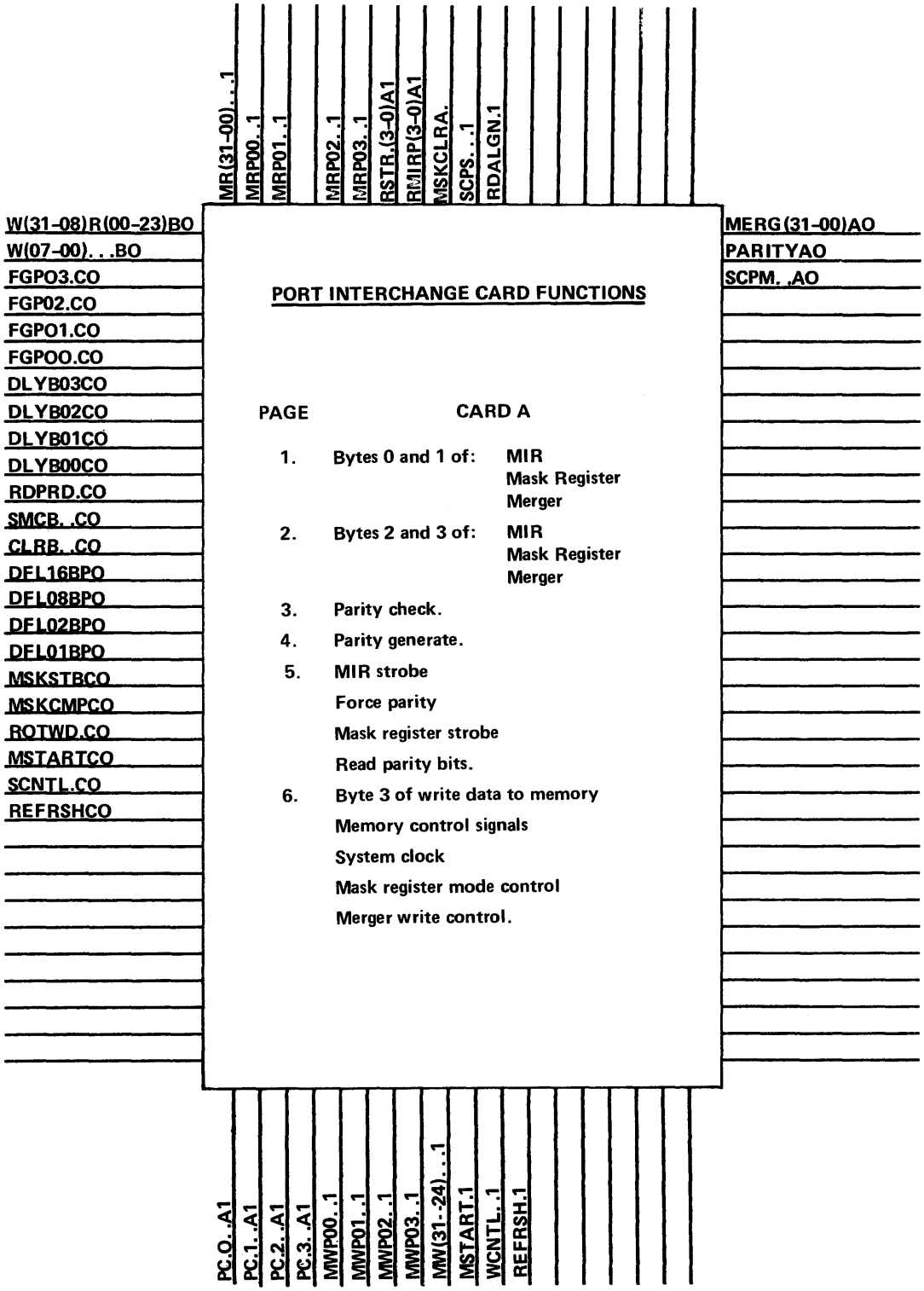


Figure 5-43. Port-Interchange Card A — Functions and Inputs/Outputs

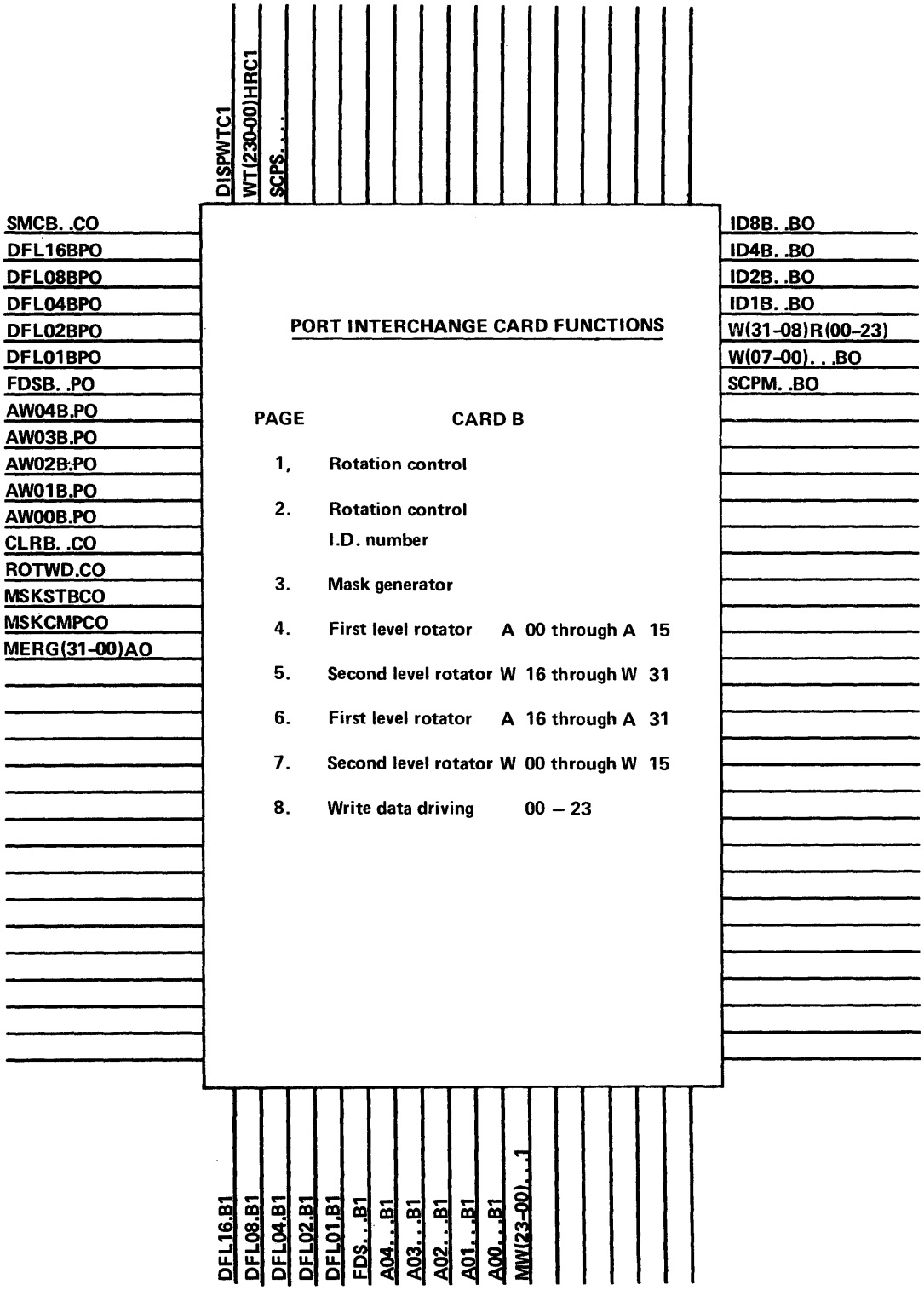


Figure 5-44. Port-Interchange Card B – Functions and Inputs/Outputs

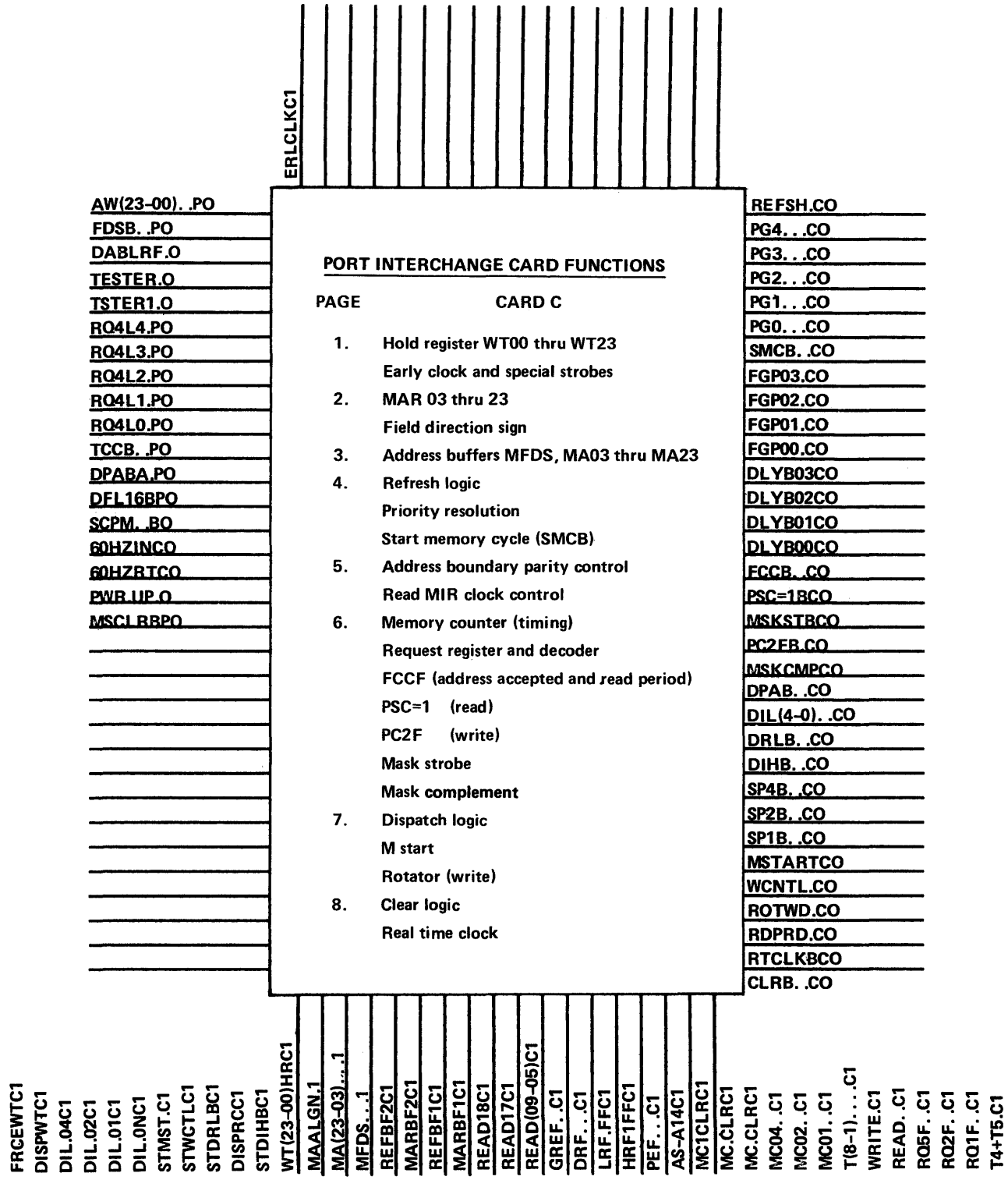


Figure 5-45. Port-Interchange Card C — Functions and Inputs/Outputs

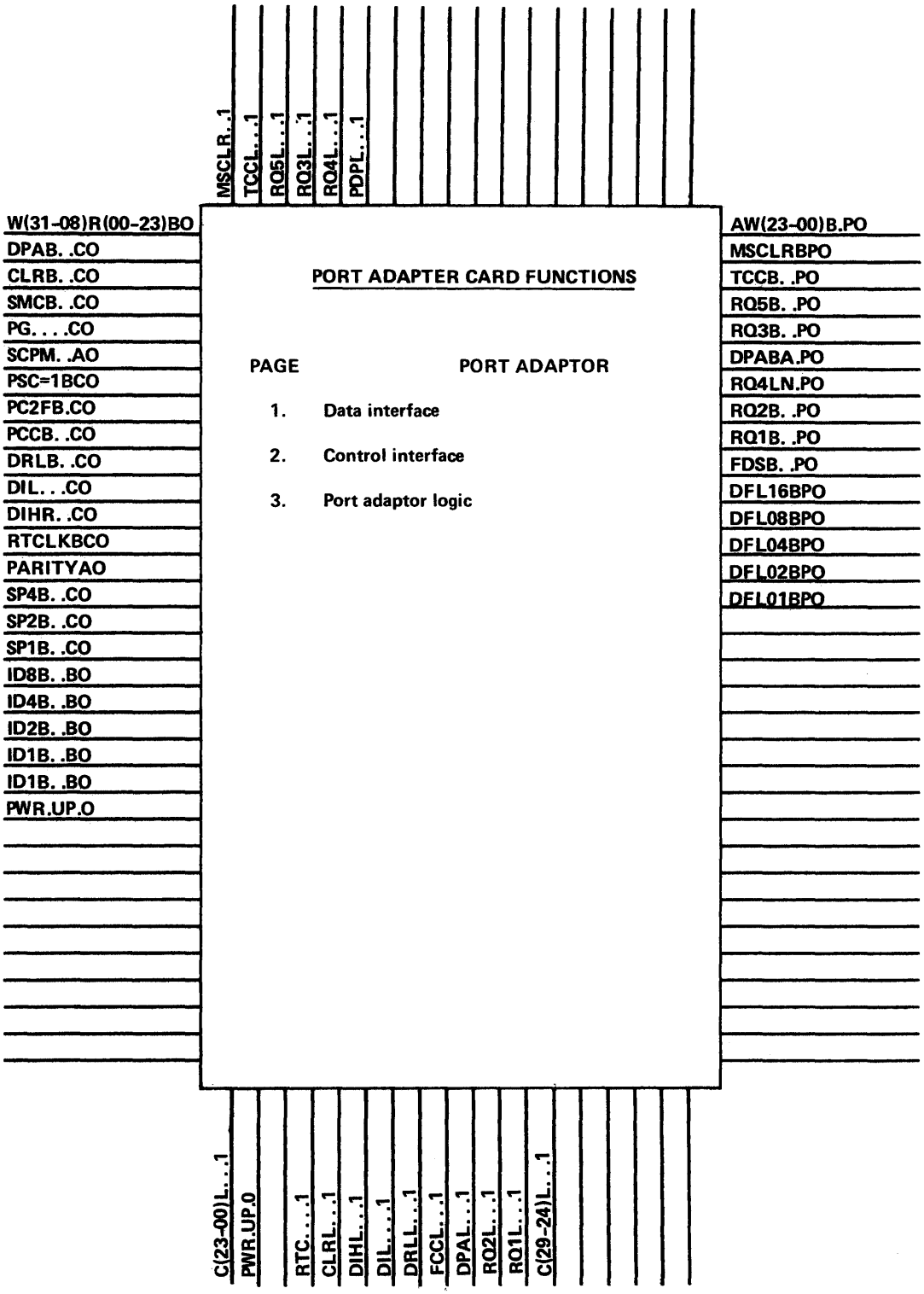


Figure 5-46. Port Adapter Card — Functions and Inputs/Outputs

SECTION 6

INSTALLATION

INTRODUCTION

This section describes the initial set-up and check-out procedures for the B 1720 Series Central System. The information contained herein is intended to amend and supplement the corresponding portions of the B 1700 Planning and Installation Manual, form 1053378, rather than to replace it. To install a system, proceed as directed in the following paragraphs, referring to that manual when so specified.

The following procedures assume that proper site planning to include environmental, physical, and electrical considerations has been carried out prior to the arrival of the equipment. These subjects are discussed in the planning and installation manual.

PHYSICAL PREPARATIONS FOR OPERATION

Physical preparation of the central system involves unpacking, mechanical assembly, and making the necessary electrical (power) connections. Each consideration is discussed in turn.

UNPACKING

The central system is prepared for shipment by covering damage-prone areas of the cabinet (such as the control panel) with foam rubber pads, then enclosing the entire unit in a protective plastic wrap. The cabinet is then positioned on a cushioned pallet, and secured with straps. This assembly is then crated, and attached to a forklift pallet.

Unpacking the B 1720 Series Central System is similar to the procedure for the B 1710 Series, which is described in the planning and installation manual. Refer to that section for further information.

MECHANICAL ASSEMBLY

The central system, except for the operator's table, is shipped in an assembled condition. Therefore, no assembly, other than attaching the table, is required. This step is described in the planning and installation manual, and should be performed when so directed. Proceed at this time with the sub-assembly checklist (following).

SUB-ASSEMBLY CHECKLIST

It is appropriate at this time that a check to ensure the presence of all necessary subassemblies be performed. Check the following items in the order listed.

a. Inspect the processor frontplane to ensure that the logic cards are installed in their proper location. Refer to table 6-1 and refer also to figure 5-25 (Section 5).

Table 6-1. Processor Frontplane Card Locations

Description	Part Number	Card Position
Processor Card A	2204 0711	C 3/4
" " B	2204 0802	C 0/1
" " C	2204 0893	E 3/4
" " D	2204 0894	D 6/7
" " E	2204 1073	B 6/7
" " F	2204 1164	B 3/4
" " G	2204 1255	A 0/1
" " H	2204 1347	A 3/4
" " J	2204 1438	D 0/1
" " K	2204 1529	C 6/7
" " L	2204 1610	D 3/4
" " M	2204 1701	F 0/1
" " N	2204 1792	E 6/7
" " Q	2207 2938	A 6/7
M-Memory Card 0	2207 3845	F 3/4
" " 1	2207 3845	F 6/7
" " 2	2207 3845	G 0/1
" " 3	2207 3845	G 3/4
Note that only the amount of M-memory ordered is present.		

b. Inspect the port interchange/port adapter frontplane to ensure that all cards are present and installed in their proper locations. Refer to Table 6-2 and refer also to figure 5-25 (Section 5).

Table 6-2. Port Interchange/Port Adapter Card Locations

Description	Part Number	Card Position
Port Interchange Card A	2207 4066	A 6/7
Port Interchange Card B	2207 4140	A 3/4
Port Interchange Card C	2207 4223	A 0/1
Port Adapter 0	2207 3498	B 0/1
Port Adapter 1	2207 3498	B 3/4
Port Adapter 2	2207 3498	B 6/7
Note that only the number of port adapters ordered is present.		

c. Inspect the S-memory frontplane(s) to ensure that the storage and control cards are installed in their proper locations. Each memory unit appears as listed in table 6-3. Note that the top or highest memory unit may or may not be equipped with a full complement of card groups. Likewise, the highest numbered card group of the top unit may be equipped with half-populated cards, depending on the amount of memory ordered. See also figure 5-25 (Section 5).

Table 6-3. S-Memory Unit Card Locations

Description	Part Number	Card Position
S-Memory FA Card	2358 4249	B 6/7
S-Memory 1C Card	2538 4223	B 3/4
S-Memory Cards:	*	
Group 0		A 0/1 - D 0/1
Group 1		A 3/4 - C 7/6
Group 2		A 6/7 - C 3/4
Group 3		B 0/1 - C 0/1
*The highest used card group (in entire memory) may contain fully-populated cards [1675 1562 (Rev. A), 1674 4054 (Rev. B) or 2539 7159 (Rev. D)]; or half-populated cards [1673 5086 (Rev. A), 2538 4009 (Rev. B) or 2539 7134 (Rev. D)].		

d. Check to ensure that all frontplane cables are installed in their proper locations. Refer to table 6-4 and refer also to figure 5-25 (Section 5).

Table 6-4. Frontplane Cable Chart

Cable Ref No.	Part No.	Type	From	To
			Processor	
C.1	2207 0213	Clock Coax	Clock Mod-Early Clk #1	M-Proc. "R" X $ Doghouse
C.2	2207 0213	Clock Coax	Clock Mod-System Clk #1	M-Proc. "R" Doghouse
C.3	2207 0213	Clock Coax	Clock Mod-System Clk #2	M-Proc. "B" Doghouse
C.4	2207 0213	Clock Coax	Clock Mod-System Clk #3	M-Proc. "H" Doghouse
C.5	2209 5681	Logic Cable	Control Panel - J2	M-Proc. "M" X #
C.6	2209 5681	Logic Cable	Control Panel - J4	M-Proc. "L" Y $
C.7	2209 5681	Logic Cable	Control Panel - J1	M-Proc. "A" X #
C.8	2207 1989	Jumper Cable	M-Proc. "F" X #	M-Proc. "P" X #
C.9	2206 3416	Flat 16	M-Proc. "Q" Chip Socket B0	I/O Distr Chip Socket J0
C.10	2207 1948	Logic Cable	M-Proc. "Q" Y #	I/O Distr X #
C.11	2207 7150	Logic Cable	M-Proc. "H" Y #	Port Adapter Y #
C.12	2207 1948	Logic Cable	M-Proc. "G" X #	Port Adapter X #
C.13	2209 5681	Logic Cable	Control Panel: J3	M-Proc. "P" X $
			Port Interchange	
C.14	2207 0213	Clock Coax	Clock Mod-System Clk #4	Port Interchg "A" Doghouse
C.15	2207 0213	Clock Coax	Clock Mod-System Clk #5	Port Interchg "B" Doghouse
C.16	2207 0213	Clock Coax	Clock Mod-Early Clk #2	Port Interchg "C" Doghouse
C.17	2207 1989	Logic Cable	Port Interchg "B" X #	Port Interchg "C" X #
C.18	2209 9642	Logic Cable	Port Interchg "A" X #	Field Address "Y" $
C.19	2209 9659	Logic Cable	Port Interchg "A" Y $	Interface Cont "X" $
C.20	2209 9634	Logic Cable	Port Interchg "A" Y #	Interface Cont "X" #
C.21	2209 9667	Logic Cable	Port Interchg "B" Y #	Interface Cont "Y" $
C.22	2209 9675	Logic Cable	Port Interchg "C" X $	Field Address "X" $
			S-Memory Base	
C.23	2207 0213	Clock Coax	Clock Mod-Adj #1	Field Address Doghouse
C.24	2207 1989	Jumper Cable	Field Address "Y" #	Interface Cont "Y" #
TP	Front Plane Test Points - Shown for Reference Only			

e. Check to ensure that all mechanical connections within the cabinet are secure, and that there are no loose wires, components, or other obvious faults in the physical integrity of the unit.

ELECTRICAL (POWER) CONNECTIONS

Refer to the B 1700 Planning and Installation Field Engineering Technical Manual, form number 1053378 for instructions on electrical (power) connections.

CENTRAL SYSTEM OPERATIONAL CHECKOUT

When physical installation of the central system has been completed, and connection has been made to the electrical power source, an operational checkout should be made. This insures that the unit performs as expected. The operational checkout procedure consists of the following phases:

Static Tests (power off)

Static Tests (power on)

Dynamic Tests

These procedures should be performed in the order listed.

Note that the central system operational checkout concerns the processor, M-memory, port interchange, and S-memory only. Therefore, do not connect or attempt to utilize any peripheral devices until the checkout is completed.

STATIC TESTS (POWER OFF)

These tests are to ensure that the central system does not fail due to either short or open circuits.

a. Make sure that both the system and source circuit breakers are in the OFF position.

b. Using a Tripplett 630 VOM or equivalent, make sure that a measurable amount of resistance exists between the +4.75, -2.0, +12, and -12 volt logic supply outputs and ground. See figure 4-29 (Section 4) for test points.

c. Check to ensure that a measurable amount of resistance is present between each of the logic supply outputs mentioned in step b.

d. Check for continuity between the logic power supply outputs and the backplane pins listed in table 6-5.

Table 6-5. Logic Power Continuity Test

Voltage	Backplane Pins	Cards
+4.75	0AX and 1AX	All Logic Cards
-2.00	0ZY and 1ZY	All Logic Cards
+12.00 -12.00	Check at small bus bars on right side of backplane.	

e. Check the memory power supply or supplies to ensure that a measurable amount of resistance exists between logic ground and the +19.0, +23.0, and -5.0 volt supply outputs.

f. Check for continuity between the memory supply outputs and the backplane pins listed in table 6-6. Note that each memory supply is associated with a single memory unit. Make sure that the memory backplane and supply being tested correspond. Perform the same test for all memory supply/memory unit groups.

Table 6-6. Memory Power Continuity Test

Voltage	Backplane Pin(s)	Card(s)
+19.0	0AY	All Memory Cards*
+23.0	1AY	" " "
-5.0	0ZX	" " "
*Of unit being tested.		

STATIC TESTS (POWER ON)

These tests are performed to ensure that the basic conditions for proper system operation are present, and that those parameters which are adjustable are set to the proper values. The power on static tests involve three major areas within the system:

a. Power supply tests

b. Clock circuit tests

c. Console tests

Powering Up

Prior to proceeding with the three areas mentioned, apply power to the system as follows:

a. Place both the source and system circuit breakers in the ON position.

b. Depress the POWER switch on the console. Successful application of power is evidenced by the illumination of the ON lamp, and operation of the ventilating fans. Also, some of the 24 LED console lamps may be illuminated in a random pattern. If the attempted application of power fails, refer to Section 5 (Maintenance) of this manual.

Power Supply Tests

When the system has been successfully powered up, check the output voltages of the logic and memory power supplies. The proper voltages are shown in table 6-6. These supply voltages are measured on the backplane at the voltage supply pins of the individual cards. See table 6-7. Refer to Section 3 (Adjustments) of this manual if any voltage is not within specifications. Refer to tables 6-5 and 6-6 for voltage measurement points.

Table 6-7. System Operating Voltages

Voltage Output	Nominal Voltage	Tolerance
+4.75	+4.85	±.010
-2.0	-2.08	±.010
+12.0	+12.00	±.10
-12.0	-12.00	±.10
+19.0	+18.75	±.05
+23.0	+3.60*	±.02
-5.0	-5.00	±.05
*Reference to +19 volt supply.		

Clock Circuit Tests

Perform the clock circuit adjustment procedure outlined in Section 4 (Adjustments) of this manual, taking corrective action only if the measured signals fail to meet specifications.

Console Tests

Before operation under program control is attempted, it is desirable to verify that all operational functions and capabilities of the central system logic are working correctly and accessible. This is done by manipulation of the console controls, a means by which the operator may manually load data, implement the commands of micro operators, and view the results. For further information on use of the console controls, refer to Section 1 of this manual. Using the test execution procedure below, perform the tests in the subsequent paragraphs in the order given.

Test Execution Procedure

a. Press the CLEAR pushbutton.

b. Place the REGISTER select rotary switch in the designated position.

c. Place the register group rotary switch in the designated position.

d. Set the 24 console switches to the required value (expressed in hexadecimal).

e. Press the LOAD pushbutton.

f. Follow the special instructions specified for the test.

Test 1: Verify the ability to load all registers in the register select 2 grouping. Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to X

c. Toggle switches = FFFFFF (all up)

d. Press the LOAD pushbutton.

Special Instructions:

After each depression of the load pushbutton rotate the register group rotary switch in a clockwise direction one position, until one revolution is completed. The test results are shown in table 6-8.

Table 6-8. Test 1 Results

Register Group	Indicator Lights
X	FFFFFF
Y	FFFFFF
T	FFFFFF
L	FFFFFF
A	-3FFF-
M	--FFFF
BR	FFFFFF
LR	FFFFFF
FA	FFFFFF
FB	FFFFFF
FL	--FFFF
TAS	No change (source only)
CP	----FF

Test 2: Verify the ability to change all registers in the register select 2 grouping. Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to X

c. Toggle switches = 000000 (all down)

d. Press the LOAD pushbutton

Special Instructions:

After each depression of the LOAD pushbutton, rotate the register group rotary switch one position in a clockwise direction until one revolution is completed. The test results are shown in table 6-9.

Table 6-9. Test 2 Results

Register Group	Indicator Lights
X	000000
Y	000000
T	000000
L	000000
A	-0000-
M	--0000
BR	000000
LR	000000
FA	000000
FB	000000
FL	--0000
TAS	No change (source only)
CP	----00

Test 3: Verify the ability to address the T- and L-registers in 4-bit increments Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to T

c. Toggle switches = ABCDEF (10,11,12,13,14,15) values

d. Press LOAD pushbutton

e. Register group rotary switch to L

f. Press LOAD pushbutton

Special Instructions:

a. Set the REGISTER select rotary switch to 0.

b. Rotate the register group rotary switch from TA through LF, disregarding the contents of CA and CB.

The rest results are shown in table 6-10.

Table 6-10. Test 3 Results

Register Group	Indicator Lights
TA	00000A
TB	00000B
TC	00000C
TD	00000D
TE	00000E
TF	00000F
LA	00000A
LB	00000B
LC	00000C
LD	00000D
LE	00000E
LF	00000F

Test 4: Verify the ability to address registers CA, CB, CC, and CD as either a four-bit sink or source. Set the following controls as indicated:

a. REGISTER select rotary switch to 0

b. Register group rotary switch to CA

c. Toggle switches = 00000F

d. Press the LOAD pushbutton

Special Instructions:

a. After each depression of the LOAD pushbutton, rotate the register group rotary switch clockwise to the next sub-register. Continue until all four registers have been loaded.

b. Return the register group rotary switch to the CA position.

c. Place all of the toggle switches in the off position (down). Repeat the above steps.

The test results are shown in table 6-11.

Table 6-11. Test 4 Results

Toggle Switches = 00000F	Toggle Switches = 000000
CA=00000F CB=00000F CC=00000F CD=00000F	CA=000000 CB=000000 CC=000004 CD=000000
Note: Real time clock interrupt sets CC register bit 2.	

Test 5: Verify the ability to address the FB register in four-bit increments. Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to FB

c. Toggle switches = ABCDEF

d. Press the LOAD pushbutton

Special Instructions:

a. Set the REGISTER select rotary switch to 1.

b. Rotate the register group rotary switch one position at a time from FU to FLF.

The test results are shown in table 6-12.

Table 6-12. Test 5 Results

Register Group	Indicator Lights
FU	A
FT	B
FLC	C
FLD	D
FLE	E
FLF	F

In tests 6 through 11 the X and Y conditions are checked. Performing these tests verifies that the basic logic of the 24-bit arithmetic unit is functioning properly.

Test 6: Verify that X = Y when X and Y are set to 0. Set the following controls as indicated:

a. X- and Y-registers to 0

b. REGISTER select rotary switch to 1

c. Register group rotary switch = XYCN (X and Y condition)

No further action is needed. The results appear on the indicator lights, and should be 000004 (X = Y, as indicated by bit 2 of XYCN register being true).

Test 7: Verify that X > Y when the contents of the X-register is greater than the Y-register. Set the following controls as indicated:

a. Y register to 0

b. REGISTER select rotary switch to 2

c. Register group rotary switch to X

d. Toggle switches = 000001

e. Press the LOAD pushbutton

f. REGISTER select rotary switch to 1

g. Register group rotary switch to XYCN

The results should be 000001 (X > Y).

Test 8: Verify that X < Y when the contents of the X-register is less than the Y-register. Set the following controls as indicated:

a. X-register to 0

b. REGISTER select rotary switch to 2

c. Register group rotary switch to Y

d. Toggle switches = 000001

e. Press the LOAD pushbutton

f. REGISTER select rotary switch to 1

g. Register group rotary switch to XYCN

The results should be 000002 (X < Y).

Test 9: Detect the most significant bit of X (MSBX). Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to X

c. Toggle switches = 800000

d. Press the LOAD pushbutton

e. Register group rotary switch to CP

f. Toggle switches = 000018

g. Press the LOAD pushbutton

h. REGISTER select rotary switch to 1

i. Register group rotary switch to XYCN

The results should be 000009 (MSBX and X > Y).

Test 10: Verify that the X-register not equal to zero ($X \neq 0$) and the least significant unit of X (LSUX) being true may be detected. Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to X

c. Toggle switches = 000001

d. Press the LOAD pushbutton

e. Register group rotary switch to Y

f. Toggle switches = 000000

g. Press the LOAD pushbutton

h. REGISTER select rotary switch to 1

i. Register group rotary switch to XYST

The results should be 00000D ($X \neq 0$, LSUX, and INTOR).

Test 11: Verify that Y-register not equal to zero ($Y \neq 0$) may be detected. Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to Y

c. Toggle switches = 000001

d. Press the CLEAR pushbutton, then the LOAD pushbutton

e. REGISTER select rotary switch to 1

f. Register group rotary switch to XYST

The results should be 000006 (Y ≠ 0 and INT OR).

Test 12: Verify the ability to address the CP and CPU register as a sink. Set the following controls as indicated:

a. Press CLEAR (zeros CP)

b. REGISTER select rotary switch to 1

c. Register group rotary switch to CPU

d. Toggle switches = 000003

e. Press the LOAD pushbutton

f. REGISTER select rotary switch to 2

g. Register group rotary switch to CP

Results should be 000060 (CPU is bits 5 and 6 of CP).

Tests 13, 14, and 15 check the ability to read and write into memory from the console.

Test 13: Verify that the A-register is incremented when the INC pushbutton is depressed. Set the following controls as indicated:

a. Press CLEAR pushbutton

b. REGISTER select rotary switch to 2

c. Register group rotary switch to MSM

d. Depress the INC pushbutton seven times (disregard any display seen at this time)

e. Register group rotary switch to A

Results should be 000070 (each increment of A is binary weight 16). The A-register may be incremented only when MSM is selected.

Test 14: Verify that 16 bits may be written into M-string memory at address zero (0). Set the following controls as indicated:

a. Press CLEAR pushbutton (this action clears A)

b. Register group rotary switch to MSM

c. Toggle switches = FFFFFF (write data)

d. Press the LOAD pushbutton. The result should be 00FFFF.

Test 15: Verify that the FA register is incremented when selected and the INC pushbutton is depressed. Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to FA

c. Toggle switches = 000000

d. Depress the LOAD pushbutton (this action clears FA)

e. Register group rotary switch = READ

f. Press the INC pushbutton once

g. Register group rotary switch to FA

Result should be 000010 (FA increments by binary weight 16 with each depression of INC). FA increments in either read or write register group.

Tests 16 and 17 may be done using any available memory location by setting the desired starting address with the console switches. This is done in step c in test 16. Different data configurations may be written by setting the 24 console switches to the desired patterns in step f of test 16.

Test 16: Verify that 24 bits may be written into S-memory at address zero (0). Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to FA

c. Toggle switches = 000000 (address)

d. Press the LOAD pushbutton (this selects address zero).

e. Register group rotary switch to WRITE

f. Toggle switches = FFFFFF (write data)

g. Press READ/WRITE pushbutton

Result should be 000000 (write data is not displayed).

Test 17: Read the 24 bits that have been written in memory (starting at memory address zero). Set the following controls as indicated:

a. REGISTER select rotary switch to 2

b. Register group rotary switch to READ

c. Toggle switches = 000000

d. Press READ/WRITE pushbutton

Result should be FFFFFF.

DYNAMIC TESTS

Once it has been determined that the basic central system functions may be implemented under manual control, tests under program control can begin. The dynamic tests consist of software routines which exercise the processor and memory with sequences of micro instructions. The tests are arranged to check out the various portions of the logic against known proper responses, and to indicate error conditions by way of halt interrupts, and bit patterns stored in various registers. Note that these same tests are also used for maintenance purposes.

There are four dynamic tests for proving the operational integrity of the central system:

a. Processor maintenance test routine (MTR)

b. Processor dynamic routine

c. M-memory dynamic routine

d. S-memory dynamic routine

Each of these programs is contained in a cassette tape, and is loaded into the machine by way of the console cassette tape reader. Some are executed directly from the tape, while others are loaded into M- or S-memory for execution. Since each is different, the directions and references contained in the accompanying documentation should be followed.

Perform each of the dynamic tests in the order listed above. Refer to Section 5 of this manual for further guidance if successful completion of the tests is not achieved. When a satisfactory central system is obtained, continue with the mechanical assembly of the central system and installation of the I/O subsystem as directed in Section 4 of the B 1700 Planning and Installation Manual, and the I/O Base Technical Manual.

MEMORY EXPANSION

The B 1720 Series Central System S-Memory may be of any desired size, from the minimum practical for reasonable system performance (about 32K bytes) up to a maximum of 256K bytes. S-memory is modular, and may be expanded in increments as small as 8K bytes.

In practical terms, S-memory is constructed from S-memory adapters (storage card groups) and memory base units (backplane assemblies). Each S-memory adapter consists of two storage cards, containing a maximum of 16K bytes storage. Four memory adapters may be installed in each memory base unit, for a storage capacity of 64K bytes. Total system memory may consist of up to four memory base units, each of which contains its own addressing and control logic, and is powered by a separate memory power supply.

S-MEMORY COMPONENTS

Increasing the size of S-memory requires that the appropriate number of memory adapters (two-card groups) be ordered. To allow flexibility in meeting user requirements, memory adapters are available in fully-populated (16K byte storage) and half-populated (8K) versions The reduced capacity adapters are for use only as the last increment (top) of memory, thus being restricted to one per system.

Since the memory base units may contain a maximum of four memory adapters (16K bytes each), the major physical divisions of memory occur at 64K byte intervals. Therefore, any proposed memory expansion which crosses the 64K, 128K, or 196K boundary requires the installation of an additional memory base unit. Each such unit includes a 10-card backplane with FA and IC cards, power cabling, and a memory power supply.

Order parts from table 6-13 as required when installing additional memory.

Table 6-13. Memory Parts Ordering Information

Part Number	Description	Notes
2539 7159	Memory Card, Fully-Populated (Rev. D)	Order in pairs (16K)
2534 7134	Memory Card, Half-Populated (Rev. D)	One pair only per system (8K)
2205 7418	Memory Base Unit	Includes backplane, FA * K cards
2210 9052	Memory Power Supply	Order one per memory base unit
2207 8521	Memory Voltage Cable	Order one per memory power supply
2207 5210	AC Power/Memory Power Cable #1	Order one per memory power supply
2209 7679	AC Power Extension	Order one per memory power supply
2208 2770	Memory Cable Kit #2	Order when installing second memory unit
2210 7031	Coaxial Cable	Order with memory cable kit #2
2208 2788	Memory Cable Kit #3	Order when installing third memory unit
2211 2510	Coaxial Cable	Order with memory cable kit #3
2208 2796	Memory Cable Kit #4	Order when installing fourth memory unit
2211 2528	Coaxial Cable	Order with memory cable kit #4

NOTES: Rev. A Fully populated S-memory cards are P/N 1675 1562
Rev. B Fully populated S-memory cards are P/N 1674 4054
Rev. A Half populated S-memory cards are P/N 1673 5086
Rev. B Half populated S-memory cards are P/N 2538 4009

NOTE

Memory storage cards are available in two versions which cannot be mixed. Those listed in table 6-13 are known as revision D type, and are used in current production. The older revision B cards are found on many previously installed systems, and may be identified (and ordered) by part numbers 1674 4054 (16K) or 2538 4009 (8K). Note that the memory voltages employed by the two versions differ somewhat.

INSTALLATION PROCEDURE

Install additional memory as directed in the following paragraphs. Note that the procedure differs depending on whether or not an additional memory base unit is required.

Memory Adapter Installation

If the memory adapters to be added can be contained within the highest existing memory base unit, installation consists simply of inserting the adapters in the appropriate slots, and making the necessary jumper chip adjustments. Install the adapters as shown in figure 6-1, proceeding from lowest to highest card position. Note that when a half-populated (8K) adapter is used, it must be in the highest-numbered group position which is occupied.

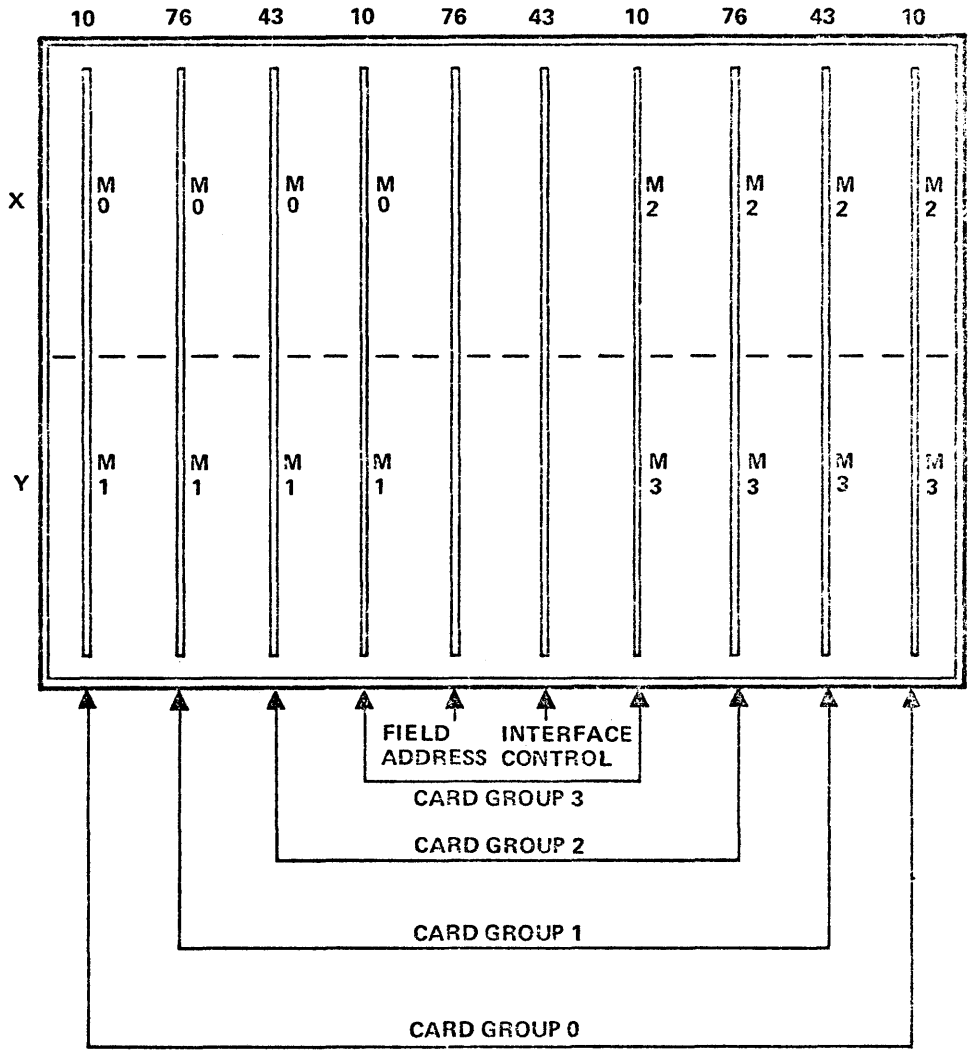


Figure 6-1. The 64 K-Byte S-Memory Unit

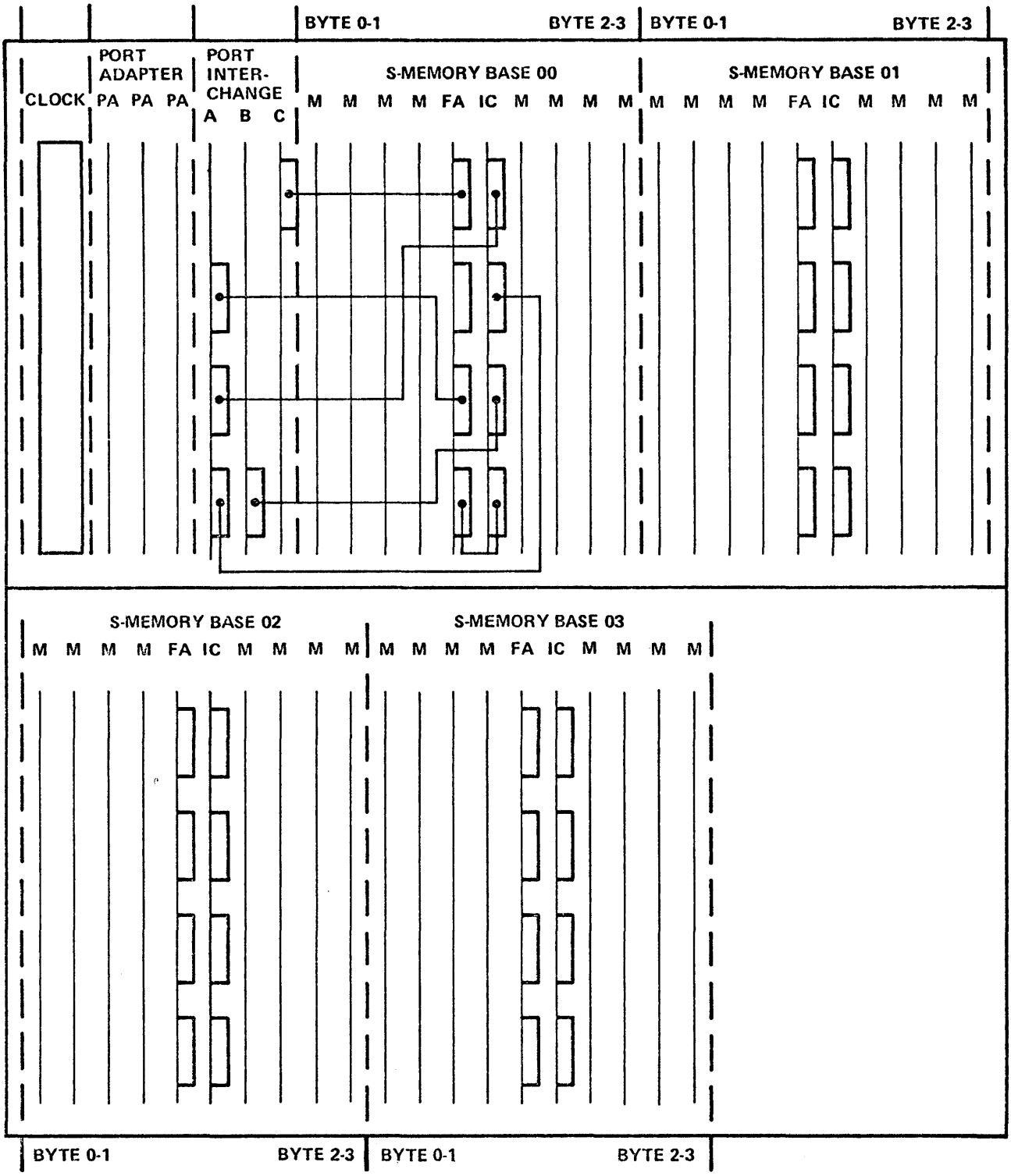


Figure 6-2. Memory Unit Location

Memory Base Backplane Installation

When it is necessary to install an additional memory base unit, the new backplane assembly should be located in the lowest (numerical) unoccupied position as shown in figure 6-2. This involves simply removing the blank filler panels and replacing them with the ten-card memory unit backplane. Since the new backplane is supplied with the memory voltage distribution cable installed, and there are no interconnections between memory unit backplanes, no wiring steps are involved (other than connection to the new memory power supply). Refer to figure 6-3.

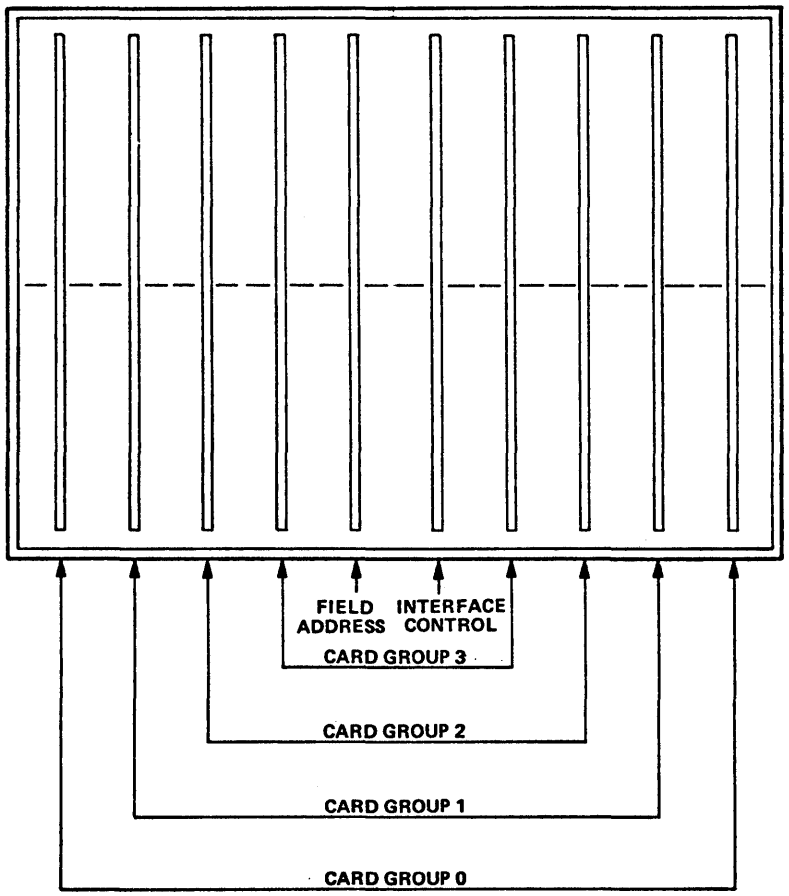


Figure 6-3. Memory Base Unit Installation

Memory Power Supply Installation

The new memory power supply, which accompanies the memory base unit, is to be physically installed in the rear of the system cabinet. As with the backplane assemblies, the power supplies are positioned in accordance with the memory unit number with which they are associated. Figure 6-4 shows the location of a second memory power supply. Third and fourth supplies should be located below the first and second respectively. Note that the second and fourth supplies are installed upside down with respect to the first and third.

Cabling

AC power is supplied to each memory supply by way of two ac cables. These are AC Power/ Memory power cable #1 (2209 5210) and AC Power Extension (2209 7679). Each pair of cables should be joined together (end-to-end), then connected between the ac distribution box and a memory power supply. Refer to figure 6-5.

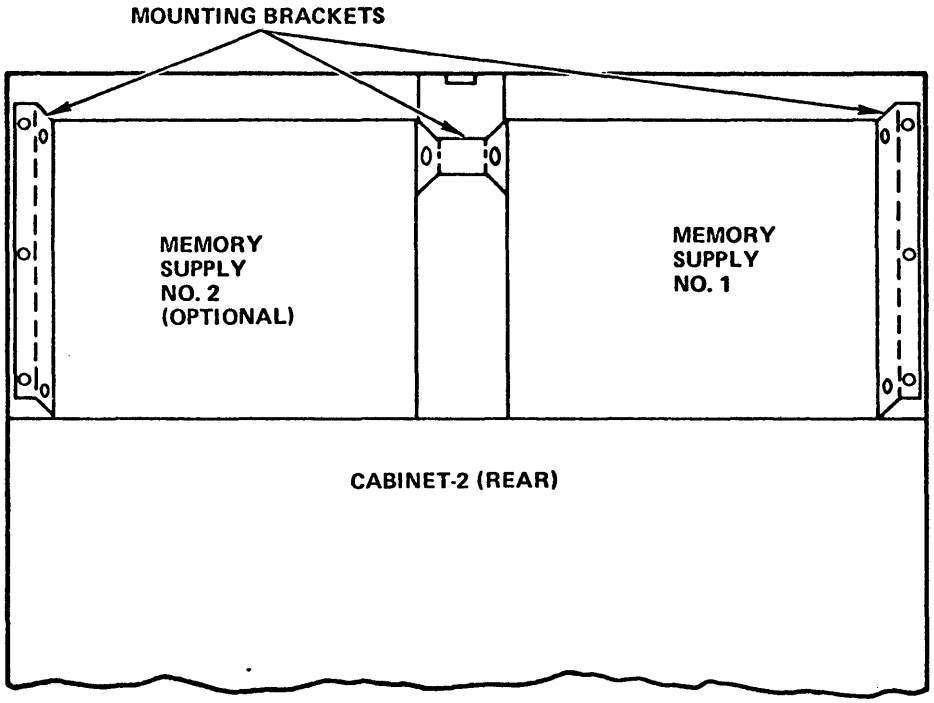


Figure 6-4. Memory Power Supply Installation

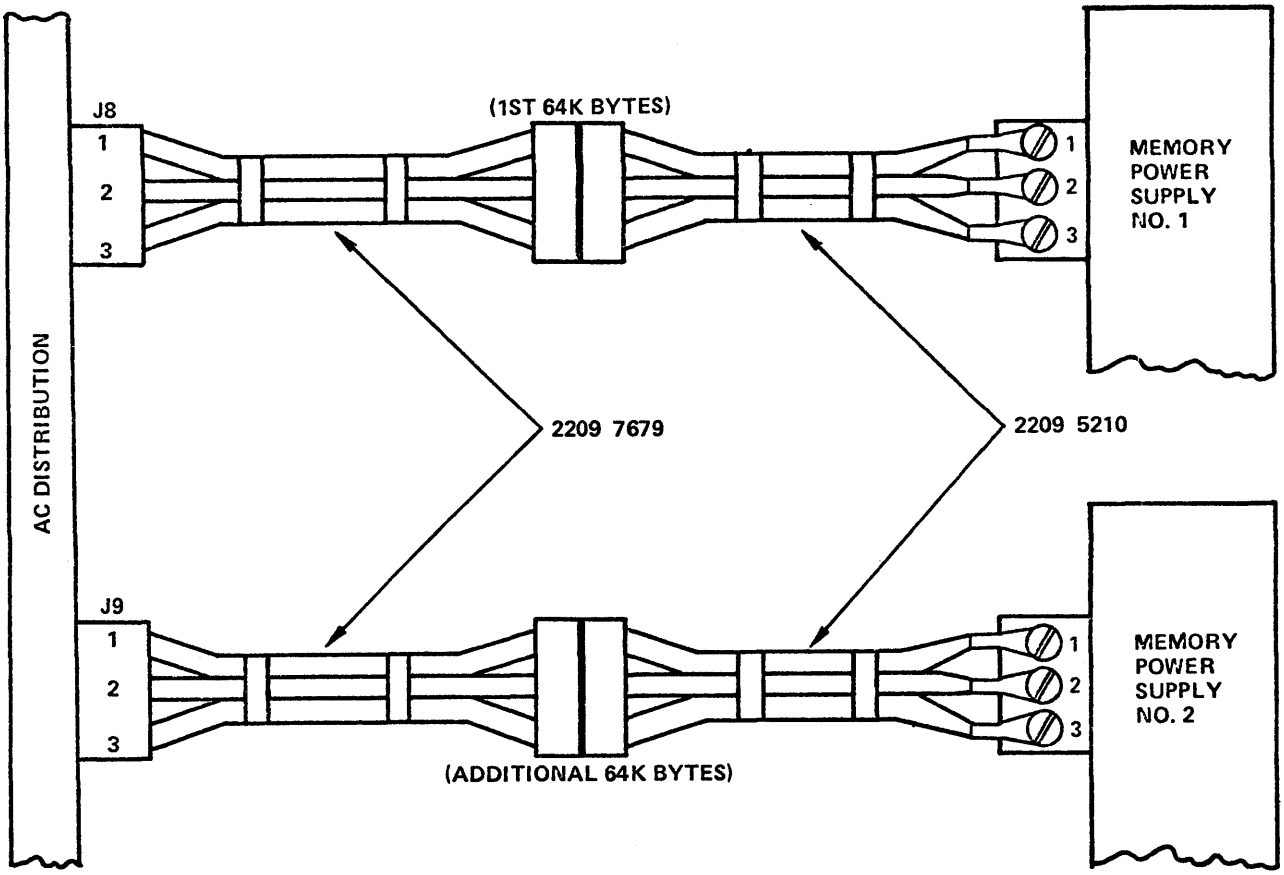


Figure 6-5. AC Connection

The Memory Voltage Distribution Cable (2204 5285), which is permanently attached to the memory base backplane, terminates in a connector (P2) that mates with the connector (J2) on the Memory Voltage Cable (2207 8521). Connect these cables as shown in figure 6-6. Refer to table 6-14 for the purposes of the various voltage and sense lines.

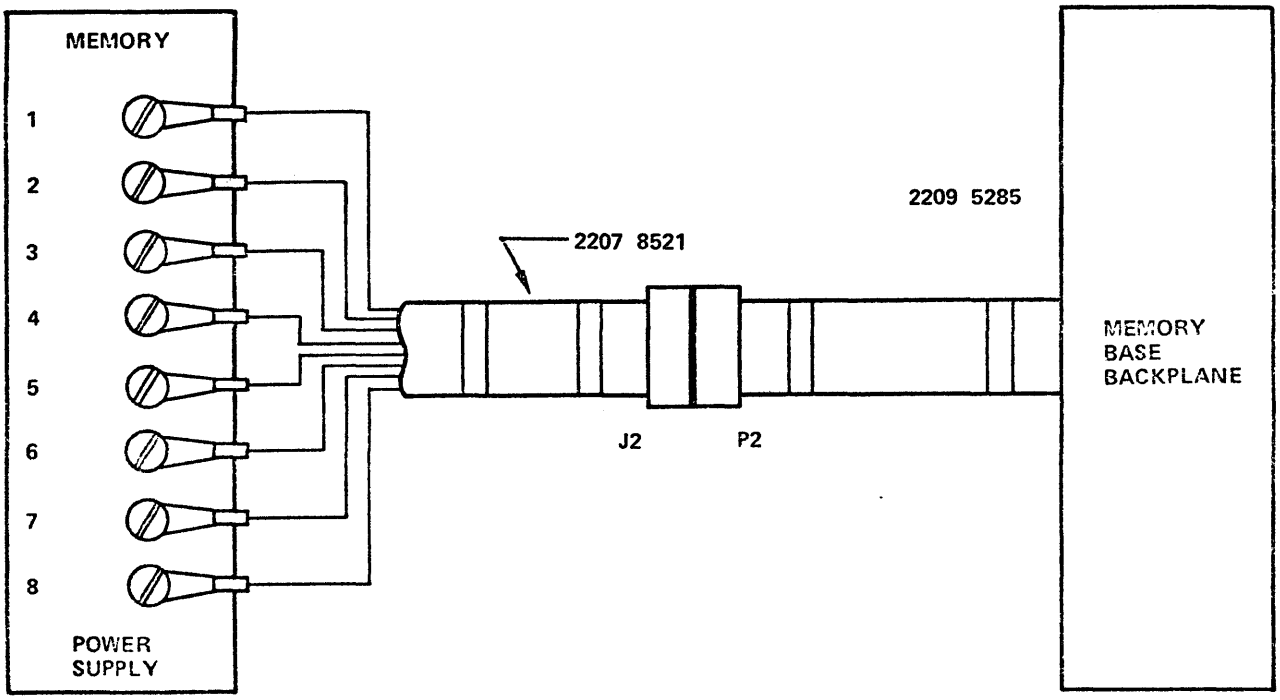


Figure 6-6. Memory Voltage Cable Assembly

Table 6-14. Voltage Distribution

Connector Pin Number (J2 or P2)	Wire Identification			S-Memory Backplane Pin Identification (Reference)
1	+23 VDC	Sense	A	YB0A
5	+23 VDC	Output	B	YC6A & YD0A
9			C	YA6A & YB0A
13			D	YA0A & YA3A
17			E	YC0A & YC3A
2	+19 VDC	Sense	F	YB1A
6	+19 VDC	Output	G	YC1A & YC4A
10			H	YC7A & YD1A
14			J	YA1A & YA4A
18			K	YA7A & YB1A
3	Logic Gnd	Sense	L	XB4W
7	Logic Gnd	Output	M	YC4W
11			N	YB4D
4	-5 VDC	Sense	P	XB0Z
8	-5 VDC	Output	R	XA6Z & XB0Z
12			S	XA0Z & XA3A
16			T	XC0Z & XC3Z
20			U	XC6Z & XD0Z

Memory Interfacing

The FA and IC cards of all memory units interface with the port interchange in exactly the same manner, being connected together in "daisy chain" fashion. This means that all data and control lines are connected to the memory units in parallel, with terminations being used at the end unit only. Accordingly, a different frontplane cable assembly is needed depending upon the number of memory units installed. The frontplane cabling configurations for the various memory sizes are shown in table 6-15.

Table 6-15. B 1720 Memory Frontplane Cabling

Memory Size	Cable P/N	From	To(From)	To(From)
		(Cable Kit-1) 2208 2762		
Up to 64K	2209 9675	PI CX$	M0FAX$	
	2209 9642	PI AX#	M0FAY$	
	2209 9659	PI AY$	M0ICX$	
	2209 9634	PI AY#	M0ICX#	
	2209 9667	PI BY#	M0ICY$	
		(Cable Kit-2) 2208 2770		
72K to 96K	2209 9626	PI CX$	M0FAX$	M1FAXS
	2209 9592	PI AX#	M1FAY$	M0FAY$
	2209 9618	PI AY$	M0ICX$	M1ICX$
	2209 9584	PI AY#	M1ICX#	M0ICX#
	2209 9600	PI BY#	M0ICY$	M1ICY$
Coaxial Cable	2210 7031	M0 FA Lower Doghouse	M0 FA-1 Upper Doghouse	M2 FA-2 Lower Doghouse

Memory Size	Cable P/N	From	To	To	To
		(Cable Kit-3) 2208 2788			
108K to 160K	2210 4210	PICX$	M0FAX$	M1FAX$	M2FAX$**
	2210 4228	PIAY$	M0ICX$	M1ICX$	M2ICXS**
	2210 4236	PIBY#	M0ICY$	M1ICY$	M2ICY$**
	2210 4244	PIAX#*	M2FAY#	M1FAY$	M0FAY$**
	2210 4251	PIAY#*	M2ICX#	M1ICX#	M0ICX#
Coaxial Cable	2211 2510	M1 Upper Doghouse	M2 Lower Doghouse		

Memory Size	Cable P/N	From	To	To	To	To
		(Cable Kit-4) 2208 2796				
168K to 256K	2210 4160	PICX$	M0FAX$	M1FAX$	M3FAX$	M2FAX$**
	2210 4178	PIAY$	M0ICX$	M1ICX$	M3ICX$	M2ICX$**
	2210 4186	PIBY#	M0ICY$	M1ICY$	M3ICY$	M2ICY$**
	2210 4194	PIAX#*	M2FAY$	M3FAY$	M1FAY$	M0FAY$**
	2210 4202	PIAY#*	M2ICX#	M3ICX#	M1ICX#	M0ICX#**
Coaxial Cable	2211 2528	M3 Upper Doghouse	M2 Lower Doghouse			
	2211 2510	M1 Upper Doghouse	M3 Lower Doghouse	Move from M2 Lower DH (Cable already present)		

* 160Ω Termination
**100Ω Termination
NOTE: PI = Port Interchange. Mn = Memory Unit (Number)

Jumper Chip Configuration

Any change in memory size requires that the jumper chips controlling MAXS and force good parity (in the processor) be set accordingly. Therefore, make the appropriate connections listed below as determined by the final memory size.

a. Processor card K, chip L7 (MAXS): Connect those jumpers which constitute the sum of installed memory. Refer to figure 6-7.

   Example: for 64K, connect E-K; for 56K, connect F-J(32K), G-H(16K), and R-S(8K), such that 32K + 16K + 8K = 56K.

b. Memory control card K (force good parity): Connect those jumpers which constitute the sum of installed memory minus 4K bytes. Refer to figure 6-8.

   Example: for 64K, connect E-K(32K), F-J(16K), G-H(8K), and R-S(4K) such that 32K + 16K + 8K + 4K = 60K.

NOTE

Existing connections which are not part of the new MAXS or force good parity settings should be removed.

MAXS

JUMPER	WEIGHT		CORR. MAIN EXCHANGE BIT
	BYTES OF S-MEMORY	BITS OF S-MEMORY	
A–P (L7)	1024K	8192K	23
B–N (L7)	512K	4096K	22
C–M (L7)	256K	2048K	21
D–L (L7)	128K	1024K	20
E–K (L7)	64K	512K	19
F–J (L7)	32K	256K	18
G–H (L7)	16K	128K	17
R–S (L7)	8K	64K	16
A–P (L8)	4K	32K	15

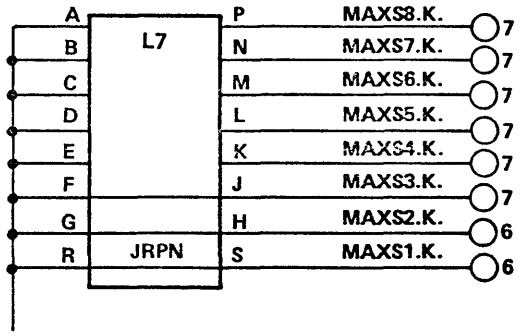


Figure 6-7. MAXS Jumper Chip Wiring
(56K Configuration Shown)

FORCE GOOD PARITY

JUMPER	BYTES OF S-MEMORY	BITS OF S-MEMORY
A–P (L1)	512K	4096K
B–N (L1)	256K	2048K
C–M (L1)	128K	1024K
D–L (L1)	64K	512K
E–K (L1)	32K	256K
F–J (L1)	16K	128K
G–H (L1)	8K	64K
R–S (L1)	4K	32K
A–P (L4)		

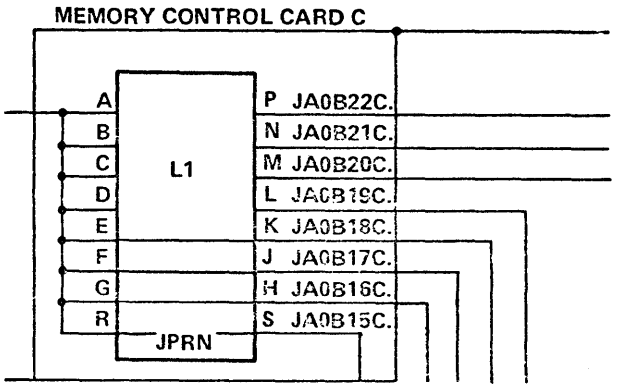


Figure 6-8. Force Good Parity Jumper Chip Wiring
(64K Configuration Shown)

The installation of a new memory base unit requires that a jumper chip setting be made within that unit's control logic. This adjustment serves to inform the unit of it's relative position in memory. The jumper chip setting is made at location G2 on each unit's FA card. Figure 6-9 illustrates the configuration required in each of the four possible memory units. Make sure that each unit employed is set up properly.

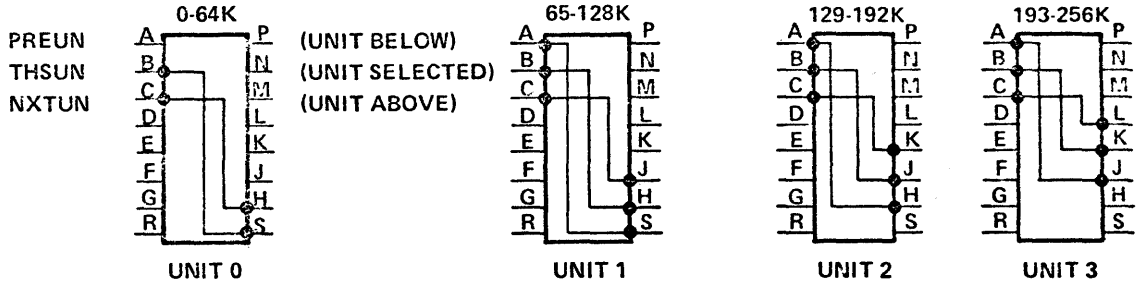


Figure 6-9. Memory Unit Jumper Chip Wiring

This completes the installation procedure for additional memory storage. Memory voltage adjustments and memory timing adjustments should be performed for each memory unit which is new, or has been added to. Refer to Section 4 (Adjustments) of this manual for further information.

M-MEMORY EXPANSION

Expansion of M-memory involves simply placing the new memory card(s) in the next sequential frontplane position(s), then changing the jumper chip wiring which sets the value of MAXM. Refer to figure 6-10 for M-memory card locations. Additional 1K M-memory storage cards may be ordered by part number 2204 8847.

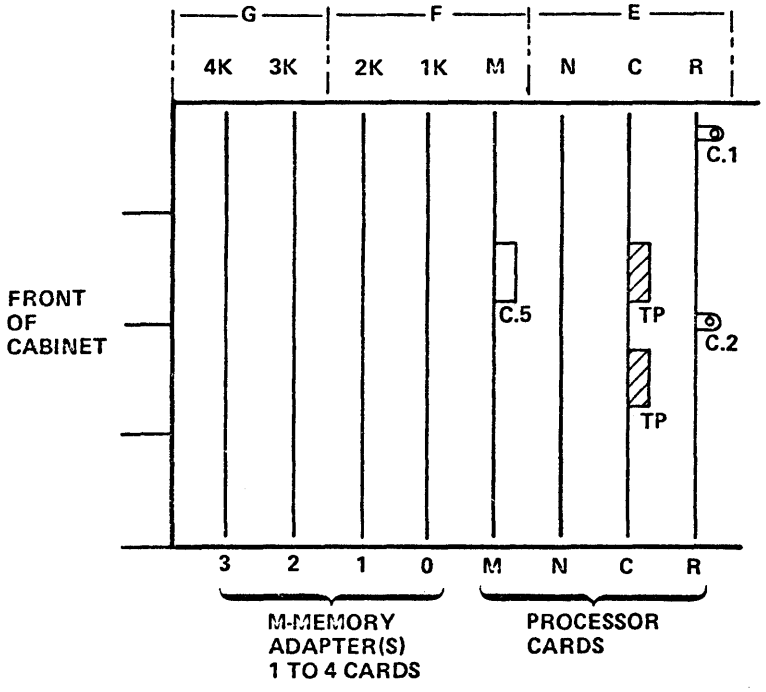


Figure 6-10. M-Memory Card Locations
(Frontplane View)

The value of MAXM is changed by modifying the jumper configuration at location L8 on processor card K. The connections made should correspond to the sum of installed M-memory. Example: for 3K (3 memory cards), connect F-J and C-H. Refer to table 6-16.

Table 6-16. MAXM Jumper Wiring

Jumper	Words of M-Memory	Bits of M-Memory	Corresponding MEX Bit
E-K	4K	64K	12
F-J	2K	32K	11
C-H	1K	16K	16

cut along dotted line

cut along dotted line

BURROUGHS CORPORATION
DATA PROCESSING PUBLICATIONS
REMARKS FORM

TITLE: B1700 SERIES CENTRAL SYSTEM TECHNICAL MANUAL

FORM: 1066941
DATE: 10/75

CHECK TYPE OF SUGGESTION:

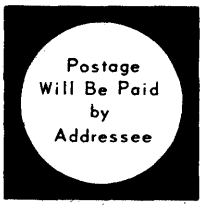
☐ ADDITION ☐ DELETION ☐ REVISION ☐ ERROR

GENERAL COMMENTS AND/OR SUGGESTIONS FOR IMPROVEMENT OF PUBLICATION:

FROM: NAME _____ DATE _____
TITLE _____
COMPANY _____
ADDRESS _____
_____

STAPLE

FOLD DOWN SECOND FOLD DOWN



BUSINESS REPLY MAIL
First Class Permit No. 817, Detroit, Mich. 48232

Burroughs Corporation
Burroughs Place
Detroit, Michigan 48232

attn: Systems Documentation
Technical Information Organization, TIO – Central

FOLD UP FIRST FOLD UP